А. В. Пантелеев
М. М. С. Каранэ

МУЛЬТИАГЕНТНЫЕ И БИОИНСПИРИРОВАННЫЕ МЕТОДЫ ОПТИМИЗАЦИИ ТЕХНИЧЕСКИХ СИСТЕМ

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

А. В. Пантелеев, М. М. С. Каранэ

МУЛЬТИАГЕНТНЫЕ И БИОИНСПИРИРОВАННЫЕ МЕТОДЫ ОПТИМИЗАЦИИ ТЕХНИЧЕСКИХ СИСТЕМ

Монография

Москва

Издательство «Доброе слово и Ко»

2024

УДК 517.977
ББК 22.14
П 16

А в т о р ы:
Пантелеев Андрей Владимирович – доктор физико-математических наук, профессор, заведующий кафедрой «Математическая кибернетика» Московского авиационного института (национального исследовательского университета);
Каранэ Мария Магдалина Сергеевна – ассистент кафедры «Математическая кибернетика» Московского авиационного института (национального исследовательского университета).

Р е ц е н з е н т ы:
Босов А. В. – доктор технических наук, главный научный сотрудник (Федеральный исследовательский центр «Информатика и управление» Российской академии наук);
Куравский Л. С. – доктор технических наук, профессор, лауреат Премии Правительства РФ в области образования, Почетный работник науки и техники РФ, декан факультета информационных технологий (Московский государственный психолого-педагогический университет).

Пантелеев А. В., Каранэ М. М. С.
П16 **Мультиагентные и биоинспирированные методы оптимизации технических систем**: монография / А. В. Пантелеев, М. М. С. Каранэ. – М. : Издательство «Доброе слово и Ко», 2024. – 336 с.

ISBN 978-5-6050159-7-0

В книге изложены новые мультиагентные и биоинспирированные метаэвристические методы глобальной условной оптимизации. Приведены стратегии поиска условного экстремума функций многих переменных, пошаговые алгоритмы, результаты решения модельных примеров и прикладных задач оптимизации технических систем. Описано применение мультиагентных и биоинспирированных методов оптимизации в задачах поиска оптимального программного управления дискретными и непрерывными, стохастическими и детерминированными динамическими системами, а также систем с обратной связью в условиях неопределенности задания начальных условий движения и неполной информации о параметрах объекта управления.

Для студентов и аспирантов технических вузов и университетов, а также инженеров, интересующихся проблемами глобальной оптимизации.

УДК 517.977
ББК 22.14

ISBN 978-5-6050159-7-0

ВВЕДЕНИЕ

Отличительной особенностью большинства актуальных научно-технических задач проектирования современных образцов ракетно-космической и авиационной техники является поиск рациональных решений в многомерном пространстве альтернатив. Показатели качества сравниваемых вариантов, как правило, описываются нелинейными зависимостями и оцениваются при помощи сложных моделирующих алгоритмов, что обуславливает высокую трудоемкость вычислений при решении задач оптимизации. Применение классических численных методов поиска экстремума многоэкстремальных функций со сложным рельефом поверхностей уровня во многих прикладных задачах является малоэффективным [2, 3, 46, 55, 58].

В настоящее время существуют два основных подхода к формированию численных методов оптимизации.

Первый подход связан с реализацией детерминированных и стохастических процедур поиска экстремума при наложении различных, иногда весьма существенных ограничений на свойства и структуру целевой функции. Достоинством этой группы методов является наличие доказательств сходимости, а в некоторых случаях гарантий обеспечения желаемой скорости сходимости. К недостаткам можно отнести достаточно высокие требования к параметрам постановки задачи, которые могут быть не выполнены в прикладных задачах оптимизации или проверка которых затруднительна в силу ее сложности.

При этом применяются следующие методы и подходы:

а) численные алгоритмы удовлетворения необходимых условий условного или безусловного экстремумов [2–4, 14, 55],

б) методы штрафов (внешних штрафов, барьерных функций, комбинированный метод штрафов, метод множителей, метод точных штрафных функций) [2–4, 46];

в) методы возможных направлений (метод проекции градиента, метод Зойтендейка) [3, 46];

г) методы, использующие идеи моментов и адаптацию: быстрый градиентный метод (БГМ) (метод Ю.Е.Нестерова); адаптивный быстрый градиентный метод; ускоренный градиентный метод Нестерова (Nesterov Accelerated Gradient, NAG); метод стохастического градиентного спуска (Stochastic Gradient Descent, SGD), классический метод моментов (Classical Momentum), ускоренный градиентный метод Нестерова (Nesterov Accelerated Gradient, NAG), метод адаптивного градиента (Adaptive Gradient, AdaGrad), метод скользящего среднего (Root Mean Square Propagation, RMSProp), метод адаптивной оценки моментов (Adaptive Moment Estimation, Adam), модификация метода Adam (Adamax), ускоренный по Нестерову метод адаптивной оценки моментов (Nesterov–accelerated Adaptive Moment Estimation, Nadam), метод скользящего среднего с адаптацией шага (AdaDelta); метод стохастического градиента с уменьшенной дисперсией (Stochastic Variance Reduced Gradient (SVRG); ускоренный метод среднего стохастического градиента (SAGA) [27–29, 55, 65, 149];

д) методы на основе универсального градиентного спуска [7];

е) диагональные методы [58, 151, 152];

ж) интервальные методы [3, 34, 55, 80, 102, 106, 141];

з) методы второго порядка (метод Ньютона, метод Ньютона–Рафсона, метод Левенберга–Марквардта) и квазиньютоновские методы: метод Дэвидона–Флетчера–Пауэлла (Davidon–Fletcher–Powell, DFP), метода Бройдена–Флетчера–Голдфарба–

Шенно (Broyden, Fletcher, Goldfarb, Shanno, BFGS), метод Бройдена–Флетчера–Голдфарба–Шенно с ограниченной памятью (limited memory BFGS), метод Ньютона–Гаусса [2–4, 46];

и) модификации градиентных методов: пакетный метод градиентного спуска (Batch Gradient Descent, Vanilla Gradient Descent); метод стохастического градиентного спуска (Stochastic Gradient Descent, SGD); минипакетный метод градиентного спуска (Mini-batch Gradient Descent); метод среднего стохастического градиента (Stochastic Average Gradient, SAG) [55, 149].

Второй подход связан с применением ***метаэвристических*** методов оптимизации. Описание этих, а также других методов глобальной оптимизации, можно найти в [8, 13–15, 20, 27–29, 55, 59, 72, 75, 76, 80, 99–101, 118, 123, 145, 151, 155, 156]. Метаэвристические методы объединяют в себе один или более эвристических методов (процедур), опирающихся на стратегию поиска более высокого уровня (отсюда – *мета*). Они способны покидать окрестности локальных экстремумов и выполнять достаточно полное исследование множества допустимых решений. Метаэвристические методы оптимизации позволяют найти решение «высокого качества» за приемлемое (с практической точки зрения) время. В отличие от классических методов оптимизации метаэвристические методы могут применяться в ситуациях, когда практически полностью отсутствует информация о характере и свойствах исследуемой функции. К недостаткам данной группы методов следует отнести отсутствие (как правило) доказательства сходимости и гарантий близости получаемого результата к точке (точкам) глобального экстремума.

Классификация метаэвристических методов в настоящее время носит условный характер, поскольку характерные группы методов базируются на сходных идеях, и имеется устойчивая тенденция к созданию гибридных алгоритмов, объединяющих в себе сразу несколько хорошо зарекомендовавших себя алгоритмов. Кроме того, один и тот же алгоритм может принадлежать сразу к нескольким группам.

Можно выделить следующие группы метаэвристических методов оптимизации:

а) эволюционные методы,

б) методы «роевого» интеллекта,

в) методы, имитирующие физические процессы,

г) биоинспирированные методы,

д) мультиагентные методы,

е) мультистартовые методы,

ж) методы на основе математических операций.

Эволюционные методы (Evolutionary Methods) [69, 75, 80, 93, 99, 100, 101, 145, 153] имитируют процесс природного развития популяции особей – эволюцию. В основе эволюционных методов лежат принципы, заимствованные из биологии и генетики. Основная идея эволюционных методов состоит в создании популяции особей (индивидов, клеток). В задаче оптимизации каждая особь соответствует одному из возможных решений. Для поиска наилучшего решения используется значение целевой функции или связанной с ней функции приспособленности. Значение функции приспособленности показывает, насколько хорошо подходит особь в качестве решения задачи. Для обеспечения процесса эволюционного поиска к текущей популяции применяются основные генетические операции: селекция, скрещивание, мутация, клонирование, в результате которых генерируется новая популяция при помощи добавления новых особей с лучшими значениями функции приспособленности и удаления старых.

Подобно другим метаэвристическим методам, эволюционные методы не гарантируют обнаружения глобального решения, но они успешно работают, когда требуется найти достаточно «хорошее» решение за приемлемое время. К данной группе методов относятся: генетические алгоритмы с бинарным кодированием и с вещественным кодированием (Genetic Algorithms); методы, имитирующие иммунные системы организмов – методы искусственных иммунных систем (Artificial Immune Systems); метод рассеивания (Scatter Search); эволюционная стратегия преобразования ковариационной матрицы (Covariance Matrix Adaptation Evolution Strategy); метод динамических сеток (Variable Mesh Optimization); метод дифференциальной эволюции (Differential Evolution); метод случайного поиска с последовательной редукцией области исследований (Luus–Jaakola); меметический алгоритм (Memetic Algorithm) и др.

Методы «роевого» интеллекта [76, 80, 114] используют эффект взаимодействия членов популяции, которые обмениваются информацией и формируют стратегию своего поведения на основании сравнения с лидером популяции, лидером среди соседей, памяти о своей наилучшей позиции, подражании случайно выбранному соседу и т.д. В данную группу относят: метод частиц в стае (Particle Swarm Optimization Strategy); метод муравьиных колоний (Ant Colony Optimization); метод имитации поведения бактерий (Bacterial Foraging Optimization); метод пчелиных колоний и метод искусственной пчелиной колонии (Bees Algorithms, Artificial Bee Colony); метод, имитирующий поиск группой людей (Human Group Optimization Algorithm) и др.

Методы, имитирующие физические процессы [80, 100, 101]**,** основаны на использовании разнообразных физических явлений и закономерностей. К ним относятся: метод гравитационной кинематики (Central Force Optimization); метод имитации отжига (Simulated Annealing); адаптивный метод имитации отжига (Adaptive Simulated Annealing); метод поиска гармонии (Harmony Search); метод, использующий закон электромагнетизма (Electromagnetism-like Mechanism); метод спиральной динамики (Spiral Dynamic Algorithm); метод стохастической диффузии (Stochastic Diffusion Search); метод большого взрыва-большого сжатия (Big Bang–Big Crunch); метод фейерверков (Fireworks Algorithm); метод взрыва гранат (Grenade Explosion Method) и др.

Биоинспирированные методы это методы, порожденные природой [8, 13, 20, 54, 72, 85, 126, 127, 148, 154–156]. В их основе лежит наблюдение за поведением наземных и подводных растений и животных, летающих насекомых и птиц, микроорганизмов, анализ протекания физических и химических процессов, социального поведения людей.

Можно условно выделить шесть категорий биоинспирированных алгоритмов.

1. *Имитирующие взаимодействие наземных растений, насекомых и животных*: муравьиных колоний (Ant Colony Optimization, Ant Lion Optimizer), пчелиных колоний (Artificial Bee Colony), светлячков (Firefly Algorithm, Glowwarm Swarm Optimization), лягушек (Shuffled Frog-Leaping Algorithm), табуна лошадей (Horse herd optimization, Wild Horse Optimizer), стай львов (Lion Pride Optimization Algorithm), слонов (Elephant Herding Optimization, Elephant Search Algorithm), змей (Snake Swarm Optimizer), обезьян (Spider Monkey Optimization, Monkey Search, Chimp Optimization Algorithm), серых волков (Grey Wolf Optimizer), лис (Red Fox Optimization), белок (Squirrel Search Algorithm), кошек (Cat Swarm Optimization), кур (Chicken Swarm Optimization), ящериц (Artificial Lizard Search Optimization, Chameleon Swarm Algorithm), пингвинов (Emperor Penguin Optimizer, Emperor Penguins Colony), гиенн (Spotted Hyena Optimizer), койотов (Coyote optimization), кузнечиков (Grasshopper

Optimization Algorithm), жуков (Pity Beetle Algorithm), саранчи (Locust Swarm Optimization), тараканов (Cockroach Swarm Optimization), земляных червей (Earthworm Optimization Algorithm), слизняков (Slime Mold Algorithm), кротов (Blind Naked Mole Rats) размножение декоративных цветов, сорняков и деревьев (Invasive Weed Optimization, Tree Seed Algorithm, Vegetation Evolution, Dandelion Optimizer), рост лесного массива (Forest Optimization), опыление цветов (Flower Pollination Algorithm), миграция животных (Self-Organizing Migrating Algorithm, Animals Migration Optimization, Biogeography-based Optimization).

2. *Имитирующие взаимодействие подводных растений и животных*: стай рыб (Fish School Search, Fish Electrolocation Optimization), стай криля (Krill Herd), стай горбатых китов (Whale Optimization Algorithm), морских львов (Sea Lion Optimization), дельфинов (Dolphin Echolocation), морских хищников и водоплавающих (Marine Predators Algorithm, Salp Swarm Optimization Algorithm), речного окуня (Perch School Search Algorithm), водомерок (Water Strider Algorithm), рост коралловых рифов (Coral Reefs Optimization), белух (Beluga Whales Algorithm).

3. *Имитирующие взаимодействие летающих насекомых и птиц*: стай стрекоз (Dragonfly Algorithm), бабочек (Butterfly Optimization Algorithm, Monarch Butterfly Optimization), орлов (Eagle Strategy, Golden Eagle Optimizer, Bald Eagle Search), коршунов, ястребов (Harris Hawks Optimization), голубей (Pigeon Inspired Optimization), воробьев (Pigeon-Inspired Optimization), мух (Drosophila Food Search Optimization, Fruit Fly Optimization, Mayfly optimization algorithm), мотыльков (Moth Search Algorithm, Moth Flame Optimization), москитов (Mosquito host-seeking algorithm) кукушек (Cuckoo Search), летучих мышей (Bat–Inspired Algorithm, Dynamic Virtual Bats Algorithm), сов (Owl Search Algorithm), чаек (Seagull Optimization Algorithm), колибри (Hummingbird's Optimization Algorithm), синиц (Tomtit Flock Optimization), ворон (Crow Search Algorithm, Raven Roosting Optimization), миграции птиц (Bird Mating Optimizer, Migrating Bird Optimization, Satin Bowerbird Optimizer, Sooty Tern Optimization Algorithm).

4. *Имитирующие взаимодействие микроорганизмов*: питания бактерий (Bacterial Foraging Optimization) и размножения вирусов (Virus Colony Search), искусственных иммунных клеток (Artificial Immune System Algorithm), эволюции популяций на генетическом уровне (Genetic Algorithms, Memetic Algorithms, Differential Evolution, Covariance Matrix Adaptation Evolution Strategy).

5. *Имитирующие физические и химические процессы*: ядерные и химические реакции (Atom Search Optimization, Nuclear Reaction Optimization, Electron Radar Search Algorithm, Artificial Chemical Reaction Optimization Algorithm), механику жидкости (Intelligent Water Drops Algorithm, Vortex Search Algorithm, Flow Regime Algorithm, Archimedes' Optimization Algorithm, Water Evaporation Optimization, Water Cycle Algorithm) и твердых тел (Vibrating Particles System Algorithm, Kepler Optimization Algorithm, Crystal Structure Algorithm), влияние ветра на окружающую среду (Wind Driven Optimization), гравитационное взаимодействие системы масс (Central Force Optimization, Gravitational Search Algorithm, Space Gravitation Optimization), электромагнитное взаимодействие заряженных тел (Electromagnetism-like Mechanism, Electromagnetic Field Optimization, Magnetic Charged System Search, Magnetic Inspired Optimization, Ions Motion Optimization), оптические эффекты (Optics Inspired Optimization, Ray Optimization), термодинамические эффекты (Simulated Annealing, Adaptive Simulated Annealing, Thermal Exchange Optimization, States of Matter Search, Kinetic Gas Molecules Optimization, Henry Gas Solubility Optimizer), изменения во вселенной (Big

Bang-Big Crunch, Galaxy-based Search Algorithm, Multi-verse Optimizer, Black Hole Algorithm), механику взрывов (Fireworks Algorithm, Grenade Explosion Method).

6. *Имитирующие социальное взаимодействие людей*: реализацию мозгового штурма (Brain Storm Optimization), выработку иммунитета к коронавирусу (Coronavirus Herd Immunity Optimization), конкуренцию между империями (Imperialist Competitive Algorithm), поиск группой людей (Human Group Optimization Algorithm), распространение информации путем стохастической диффузии (Stochastic Diffusion Search), процессы изучения и обучения (Teaching–Learning Optimization, Training–Based Optimizer, Driving Training-Based Optimization, Teaching-based Optimization), распространение знаний (Gaining–Sharing Knowledge-based Algorithm, Cultural Evolution Algorithm), поведение индивидуумов в группе (Particle Swarm Optimization, Society and Civilization Optimization Algorithm, Human Mental Search, Poor and Rich Optimization), поведение людей при поиске и спасении (Search and Rescue Optimization), расследованиях криминала (Forensic-Based Investigation Optimization), поиска в очереди (Queuing Search Algorithm), поведение следопыта (Pathfinder Algorithm).

7. *Имитирующие спортивные, настольные, компьютерные игры*: Football Game Inspired Algorithm, Volleyball Premier League Algorithm, League Championship Algorithm, Puzzle Optimization Algorithm, Tug-of-war Optimization.

Общие принципы формирования биоинспирированных алгоритмов оптимизации изложены в разд. 2.2. Предложенные авторами метод, имитирующий поведение стаи речного окуня, и метод, имитирующий поведение стаи синиц, являются итогом наблюдения за процессами охоты окуней за более мелкими рыбами и процессами добывания пищи стаей синиц под управлением вожака стаи (глава 2).

Мультиагентные методы основаны на предположении, что каждое допустимое решение задачи является агентом. Агенты могут объединяться в группы, и их поведение, как правило, определяется некоторым законом управления их траекториями поиска на множестве допустимых решений. При этом движение агентов описывается некоторой математической моделью различной степени сложности. На каждой итерации агенты каждой группы двигаются под действием характерного для этой группы закона управления. Агенты обладают памятью о своих наилучших результатах, обмениваются информацией друг с другом, двигаются к лидеру на текущей итерации, изменяют свою роль в процессе поиска. Общие принципы формирования мультиагентных алгоритмов оптимизации изложены в разд. 1.2.

К ним можно отнести: метод, имитирующий поведение стаи криля (Krill Herd); метод, имитирующий поведение стаи рыб в поисках корма (Fish School Search); метод, имитирующий империалистическую конкуренцию (Imperialist Competition); метод интерполяционного поиска; методы, основанные на управлении агентами с помощью линейных регуляторов; методы, основанные на самоорганизующихся миграционных алгоритмах (Self-Organizing Migrating Algorithms) и др.

К мультиагентным методам можно также отнести метод частиц в стае (Particle Swarm Optimization Strategy), в котором скорость движения отдельной частицы (агента) определяется суммарной информацией о предыдущей скорости, расстоянии до глобального лидера стаи и локального лидера в заданной окрестности. Текущая скорость задает новое положение частицы с учетом конечно-разностной аппроксимации первой производной в кинематической связи положения x и скорости V: $\dot{x} = V$.

Аналогичный подход применяется в методе, имитирующем поведение стаи криля (Krill Herd), в котором скорость движения частицы (криля) задается суммой нескольких характерных составляющих.

В методе гравитационной кинематики (Central Force Optimization) новое положение частицы задается не только величиной скорости, но и ускорением a на основе численного интегрирования кинематического уравнения $\ddot{x} = a$.

Предложенные авторами мультиагентные методы, использующие линейные регуляторы, обобщенные ПИД-регуляторы и регуляторы с прогнозирующими моделями для различных групп агентов, опираются на результаты теорий оптимального и терминального управления динамическими системами, описываемыми уравнениями состояния. В этих методах управление движением агентов формируется на основе величины отклонения текущего положения агента и лидера среди агентов. Кроме того, может быть использована информация о первых и вторых производных отклонения, а также интегралов от отклонения на промежутке времени, выделенном на одну итерацию метода.

В стратегии интерполяционного мультиагентного поиска авторами предложено использование кривых Безье, B-сплайнов, кривых Катмулла–Рома для реализации идеи описания различных фронтов исследования, которые определяются текущими положениями конечного числа (трех или четырех) агентов-лидеров популяции.

Мультистартовые методы [15] используют идею многократного запуска алгоритма оптимизации с дальнейшим применением принципов кластеризации**.** К ним можно отнести: жадный адаптивный метод случайного поиска (Greedy Randomized Adaptive Search Procedure); метод направленного табу-поиска (Tabu Search) и др.

Методы на основе математических операций используют арифметические операции (сложение, вычитание, умножение, деление) для организации процесса поиска (Arithmetic Optimization Algorithm, Adaptive Parallel Arithmetic Optimization Algorithm, Dynamic Arithmetic Optimization Algorithm), а также элементарные функции (Sin-Cos Algorithm), принцип золотого сечения (Golden Ratio Optimization Method), теорию фракталов (Stochastic Fractal Search), геометрические идеи (Radial Movement Optimization, Hyper-Spherical Search Algorithm).

Эффективность мультиагентных методов оптимизации, как правило, исследуется на решениях общепринятых типовых задач:

а) оптимизации многоэкстремальных целевых функций со сложным рельефом поверхностей уровня (benchmark functions) [88]. Наиболее популярные из них приведены в приложении (табл. П.1);

б) параметрической оптимизации технических систем: сосуда высокого давления, сварной балки, редуктора, натяжной пружины [74, 94, 147, 150].

Наличие разнообразных методов оптимизации, постоянное появление новых оригинальных методов и модификаций уже известных обосновывается хорошо известной теоремой (No free lunch theorem, NFL theorem) [160]. Суть ее заключается в том, что если алгоритм A превосходит алгоритм B для некоторого класса целевых функций, то должно существовать столь же много других целевых функций, для которых алгоритм B превосходит A. Данная теорема свидетельствует о том, что не существует наилучшего универсального метода, способного эффективно решить все задачи оптимизации. Поэтому выбор метода определяется конкретной задачей оптимизации: видом целевой функции и ограничений, задающих множество допустимых решений.

Данная книга посвящена описанию новых мультиагентных и биоинспирированных методов, предложенных авторами, и их применению в задачах параметрической оптимизации технических систем, включая задачи оптимального управления непрерывными и дискретными динамическими системами.

В каждой главе книги изложены: постановка задачи, стратегия поиска экстремума, алгоритм решения, описание программного обеспечения и решения типовых примеров. Особое внимание уделяется исследованию влияния параметров методов на их эффективность, поскольку каждый из рассмотренных методов, в свою очередь, представляет собой целый класс аналогичных методов со своими особенностями.

Описанные в книге алгоритмы успешно применены при решении задач проектирования аэрокосмических систем и ракетно-космических комплексов, а также при решении разнообразных задач приближенного нахождения оптимального управления непрерывными и дискретными нелинейными динамическими системами, в частности задач управления ракетами класса воздух-воздух и химическими процессами [36, 53].

Описаны приложения мультиагентных и биоинспирированных методов оптимизации в задачах оптимального управления динамическими системами, отличающихся моделями объекта управления, видом функционала качества, способом задания начальных условий движения, наличием внешних воздействий. Предполагается, что на вектор управления наложены параллелепипедные ограничения (значения каждой координаты вектора управления принадлежат заданному отрезку).

Рассмотрены несколько случаев информированности о векторе состояния, определяемые возможностями измерительной системы:

а) информация о векторе состояния отсутствует (ищется оптимальное программное управление);

б) известна только часть координат вектора состояния (ищется оптимальное управление с неполной обратной связью по известным координатам вектора состояния);

в) известна информация обо всех координатах (ищется оптимальное управление с полной обратной связью).

Задача синтеза оптимального управления с неполной обратной связью в большей степени соответствует потребностям практики, так как предполагается, что система управления в текущий момент времени получает мгновенную информацию только о части координат вектора состояния, в то время как значения остальных координат неизвестны. Эта особенность позволяет решать прикладные задачи, в которых в силу свойств измерительной системы достоверная информация о части координат не доступна, а попытки найти оценки посредством синтеза наблюдателей состояния или фильтров не приводят к желаемому результату. Кроме того, возможны ситуации, когда в процессе управления информация о каких-либо координатах вектора состояния может не поступить в силу технических проблем, и наличие законов управления, зависящих от разного набора координат, может оказаться полезным для обеспечения надежности функционирования управляющего комплекса.

В книге описаны решения трех типов задач оптимального управления динамическими системами.

1. Оптимальное управление отдельной траекторией детерминированной системой. При этом начальное состояние системы считается заданным, функционал определен на траектории, порождаемой начальным состоянием и законом управления, удовлетворяющим наложенным ограничениям.

Для нахождения оптимального программного управления отдельной траекторией непрерывной динамической системы, как правило, применяется необходимое условие оптимальности в форме принципа максимума. Задача сводится к нахождению решения соответствующей двухточечной краевой задачи для обыкновенных дифференциальных уравнений [1, 9, 11, 22, 26, 35, 40, 56, 60]. Для дискретных динамиче-

ских систем используются необходимые условия оптимальности, которые сводятся к решению краевой задачи для разностных уравнений [57].

Задачи синтеза управления с неполной обратной связью рассматривались в [52], где приведены достаточные условия оптимальности и получены соотношения для нахождения оптимального управления как непрерывными, так и дискретными системами. Оптимальное управление с неполной обратной связью для начального состояния, принадлежащему некоторому многообразию, размерность которого определяется числом измеряемых координат, также порождает оптимальное программное управление и соответствующую оптимальную траекторию.

Проблема нахождения управления с полной обратной связью связана с решением соответствующего уравнения Беллмана, как для непрерывных, так и для дискретных систем [1, 11, 22, 35, 40, 86, 87, 120]. Для задач сравнительно небольшой размерности получение их численного решения становится проблематичным. Оптимальное управление с полной обратной связью для любого начального состояния порождает оптимальное программное управление и соответствующую оптимальную траекторию, что является несомненным достоинством такого способа решения.

2. Оптимальное управление пучком траекторий детерминированной системы. Предполагается, что в начальный момент состояние системы конкретно не задано, а задается множество начальных состояний, которое при известном законе управления порождает пучок (ансамбль) траекторий. Каждая траектория пучка соответствует одному из начальных состояний из заданного множества. Одним из вариантов критерия качества может быть среднее значение функционала, определенного на отдельной траектории, по множеству начальных состояний. Тогда искомое управление называется оптимальным в среднем.

Задачи управления пучками траекторий непрерывных динамических систем изучались в работах [24, 25, 115]. При этом рассматривалась задача нахождения оптимального программного управления для линейных систем, а также решение задач высокой размерности, в которых движение пучка описывается через эллипсоидальные траектории. Для нелинейных систем, определяющих динамику пучков заряженных частиц, задача исследовалась в публикациях [30–33]. В этих работах предложены необходимые условия оптимальности, как программного управления, так и управления с неполной и полной обратной связью. В качестве численных методов применялись градиентные процедуры, в общем случае позволяющие найти локальный экстремум.

В [62] для описания поведения совокупности траекторий при неопределенной информации о начальном состоянии использовался метод эллипсоидов, а в [30–33, 52, 71, 98] – уравнение Лиувилля (уравнение переноса). В [52] сформулированы достаточные условия оптимальности и определены соотношения для нахождения оптимального управления с неполной обратной связью. В [5, 6, 40] получены необходимые условия оптимальности для нахождения оптимального в среднем и оптимального гарантирующего программного и позиционного управления пучками траекторий для различных классов систем.

В [12, 34, 36, 48] предложены различные способы решения задачи управления пучками траекторий с помощью перехода к задаче параметрической оптимизации и применения методов поиска экстремума функций. В качестве методов оптимизации использовались интервальные методы, меметические алгоритмы, метод, имитирующий поведение стаи серых волков.

Задачи управления пучками траекторий дискретных динамических систем рассматривались в [32, 33], где предложены различные численные процедуры решения.

3. Оптимальное управление стохастической системой. Предполагается, что начальное состояние системы является случайным вектором, описываемым заданным законом распределения, а модель объекта управления учитывает наличие случайных внешних возмущений. Критерием качества является среднее значение функционала, определенного на отдельной траектории, по множеству реализаций (т.е. его математическое ожидание).

Задачи поиска оптимального управления непрерывными стохастическими системами возникают в случаях, когда известна достоверная статистическая информация о случайном внешнем возмущении и случайных начальных состояниях. При этом в качестве математической модели непрерывных систем обычно выступает стохастическое дифференциальное уравнение (в смысле Ито или Стратоновича). Начальный закон распределения, как правило, задается начальной плотностью вероятности. Наиболее часто рассматривается функционал, определенный на множестве реализаций, порожденных внешними возмущениями и начальными условиями. Большинство задач связано с минимизацией математического ожидания классического функционала Больца, определенного на траекториях динамической системы, однако могут рассматриваться и более сложные случаи вероятностных критериев, связанные, например, с максимизацией вероятности пребывания в заданном множестве, или функционала квантили.

Законы управления стохастическими системами формируются на основе анализа математических моделей измерительной системы, которые могут быть как динамическими, так и статическими. Они характеризуют полноту имеющейся информации о координатах вектора состояния, полностью характеризующего текущее положение системы, а также погрешности и помехи, возникающие в ходе измерений. Информация о поступающих измерениях может накапливаться, и тогда обычно формируется алгоритм управления, использующий оценку вектора состояния для выработки сигнала управления. В таких случаях формируется система совместного оценивания и управления с обратной связью, содержащей модуль фильтрации, формирующий оптимальную оценку вектора состояния по какому-либо критерию, и регулятор, обрабатывающий эту оценку [12]. В системах с линейными моделями объекта управления и измерительной системы, качество поведения которых характеризуется значением квадратичного функционала, данная проблема решается с помощью применения принципа разделения. Он заключается в независимом решении задач синтеза оптимального линейного регулятора и оптимального фильтра Калмана. В общем случае проблема в настоящее время относится к нерешенным. Поэтому применяются различные приближенные методы нелинейной фильтрации и поиска управления, зависящего от вектора оцениваемых параметров.

К более простому классу задач относятся проблемы управления с учетом только мгновенной поступающей информации, в которых информация не накапливается. Как правило, при этом закон управления зависит от конечного набора координат вектора состояния, доступных измерению [35, 40, 52]. Частным случаем такой задачи является проблема поиска оптимального управления, зависящего только от времени. В этом случае формируется система управления без обратной связи, а наилучшее значение функционала качества в общем случае больше по сравнению со случаями, в которых используется информация о доступных координатах вектора состояния, что является недостатком такой постановки задачи. Для нахождения оптимального

управления обычно используется принцип максимума для стохастических систем [103]. Его применение связано с решением краевой задачи, образованной двумя дифференциальными уравнениями с частными производными и соответствующими краевыми условиями. Структура оптимального управления находится из условия максимума стохастического гамильтониана. В качестве первого дифференциального уравнения (одновременно уравнения модели объекта) используется уравнение Фоккера–Планка–Колмогорова, описывающего эволюцию плотности вероятности вектора состояния, а в качестве второго уравнения - уравнение для вспомогательной функции. Дифференциальные операторы обоих уравнений являются сопряженными. Применение принципа максимума для нелинейных систем обычно связано с реализацией различных численных методов. Как правило, решение каждого из уравнений ищется в виде разложения по заданной системе базисных функций с неизвестными коэффициентами, которые находятся с помощью градиентных процедур или методов оптимизации, не использующих производные.

Другим частным случаем является проблема нахождения оптимального управления с полной обратной связью, что с прикладной точки зрения является более предпочтительным. Задача сводится к поиску решения уравнения Беллмана для непрерывных стохастических систем, т.е. нелинейного дифференциального уравнения с частными производными второго порядка [65, 86, 87, 95, 97, 116, 125]. При этом решение ищется в виде разложения по базисным системам функций, что принято в спектральном и псевдоспектральном методах [50, 51, 70, 84, 90, 91]. Также могут применяться различные разностные схемы, модифицированные для решения данной проблемы. Альтернативным подходом является поиск обобщенного решения, понимаемого как вязкостное решение. Задача упрощается только для случая линейных систем с квадратичным критерием, когда задача сводится к нахождению решения уравнения Риккати и синтеза оптимального линейного регулятора.

Для синтеза оптимального программного управления стохастическими дискретными системами обычно применяется стохастический дискретный принцип максимума, а для нахождения оптимального управления с полной обратной связью – уравнение Беллмана для дискретных стохастических систем [1, 40, 87, 97]. Для частных случаев дискретных стохастических систем возможно получение решения задачи совместного оценивания и управления [40].

В книге применяется прямой подход к решению задач оптимального управления:

1) для нахождения оптимального программного управления дискретными системами реализуется процедура поиска значений координат расширенного блочного вектора, блоками которого являются векторы управления во все требуемые в постановке задачи моменты дискретного времени;

2) для нахождения оптимального управления непрерывными системами используется параметрическая оптимизация закона управления. Предлагается искать управление в виде функции насыщения, которая гарантирует выполнение параллелепипедных ограничений, наложенных на управление, с помощью разложения по системе базисных функций. При этом структура закона управления должна удовлетворять необходимым условиям оптимальности, соответствующим решаемой задаче. В качестве систем базисных функций могут применяться используемые в спектральном методе, включая нестационарные косинусоиды, многочлены Лежандра, финитные функции нулевого, первого, второго и третьего порядков, радиально-базисные функции [50, 51, 73, 146], а также разложение по полиномам Чебышева, которое часто

употребляется в псевдоспектральных методах [70, 84, 91]. Количество элементов в каждой из систем базисных функций входит в число неизвестных параметров, как и неизвестные коэффициенты разложений. Таким образом, ставится задача оптимизации с целочисленно-непрерывными переменными, для которой предложена последовательная процедура, включающая линейный поиск по ограниченным целочисленным переменным. За ними следует процесс нахождения коэффициентов разложения с помощью мультиагентных метаэвристических алгоритмов оптимизации. Таким образом, ставится и решается задача о выборе наилучших значений параметров, определяющих предлагаемую структуру управления, т.е. задача о поиске наилучшего закона управления в фиксированном классе управлений.

Полученные результаты являются промежуточным итогом исследований, проводимых на кафедре математической кибернетики института компьютерных наук и прикладной математики Московского авиационного института (национального исследовательского университета).

Материалы разделов 1.1–1.6, 1.9 и главы 3 написаны профессором, доктором физико-математических наук А.В. Пантелеевым совместно с М.М.С. Каранэ, разделов 1.7, 1.8, 4.3–4.5 – совместно с В.М. Ракитянским, главы 2 и раздела 4.3 – совместно с А.А. Колесса.

ГЛАВА 1
МУЛЬТИАГЕНТНЫЕ МЕТОДЫ ОПТИМИЗАЦИИ

1.1. ПОСТАНОВКА ЗАДАЧИ ОПТИМИЗАЦИИ

Дана целевая функция $f(x) = f(x_1, x_2, ..., x_n)$, определенная на множестве допустимых решений $D \subseteq R^n$.

Задача 1. Требуется найти условный глобальный минимум функции $f(x)$ на множестве D, т.е. такую точку $x^* \in D$, что

$$f(x^*) = \min_{x \in D} f(x), \tag{1.1}$$

где $x = (x_1, x_2, ..., x_n)^T$, $D = \{x \mid x_i \in [a_i, b_i], i = 1, 2, ..., n\}$.

Задача 2. Требуется найти условный глобальный максимум функции $f(x)$ на множестве D, т.е. такую точку $x^* \in D$, что

$$f(x^*) = \max_{x \in D} f(x), \tag{1.2}$$

где $x = (x_1, x_2, ..., x_n)^T$, $D = \{x \mid x_i \in [a_i, b_i], i = 1, 2, ..., n\}$.

З а м е ч а н и я.

1. Задача поиска максимума функции $f(x)$ сводится к задаче поиска минимума путем замены знака перед функцией на противоположный: $f(x^*) = \max_{x \in D} f(x) = -\min_{x \in D} [-f(x)]$. Аналогично задача поиска минимума функции $f(x)$ сводится к задаче поиска максимума путем замены знака перед функцией на противоположный: $f(x^*) = \min_{x \in D} f(x) = -\max_{x \in D} [-f(x)]$.

2. Предполагается, что целевая функция является непрерывной.

1.2. ПРИНЦИПЫ ФОРМИРОВАНИЯ МУЛЬТИАГЕНТНЫХ МЕТОДОВ ОПТИМИЗАЦИИ

Мультиагентные методы и алгоритмы оптимизации объединяют оригинальные приемы (эвристики), используемые в различных эволюционных алгоритмах, методах роевого интеллекта, биоинспирированных и мультистартовых алгоритмах. Эти эвристики управляются алгоритмом более высокого уровня (отсюда термин *мета*), как правило, с целью преодоления окрестностей локальных экстремумов и получения хорошего приближения к глобальному экстремуму. Мультиагентные алгоритмы являются подмножеством метаэвристических [20, 54, 80, 100, 101]. Они основаны на процессах, происходящих в среде, которая имеет множество агентов. Агенты обмениваются информацией для того, чтобы найти решение задачи.

Агенты принимают решение, учитывая текущее положение, память о своем предыдущем опыте, информацию о положении абсолютных агентов-лидеров и агентов-лидеров подгрупп, в которые могут объединяться.

Мультиагентные методы, как и большинство метаэвристических алгоритмов, начинают свою работу с этапа генерации популяции агентов на множестве допусти-

мых решений задачи. Как правило, при этом используется равномерный закон распределения с учетом принадлежности множеству допустимых решений. Далее популяция делится на группы различными способами:

а) равномерно без учета соответствующих значений целевой функции;

б) равномерно с учетом расстояний между агентами (как в методе рассеивания [92]);

в) равномерно с учетом значений целевой функции. При этом первоначально производится ранжирование агентов по величине целевой функции. Порядок формирования M групп: наилучшее решение помещается в первую стаю, следующее – во вторую, M-е – в M-ю стаю, $(M+1)$-е – снова в первую стаю и т.д. Такой подход применяется в методе, имитирующем поведение стаи лягушек [81–83].

Агенты (группы агентов) классифицируются по реализуемым функциям, которые они выполняют на основании получаемой на каждой итерации информации. Предлагается использовать следующие типы агентов:

а) локально-оптимальный агент – реализует движение к лидеру группы или лидеру популяции, но только на текущей итерации;

б) консервативный агент – применяет информацию о своем наилучшем результате, полученном за время поиска;

в) автономный агент – использует текущее положение и изучает только окрестность своего положения;

г) сомневающийся агент – исследует окрестность своей позиции при помощи случайных блужданий разнообразной конфигурации;

д) агент-манипулятор – объявляет себя абсолютным лидером популяции и обеспечивает движение остальных членов популяции к своей текущей позиции;

е) командный, коалиционный агент – организует или входит в состав компактных групп, образованных двумя или тремя лидерами. Командные агенты, как правило, генерируют положение нового агента на основании обработки информации о позициях членов команды.

Движение агентов происходит итерационно в соответствии с операциями, применяемыми к каждому из агентов. В результате агенты меняют свое положение. Совокупность аналогичных операций, реализующих управление движением агентов, образует проход. По окончании прохода производится межгрупповой обмен информацией:

а) из текущей популяции агентов удаляются q наихудших агентов, а на их место генерируются столько же новых тем же способом, как и в начале процесса поиска;

б) из каждой группы удаляются наихудшие агенты и заменяются лидерами из других групп;

в) после реализации процедуры из п.а) и упорядочивания по величине целевой функции можно сформировать группы заново. Порядок формирования M групп такой же, как и при делении на группы начальной популяции.

По окончании очередного прохода агенты могут поменять свой тип, характеризующий поведение агента. Опишем сценарии поведения агентов различных типов.

Поведение локально-оптимального агента.

Сценарий 1. Найти наилучшее текущее решение x_{gb} в популяции агентов, а в рамках группы выбрать наилучшее решение x_{best} (положение агента) в группе.

Определить новое положение локально-оптимального агента (сначала реализуется движение к лидеру своей группы):

$$x^{j,k+1} = x_{best} + C\,\mathrm{rand}[0;1] \odot [x^{j,k} - x_{best}],$$

где $x^{j,k}$ – текущее положение j-го агента на k-й итерации, $\mathrm{rand}[0;1]$ – вектор с независимыми случайными координатами, равномерно распределенными на отрезке $[0;1]$, $\odot$ – операция покоординатного произведения векторов по Адамару; для поддержания эффективности поиска можно использовать линейный закон изменения числа C от 1,0 до 2,5 с ростом числа итераций: $C = 1 + 1{,}5\,(k-1)/(ITER-1)$, $ITER$ – максимальное число итераций. Если значение целевой функции в полученном положении улучшилось, то считать, что итерация удачная. Иначе реализовать движение к абсолютному лидеру популяции:

$$x^{j,k+1} = x_{gb} + C\,\mathrm{rand}[0;1] \odot [x^{j,k} - x_{gb}].$$

Если при этом значение целевой функции улучшилось, то считать итерацию удачной, иначе сгенерировать решение случайно на множестве D с помощью равномерного закона распределения.

Сценарий 2. Для реализации движения к агенту-лидеру использовать управляемое движение агентов, минимизируя локально-оптимальный критерий близости. При этом применяется управление, минимизирующее величину критерия через достаточно малый промежуток времени управления [40].

Поведение сомневающегося агента.

Сценарий 1. Используя равномерный закон распределения на отрезке $[0;1]$, сгенерировать число u; если $u > 0{,}5$, то положить $x_i^{j,k+1} = x_i^{j,k} + \mathrm{rand}[0; b_i - x_i^{j,k}] \cdot r$; если $u \le 0{,}5$, то $x_i^{j,k+1} = x_i^{j,k} - \mathrm{rand}[0; x_i^{j,k} - a_i] \cdot r$, где $i = 1,...,n$, $\mathrm{rand}[a,b]$ – случайная величина, равномерно распределенная на отрезке $[a,b]$, r – параметр мутации (рекомендуемое значение 0,01); если $x_i^{j,k+1} \notin [a,b]$, то процедуру повторить.

Сценарий 2. Реализовать процедуру локального линейного поиска [36]. Метод имеет параметр – шаг поиска h.

Шаг 1. Упорядочить переменные x_i, $i = 1,...,n$, агента x случайным образом. Положить $New = false$ – индикатор наличия улучшений, $j = 1$ – номер переменной в упорядоченном списке.

Шаг 2. Пусть на j-м месте в списке стоит i-я переменная. «Просканировать» множество, т. е. вычислить значение функции приспособленности для особей из множества $ls(x,h,i) = \{y \in R^h \mid y = x + khe_i, k \in Z, a \le y \le b\}$, e_i – единичный орт, состоящий из всех нулей и одной единицы на i-м месте; $a = (a_1, a_2, ..., a_n)^T$, $b = (b_1, b_2, ..., b_n)^T$ – векторы соответственно нижних и верхних границ множества D.

Шаг 3. Если в результате сканирования находится лучшее положение агента, то агента x заменить агентом y и положить $New = true$.

Шаг 4. Если $j < n$, то положить $j = j + 1$ и перейти к шагу 2. Если $j = n$, то при $New = true$ перейти к шагу 1, иначе процесс локального линейного поиска завершить.

Поведение консервативного агента. Основывается на учете своего наилучшего результата за все предыдущие итерации и наилучшего результата среди агентов-соседей. При этом может учитываться эффект забывания текущей позиции.

Новое положение находится по формуле, обобщающей используемую в методе частиц в стае [76]:

$$x^{j,k+1} = \omega x^{j,k} + \delta\left[\alpha r_1\left(\hat{x}^{j,k} - x^{j,k}\right) + \beta\omega r_2\left(\hat{x}^{n_j,k} - x^{j,k}\right)\right]\left(m - \left[\frac{nstep}{2}\right] - 1\right),\ m = 1,\ldots,nstep,$$

где α, β – положительные параметры (как правило, принимающие значения около 0,5); r_1, r_2 – случайные величины, равномерно распределенные на отрезке $[0;1]$; $nstep \in [5;15]$, $\hat{x}^{j,k}$ – наилучшая позиция за все выполненные итерации; $\hat{x}^{n_j,k}$ – лучшая позиция среди случайно выбранного целого числа NI^j агентов-соседей на отрезке $[NI_{\min}, NI_{\max}]$; $NI_{\min} = 5, NI_{\max} = 25$; ω – параметр забывания, $\omega \in [0,01;0,9]$; δ – случайное число, равномерно распределенное на отрезке $[-0,5;0,5]$.

Поведение автономного агента. Он использует текущее положение и исследует окрестность своего положения.

Сценарий 1. Генерировать новые положения как в методе распространения сорняков [124]. Количество позиций s^j, исследуемых агентом, определяется линейной зависимостью и находится в диапазоне $[s_{\min}, s_{\max}]$. Они располагаются вокруг агента, согласно нормальному закону распределения. Математическое ожидание определяется текущим положением агента, а среднеквадратическое отклонение – числом k выполненных итераций, начальным $\sigma_{initial}$ и конечным σ_{final} значениями:

$$\sigma_{iter} = \left(\frac{ITER - k}{ITER}\right)^p \cdot \left(\sigma_{initial} - \sigma_{final}\right) + \sigma_{final}.$$

Число исследуемых позиций находится по формуле

$$s^j = \left[s_{\min} + \frac{s_{\max} - s_{\min}}{f_{\max} - f_{\min}} \cdot \left(f^j - f_{\min}\right)\right],\ j = 1,\ldots,NP,$$

где f^j – значение целевой функции для j-го агента, $f_{\max}, f_{\min}$ – максимальное и минимальное значения целевой функции на текущей итерации. Обычно $S_{\min} = 0$, $S_{\max} \in [5;15]$, $\sigma_{final} \in [10^{-6};10^{-1}]$, $\sigma_{initial} \in [10;300]$, $r = 3$. Вырабатываемые позиции должны принадлежать множеству D (если положение не принадлежит множеству D, то процесс генерации продолжается).

Сценарий 2. Найти случайное число NI агентов-соседей на отрезке $[NI_{\min}, NI_{\max}]$; $NI_{\min} = 5, NI_{\max} = 25$ при помощи равномерного закона распределения с нахождением целой части сгенерированного значения. Далее определить «центр масс» решений, соответствующих агентам-соседям, как взвешенное среднее:

$$x_{com} = \left[\sum_{j=1}^{NI} \frac{x^j}{f(x^j) + \mu}\right] / \left[\sum_{j=1}^{Nl} \frac{1}{f(x^j) + \mu}\right],$$

где μ – малое положительное число.

Найти новое положение автономного агента:

$$x_i^{j,k+1} = \beta x_{com} + (1 - \beta) x_i^{j,k} + \text{nrand}\,\frac{\alpha(b_i - a_i)}{k},$$

где β – параметр, определяющий влияние «центра масс» и текущего решения; $\text{nrand} \sim N[0;1]$ – нормальное распределение; α – параметр, ограничивающий область поиска. Если новое положение лучше текущего, произвести замену.

Поведение агента-манипулятора. Используется факт объявления текущим абсолютным лидером агента, положению которого соответствует наилучшее значение целевой функции.

Сценарий 1. Члены популяции агентов реализуют процесс миграции в направлении абсолютного лидера, применяя самоорганизующийся метод миграции [77, 161, 162]. При этом обновляются позиции агентов, участвующих в миграции. Они заменяются наилучшими достигнутыми в процессе миграции позициями.

Сценарий 2. Агенты реализуют управляемое движение к абсолютному лидеру с помощью оптимального регулятора с полной обратной связью [35, 40]. При этом управление, используемое агентом, зависит от текущего отклонения положения агента от положения абсолютного лидера. После реализации очередного прохода информация об абсолютном лидере обновляется и формируется новый закон управления с обратной связью по положению агента, применяемый на протяжении следующего прохода. В качестве критерия близости агента к агенту-лидеру используется величина квадратичного критерия, отражающего как степень отклонения от лидера, так и затраты на процесс управления агентом.

Сценарий 3. Может применяться упрощенный по сравнению со сценарием 2 подход. В этом случае все члены группы используют один и тот же найденный закон управления приближением к лидеру, в котором в конце каждого прохода меняется требуемое положение цели, а именно значения координат нового агента-лидера.

Сценарий 4. Агенты реализуют процедуру, описанную в сценарии 2, только в качестве критерия близости к лидеру применяют функционал обобщенной работы, учитывающий затрачиваемую энергию управляемых сигналов [40]. При этом за счет введения полуопределенного критерия упрощается процедура синтеза управления с обратной связью, и процесс выработки решения агентом становится более оперативным.

Поведение командного, коалиционного агента. Он использует идею обмена информацией между членами подгруппы.

Сценарий 1. В популяции случайным образом выбираются четыре агента. Среди них либо выявляется лидер, либо не выявляется в зависимости от способа параметрической интерполяции. Реализуется поиск внутри выпуклой оболочки, образованной выбранными агентами. При этом генерируется положение нового агента. Такой сценарий может быть применен для поиска в новых подобластях множества допустимых решений [19].

Сценарий 2. Среди агентов, ранжированных по величине целевой функции, выбираются два, три или четыре агента-лидера. Решается задача параметрической интерполяции и находится положение нового агента. Применяется для реализации фронтального поиска решения задачи [19, 44, 130]. Линия фронта образуется лидерами процесса поиска экстремума.

Сценарий 3. В конце каждого прохода из группы выделяется агент, которому соответствует наихудшее значение целевой функции. Он исключается из группы и популяции в целом. На его место в группу включается один из лидеров других групп.

Сценарий 4. В конце каждого прохода из популяции исключается фиксированное число наихудших агентов. Команды далее формируются заново с помощью разных вариантов генерирования начальной популяции агентов.

Процедуры выбора агентов друг другом могут быть реализованы несколькими способами.

1. По информации о величине целевой функции, соответствующей положениям агентов.

2. По информации о репутации агента $r_j \in [0;1]$, изменяющейся в процессе поиска. Она формируется на основании показателя доверия a_{jl} агента с номером j к агенту с номером l. В начале поиска все элементы матрицы $A = (a_{jl})$ полагаются равными нулю. Если движение агента с номером j к агенту с номером l сопровождается улучшением значения целевой функции, то показатель доверия к агенту с соответствующим номером увеличивается:

$$a_{jl}^{New} = a_{jl} + \left| \left[f(x^{j,k+1}) - f(x^{j,k}) \right] / f(x^{j,k}) \right|, \quad j, l = 1,...,NP.$$

Иначе показатель доверия не изменяется: $a_{jl}^{New} = a_{jl}$.

Репутация агента с номером l определяется суммой элементов l-го столбца матрицы A:

$$\overline{r}_l = \sum_{j=1}^{NP} a_{jl}$$

или ее нормированным значением

$$r_l = \frac{\overline{r}_l}{\sum_{j=1}^{NP} \overline{r}_j}, \; l = 1,...,NP.$$

Выбор агентов может производиться методом рулетки с учетом значений репутации агентов.

1.3. ГИБРИДНЫЙ МУЛЬТИАГЕНТНЫЙ МЕТОД ИНТЕРПОЛЯЦИОННОГО ПОИСКА

1.3.1. Стратегия поиска решения

Рассматривается задача (1.1) поиска условного глобального минимума целевой функции $f(x)$ на множестве допустимых решений D, т.е. такой точки $x^* \in D$, что

$$f(x^*) = \min_{x \in D} f(x),$$

где $x = (x_1, x_2, ..., x_n)^T$, $D = \{x \mid x_i \in [a_i, b_i], i = 1, 2, ..., n\}$.

Метод основан на процессах, происходящих в среде, имеющей множество *агентов*. Агенты имеют возможность обмениваться информацией для того, чтобы найти решение задачи.

Стратегия поиска включает *интерполяционный поиск*, который использует несколько точек текущей популяции и сводит задачу нахождения новых решений к задачам одномерной параметрической минимизации, метод роевого интеллекта для минимизации вдоль интерполяционной кривой [64, 89] и самоорганизующийся метод миграции (Self-Organizing Migrating Algorithm, SOMA)[77, 161, 162].

Рассматриваемая целевая функция $f(x)$ называется функцией приспособленности, а вектор параметров $x = (x_1, x_2, ..., x_n)^T$ целевой функции – *агентом*. Предполагаемым решением поставленной задачи оптимизации может быть каждый вектор $x = (x_1, x_2, ..., x_n)^T \in D$.

При решении задачи (1.1) используются конечные наборы $I = \{x^{(j)} = (x_1^j, x_2^j, \ldots, x_n^j)^T, j = 1,2,\ldots,NP\} \subset D$ возможных решений (агентов), называемые популяциями, где $x^{(j)}$ – агент с номером j, NP – размер популяции.

Гибридный мультиагентный метод интерполяционного поиска имитирует эволюцию начальной популяции $I_0 = \{x^{(j)}, j = 1,2,\ldots,NP \mid x^{(j)} = (x_1^j, x_2^j, \ldots, x_n^j)^T \in D\}$ и представляет собой итерационный процесс, исследующий множество D.

Процедура поиска решения начинается с генерации начальной популяции агентов $x^{(j)}$, $j = 1,2,...,NP$ на множестве D, используя равномерный закон распределения.

Первой фазой поиска является *интерполяционный поиск*.

Для его реализации в популяции выбираются четыре агента P_1, P_2, P_3, P_4, среди них лидер $P_1 = x^{(1)}$, а в качестве P_2, P_3, P_4 - три случайных агента популяции, отличных друг от друга и от $x^{(1)}$. Для их обработки используется кривая Безье. Она проходит внутри выпуклой оболочки, образованной выбранными четырьмя точками (агентами). При $t = 0$ она проходит через P_1, а при $t = 1$ через точку P_4. Далее находится решение задачи параметрической оптимизации

$$x^{Bezier\,4} = \arg\min_{t\in[0;1]} f[(1-t)^3 P_1 + 3(1-t)^2 t P_2 + 3(1-t)t^2 P_3 + t^3 P_4],$$

и популяция пополняется новым агентом $x^{Bezier\,4}$.

Для исследования новых областей применяется B-сплайновая кривая, которая формируется по четырем случайным агентам популяции P_1, P_2, P_3, P_4, отличным друг от друга,

$$x^B = \arg\min_{t\in[0;1]} f\left[\frac{1}{2}[-t(1-t)^2 P_1 + (2-5t^2+3t^3)P_2 + t(1+4t-3t^2)P_3 - t^2(1-t)P_4]\right].$$

Она, как правило, не проходит ни через одну точку, лежит в выпуклой оболочке, порожденной четырьмя вершинами. В результате популяция пополняется еще одним новым агентом x^B.

Второй фазой поиска является *миграция популяции*. При этом в популяции выбирается агент-лидер (наилучшее решение) $x^{(1)}$. Все остальные агенты популяции $x^{(j)}, j = 2,...,NP$ двигаются по направлению к лидеру, совершая дискретные шаги, число которых равно *nstep*, причем половина этих шагов делается к лидеру, а затем в том же направлении совершается еще столько же шагов. Новое положение агента популяции определяется наилучшим достигнутым в ходе этого поиска решением. Направление поиска задается вектором *PRTVector*, координаты которого равны нулю или единице. Если координата равна нулю, поиск по данной координате не проводится, а если единице, то проводится. Таким образом, решение задачи ищется по всем координатам вектора *PRTVector*, равным одновременно единице. Положение лидера в процессе миграции не изменяется.

Третьей фазой является *фронтальный поиск*, который служит для уточнения итогового решения задачи. При этом используются интерполяционные кривые, для формирования которых учитывается информация о положении трех или четырех лидеров среди агентов популяции.

Среди агентов популяции $x^{(1)},...,x^{(NP)}$, расположенных по возрастанию соответствующего им значения целевой функции, выбираются три лидера

$P_1 = x^{(1)}, P_3 = x^{(2)}, P_2 = x^{(3)}$, по которым формируется кривая Безье (при $t = 0$ проходит через P_1, а при $t = 1$ через точку P_3). Далее решается задача параметрической минимизации

$$x^{Bezier3} = \arg\min_{t\in[0;1]} f[(1-t)^2 P_1 + 2(1-t)tP_2 + t^2 P_3],$$

а популяция в итоге пополняется новым агентом $x^{Bezier3}$.

Для продолжения поиска выбираются два лидера, через которых проходит интерполяционная кривая Катмулла–Рома (Catmull–Rom), а также два последующих решения: $P_1 = x^{(3)}, P_2 = x^{(1)}, P_3 = x^{(2)}, P_4 = x^{(4)}$ (при $t = 0$ проходит через P_2, а при $t = 1$ через P_3). Далее решается задача параметрической минимизации

$$x^{CR} = \arg\min_{t\in[0;1]} f\left[\frac{1}{2}[-t(1-t)^2 P_1 + (2-5t^2+3t^3)P_2 + t(1+4t-3t^2)P_3 - t^2(1-t)P_4]\right],$$

а популяция пополняется новым агентом x^{CR}.

Для аналогичного поиска используется кривая Безье, формируемая по четырем лидерам популяции $P_1 = x^{(1)}, P_4 = x^{(2)}, P_2 = x^{(3)}, P_3 = x^{(4)}$ (при $t = 0$ она проходит через P_1, а при $t = 1$ через точку P_4). Далее решается задача параметрической минимизации

$$x^{Bezier4} = \arg\min_{t\in[0;1]} f[(1-t)^3 P_1 + 3(1-t)^2 tP_2 + 3(1-t)t^2 P_3 + t^3 P_4],$$

и популяция пополняется новым агентом $x^{Bezier4}$.

Для фронтального поиска может быть использована B-сплайновая кривая, которая формируется по четырем лидерам популяции $P_1 = x^{(1)}, P_4 = x^{(2)}, P_2 = x^{(3)}, P_3 = x^{(4)}$. Кривая не проходит ни через одну выбранную точку, но лежит в выпуклой оболочке, порожденной этими вершинами. Далее находится решение задачи параметрической оптимизации

$$x^{B} = \arg\min_{t\in[0;1]} f\left[\frac{1}{2}[-t(1-t)^2 P_1 + (2-5t^2+3t^3)P_2 + t(1+4t-3t^2)P_3 - t^2(1-t)P_4]\right].$$

В результате популяция пополняется еще одним новым агентом x^{B}.

На первой и третьей фазах требуется провести одномерную минимизацию вдоль параметрических кривых. Для этого используется биоинспирированный метод роевого интеллекта – метод, имитирующий поведение популяции криля [64, 89]. Данный алгоритм основан на результатах анализа поведения стай криля – рачков, внешне напоминающих креветок. Их позиции меняются под действием трех факторов: присутствия других членов популяции, необходимости поиска пищи, случайных блужданий. Обычно движение популяции криля определяется двумя целями: увеличением плотности криля и достижением пищи.

В начале процесса поиска минимума генерируется популяция криля из NP' особей на промежутке $t \in [0;1]$ с помощью равномерного закона распределения. Предполагается, что движение j-го члена популяции криля происходит согласно уравнению $\frac{dx^j}{dt} = V^j$, где x^j – положение, а V^j – скорость, которая складывается из трех составляющих. Первая составляющая определяется влиянием соседей (членов популяции, входящих в некоторую окрестность j-го элемента определенного радиуса), наилучшего элемента во всей популяции и информации о своей старой скорости. Вторая составляющая определяется движением в сторону источника пищи (за него принимает-

ся «центр масс» популяции), информацией о старой скорости в поисках пищи, памятью своего наилучшего результата за все итерации. Третья составляющая имитирует случайные блуждания особи, уменьшающиеся с ростом числа итераций. Для оживления процесса поиска применяются операции скрещивания и мутации, используемые в других эволюционных методах и методе дифференциальной эволюции. Процедура поиска минимума вдоль интерполяционной кривой завершается при достижении заданного числа итераций. В качестве ответа из последней популяции криля выбирается особь, которой соответствует наименьшее значение целевой функции вдоль параметрической кривой, а исходная популяция пополняется новым членом.

Задача одномерной параметрической минимизации может быть решена и классическими методами одномерной минимизации: дихотомии, золотого сечения, Фибоначчи и др. [3, 46].

Гибридный мультиагентный метод интерполяционного поиска заканчивает работу после того, как будет исчерпано заданное число итераций. В последней популяции происходит отбор агентов с наименьшим значением целевой функции. Агент с наименьшим значением целевой функции принимается в качестве приближенного решения.

Общая схема работы гибридного мультиагентного метода интерполяционного поиска представлена на рис. 1.1.

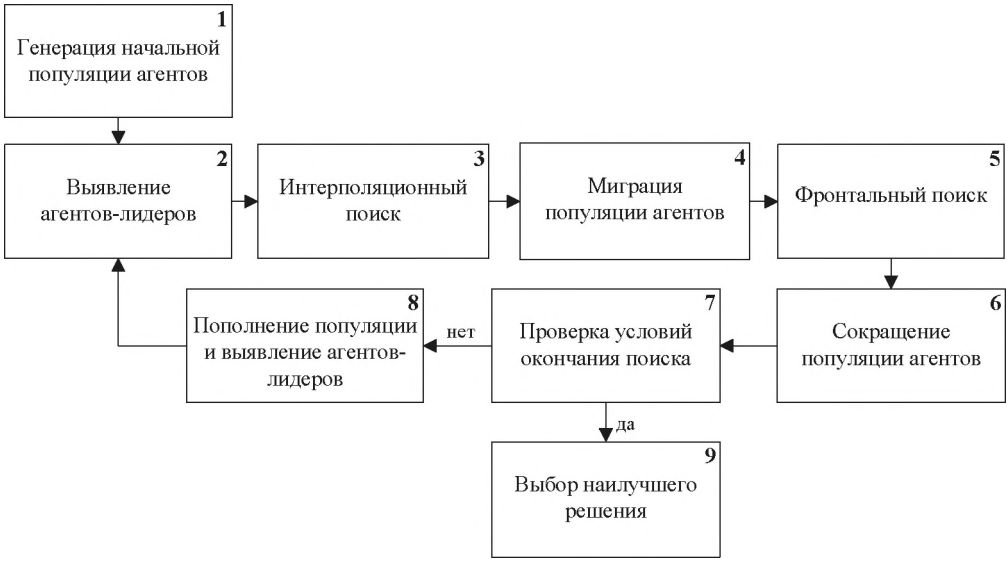


Рис. 1.1. Общая схема работы гибридного мультиагентного метода интерполяционного поиска

1.3.2. Алгоритм решения задачи

Шаг 1. Задать параметры метода:

- количество агентов в популяции NP;
- количество точек M_1, полученных с использованием кривых Безье;
- количество точек M_2, полученных с использованием B-сплайна;

- параметр PRT, определяющий активность поиска по координате;
- количество $nstep$ возможных положений агентов популяции;
- количество b_2 худших агентов для сокращения популяции;
- максимальное число итераций $I_{\max}$.

Положить $it = 1$ (счетчик числа итераций).

Шаг 2. *Формирование начальной популяции.* Генерировать начальную популяцию агентов I_0 на множестве D, используя равномерный закон распределения: $x^{(1)},\ldots,x^{(NP)}$. Вычислить значения целевой функции $f(x^{(1)}),\ldots,f(x^{(NP)})$.

Шаг 3. *Взаимное сравнение агентов популяции и выявление лидеров.* Упорядочить популяцию, состоящую из NP агентов, по величине целевой функции.

Шаг 4. *Интерполяционный поиск.*

Шаг 4.1. Данный шаг выполнить M_1 раз. В текущей популяции выбрать $P_1 = x^{(1)}$ (лучший), а в качестве P_2, P_3, P_4 – три случайных агента популяции, отличных друг от друга и от $x^{(1)}$ (рис. 1.2), и найти решение задачи параметрической оптимизации:

$$x^{Bezier\,4,j} = \arg\min_{t\in[0;1]} f[(1-t)^3 P_1 + 3(1-t)^2 tP_2 + 3(1-t)t^2 P_3 + t^3 P_4], j = 1,\ldots,M_1.$$

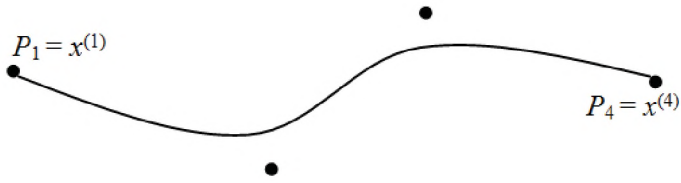


Рис. 1.2. Кривая Безье

Шаг 4.2. Данный шаг выполнить M_2 раз. В текущей популяции выбрать четыре случайных агента популяции P_1, P_2, P_3, P_4, отличных друг от друга (рис. 1.3), и найти решение задачи параметрической оптимизации:

$$x^{B,j} = \arg\min_{t\in[0;1]} f\left[\frac{1}{2}[-t(1-t)^2 P_1 + (2-5t^2+3t^3)P_2 + t(1+4t-3t^2)P_3 - t^2(1-t)P_4]\right], j = 1,\ldots,M_2.$$

Шаг 4.3. Сокращение популяции. Расположить агентов текущей популяции, пополненной $M_1 + M_2$ новыми агентами, по возрастанию соответствующего им значения целевой функции. Оставить только NP лучших агентов.

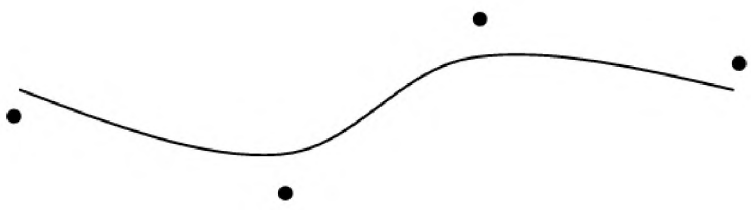


Рис. 1.3. B-сплайн

Шаг 5. *Миграция популяции.*

Шаг 5.1. Для всех $x^{(j)}, j = 2,..., NP$:

– генерировать $PRTVector^{(j)}$ с координатами

$$PRTVector_i^{(j)} = \left\{ \begin{matrix} 1, \text{если } rand_i < PRT \\ 0, \text{иначе} \end{matrix} \right\}, \; rand_i = U[0;1];$$

– найти последовательно возможные положения членов популяции

$$x^{(j),m} = x^{(j)} + \frac{x^{(1)} - x^{(j)}}{\left[\frac{nstep}{2}\right]} m \otimes PRTVector_i^{(j)}, \; m = 0,1,...,nstep,$$

где $\otimes$ – покомпонентное произведение векторов (по Адамару);

– найти наилучшие положения агентов популяции в течение миграции

$$x^{(j),new} = \arg \min_{m=0,1,...,nstep} f(x^{(j),m}), j = 2,...., NP; x^{(1),new} = x^{(1)}.$$

Шаг 5.2. Расположить новых агентов популяции после миграции по возрастанию соответствующего им значения целевой функции.

Шаг 6. *Фронтальный поиск.*

Шаг 6.1. В текущей популяции выбрать $P_1 = x^{(1)}, P_3 = x^{(2)}, P_2 = x^{(3)}$ (рис. 1.4) и решить задачу:

$$x^{Bezier3} = \arg \min_{t \in [0;1]} f[(1-t)^2 P_1 + 2(1-t)tP_2 + t^2 P_3].$$

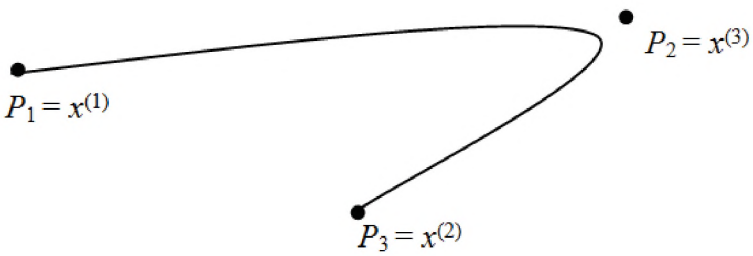


Рис. 1.4. Кривая Безье

Шаг 6.2. В текущей популяции выбрать $P_1 = x^{(3)}, P_2 = x^{(1)}, P_3 = x^{(2)}, P_4 = x^{(4)}$ (рис. 1.5) и решить задачу

$$x^{CR} = \arg \min_{t \in [0;1]} f\left[\frac{1}{2}[-t(1-t)^2 P_1 + (2-5t^2+3t^3)P_2 + t(1+4t-3t^2)P_3 - t^2(1-t)P_4]\right].$$

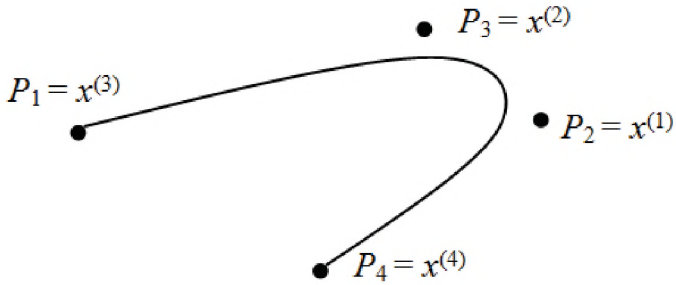


Рис. 1.5. Кривая Катмулла–Рома (Catmull–Rom)

Шаг 6.3. В текущей популяции выбрать $P_1 = x^{(1)}, P_4 = x^{(2)}, P_2 = x^{(3)}, \ P_3 = x^{(4)}$ (рис. 1.6) и найти решение задачи параметрической оптимизации:

$$x^{Bezier\,4} = \arg\min_{t\in[0;1]} f[(1-t)^3 P_1 + 3(1-t)^2 t P_2 + 3(1-t)t^2 P_3 + t^3 P_4].$$

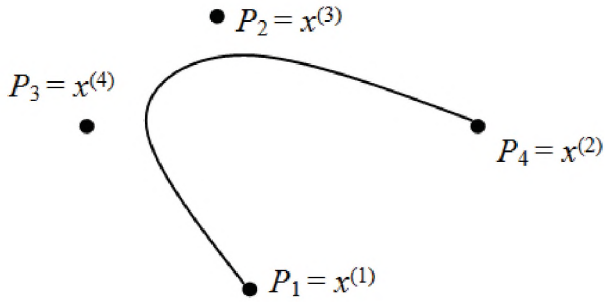


Рис. 1.6. Кривая Безье

Шаг 6.4. В текущей популяции выбрать $P_1 = x^{(1)}, P_4 = x^{(2)}, P_2 = x^{(3)}, \ P_3 = x^{(4)}$ (рис. 1.7) и найти решение задачи параметрической оптимизации:

$$x^{B} = \arg\min_{t\in[0;1]} f\left[\frac{1}{2}[-t(1-t)^2 P_1 + (2-5t^2+3t^3)P_2 + t(1+4t-3t^2)P_3 - t^2(1-t)P_4]\right].$$

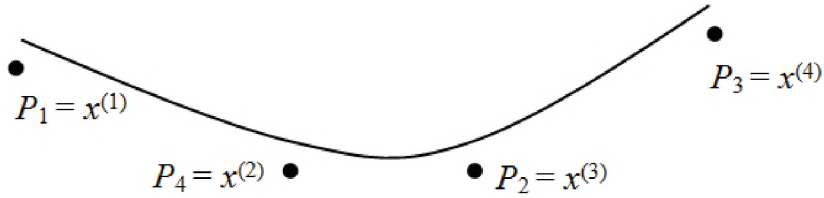


Рис. 1.7. B-сплайн

Шаг 6.5. Предварительное сокращение популяции. Расположить агентов текущей популяции, пополненной новыми агентами $x^{Bezier\,3}$, x^{CR}, $x^{Bezier\,4}$, x^{B} по возрастанию соответствующего им значения целевой функции. Оставить только NP лучших агентов.

Шаг 7. *Сокращение популяции.*

Расположить агентов текущей популяции по возрастанию соответствующего им значения целевой функции: $I_{it} = \{x^{(1)}, ..., x^{(NP)}\}$, где $NP = b_1 + b_2$, $f(x^{(1)}) = f_{\min}$, и удалить b_2 последних агентов, которым соответствуют наихудшие значения целевой функции).

Результатом шага 7 является сокращенная популяция.

Шаг 8. *Проверка условий окончания глобального поиска.*

Если $it < I_{\max}$, то положить $it = it + 1$ и перейти к шагу 9;

Если $it \geq I_{\max}$, то поиск завершить, перейти к шагу 10.

Шаг 9. *Пополнение популяции.*

Шаг 9.1. Генерировать b_2 агентов на множестве D, используя равномерный закон распределения: $x^1, \ldots, x^{b_2}$.

Шаг 9.2. Расположить агентов популяции по возрастанию соответствующего им значения целевой функции: $I_{it} = \{x^{(1)}, ..., x^{(NP)}\}$, где $NP = b_1 + b_2$, $f(x^{(1)}) = f^*_{\min}$. Перейти к шагу 4.

Шаг 10. *Выбор решения из последней популяции.*

Закончить работу алгоритма. В качестве приближенного решения задачи $f(x^*) = \min\limits_{x \in D} f(x)$ выбрать агента с наименьшим значением целевой функции из текущей популяции: $x^* \cong x^{(1)}$.

1.3.3. Программное обеспечение

На основе изложенного алгоритма разработана программа поиска глобального экстремума функций при наличии параллелепипедных ограничений [16]. Для ее создания использовалась среда разработки Microsoft Visual Studio, язык программирования C#.

Возможности программы позволяют изучить алгоритм метода, а также влияние параметров метода на результат его работы. В список выбираемых функций включены стандартные многоэкстремальные тестовые функции двух переменных, для которых известно точное решение (табл. П.1).

На рис. 1.8 представлено главное окно гибридного мультиагентного метода интерполяционного поиска, где пользователь может выбрать вид целевой функции, задать множество допустимых решений и параметры метода, просматривать результаты работы.

Разработанная программа предусматривает возможность анализа работы метода по шагам. На рис. 1.9 представлено окно пошаговой работы метода, где изображена общая схема метода, отображаются результаты работы после выполнения каждого шага (график изменения наилучшего значения целевой функции, а также графическое изображение популяции) и после завершения работы метода.

Программа позволяет реализовать статистический анализ эффективности работы алгоритма по заданному числу стартов (обычно по 100 или 1000 испытаниям). При этом подсчитывается число попаданий приближенного решения в окрестность точки глобального экстремума, среднее значение отклонения полученных решений от точного, наименьшее значение отклонения, а также среднеквадратическое отклонение.

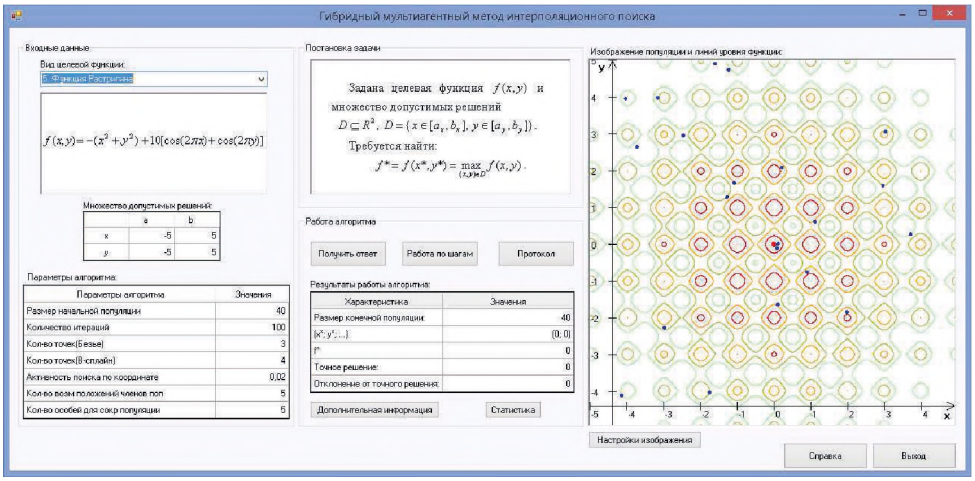


Рис. 1.8. Главное окно программы гибридного мультиагентного метода интерполяционного поиска

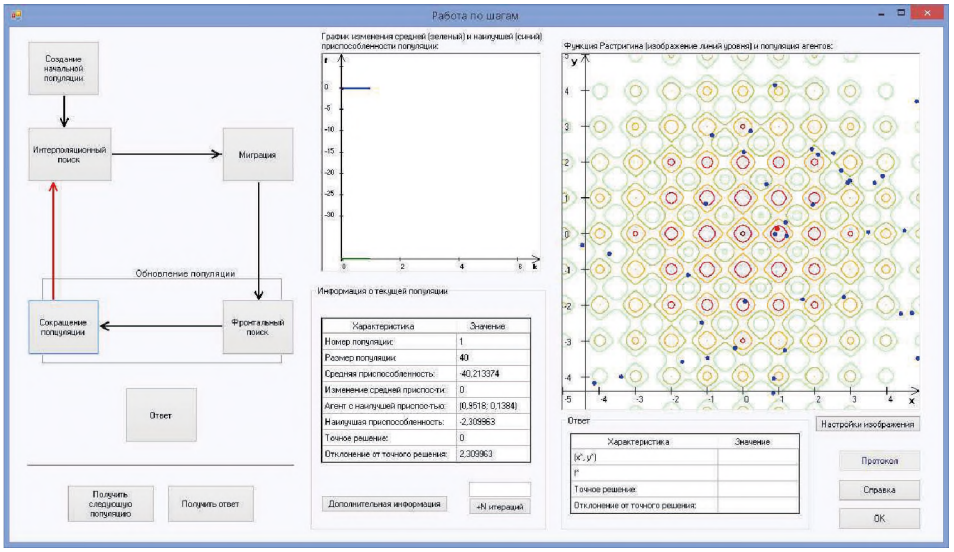


Рис. 1.9. Окно пошаговой работы гибридного мультиагентного метода интерполяционного поиска

1.3.4. Тестовые примеры

Пример 1.1. Найдем глобальный минимум корневой функции (табл. П.1). Зададим множество допустимых решений $x, y \in [-2; 2]$. Выберем следующие параметры алгоритма:

- количество агентов в популяции $NP = 30$;
- количество точек, полученных с использованием кривых Безье $M_1 = 1$;

- количество точек, полученных с использованием B-сплайна $M_2 = 6$;
- параметр, определяющий активность поиска по координате $PRT = 0,05$;
- количество возможных положений агентов популяции $nstep = 3$;
- количество худших агентов для сокращения популяции $b_2 = 10$;
- максимальное число итераций $I_{\max} = 100$.

На рис. 1.10 представлена популяция на начальной ($k = 1$), промежуточных ($k = 35, k = 70$) и конечной ($k = 100$) итерациях. Красным цветом обозначено наилучшее решение в популяции на текущей итерации.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (0,5; 0,866025)$;
- значение целевой функции $f(x^*, y^*) = 1$;
- отклонение от точного решения $\Delta = 0$.

График изменения среднего (зеленый) и наилучшего (синий) значения целевой функции представлен на рис. 1.11.

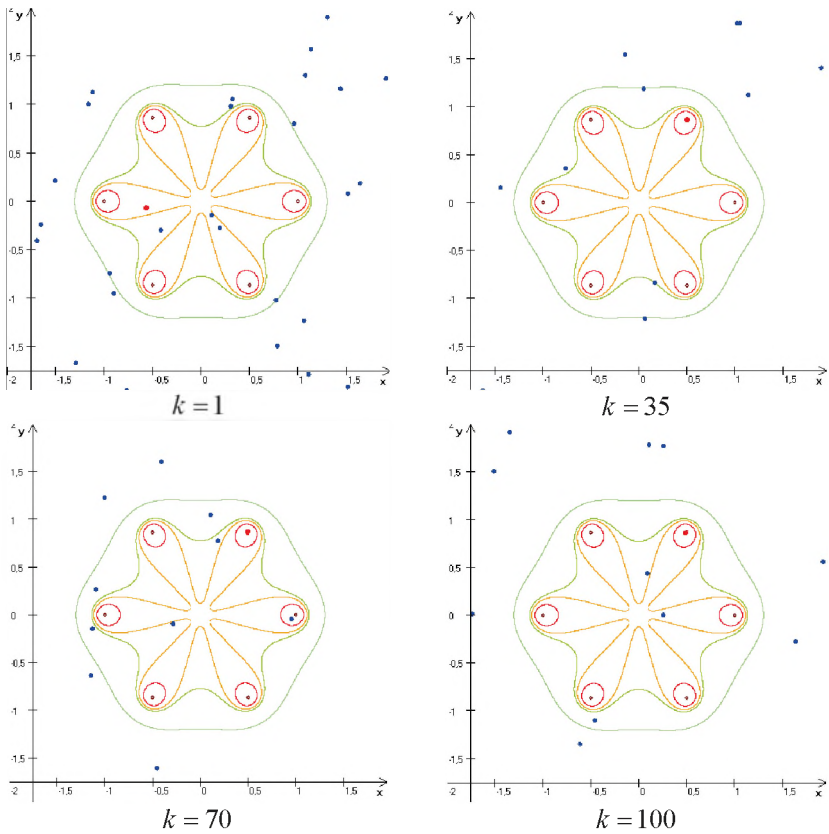


Рис. 1.10. Начальная, промежуточные и конечная итерации

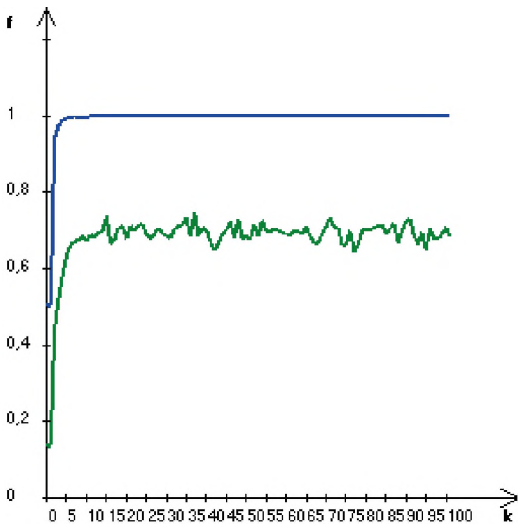


Рис. 1.11. График изменения среднего (зеленый) и наилучшего (синий) значения целевой функции

Пример 1.2. Найдем глобальный минимум функции Швефеля (табл. П.1) на множестве допустимых решений $x, y \in [-500; 500]$. Параметры алгоритма:

- количество агентов в популяции $NP = 50$;
- количество точек, полученных с использованием кривых Безье $M_1 = 6$;
- количество точек, полученных с использованием B-сплайна $M_2 = 5$;
- параметр, определяющий активность поиска по координате $PRT = 0,02$;
- количество возможных положений агентов популяции $nstep = 5$;
- количество худших агентов для сокращения популяции $b_2 = 10$;
- максимальное число итераций $I_{\max} = 1000$.

На рис. 1.12 представлена популяция на начальной ($k = 1$), промежуточных ($k = 350$, $k = 700$) и конечной ($k = 1000$) итерациях. Красным цветом обозначено наилучшее решение в популяции на данной итерации.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (420,9687; 420,9687)$;
- значение целевой функции $f(x^*, y^*) = 837,965775$;
- отклонение от точного решения $\Delta = 0,000025$.

Полученные результаты свидетельствуют о достаточно высокой точности алгоритма.

График изменения среднего (зеленый) и наилучшего (синий) значения целевой функции представлен на рис. 1.13.

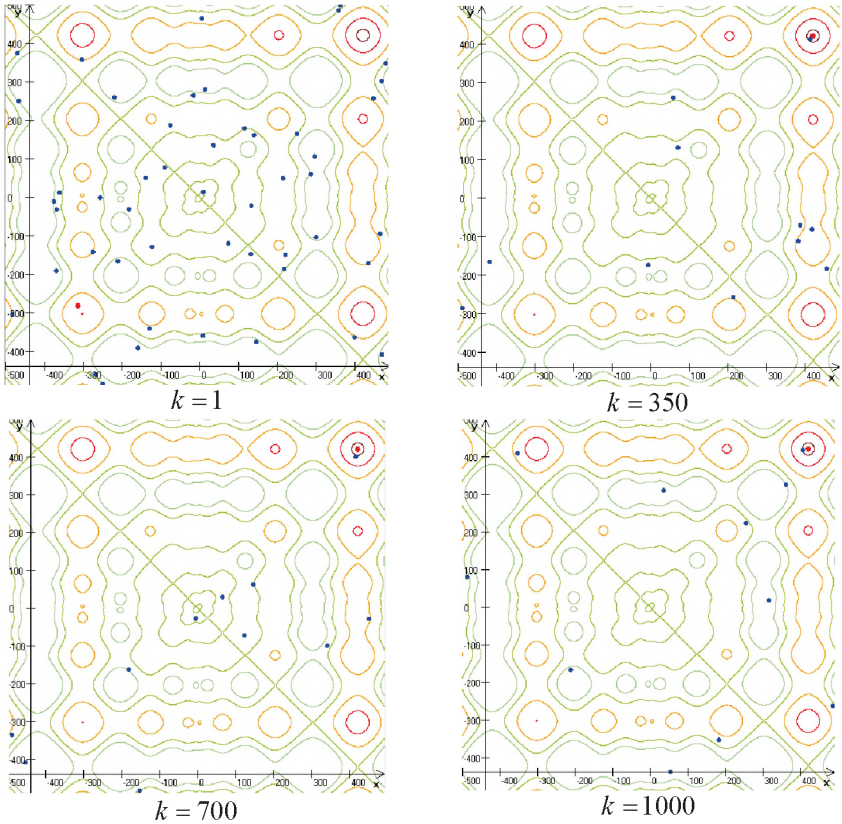


Рис. 1.12. Начальная, промежуточные и конечная популяции

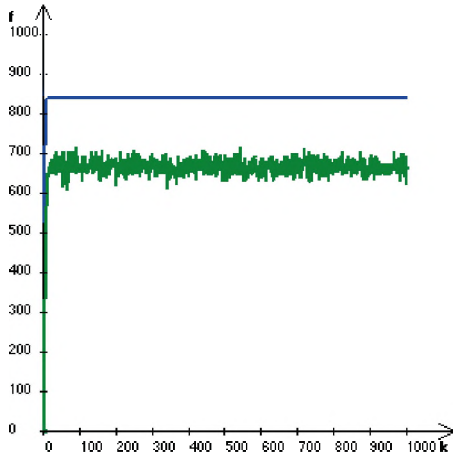


Рис. 1.13. График изменения среднего (зеленый) и наилучшего (синий) значения целевой функции

1.3.5. Анализ эффективности метода

АНАЛИЗ РАБОТЫ МЕТОДА ПРИ РАЗЛИЧНЫХ ЗНАЧЕНИЯХ ПАРАМЕТРОВ

В данном разделе приводится статистический анализ и сравнение работы гибридного мультиагентного метода интерполяционного поиска при различных значениях его параметров. Исследуемый метод применялся к некоторым функциям из набора тестовых функций (табл. П.1). Для каждой функции проводились серии из 100 решений одной и той же задачи с одними и теми же значениями параметров. Для полученной выборки $\{f^1, f^2, ..., f^{100}\}$ вычислялись: среднее значение отклонения полученного решения от точного $\overline{\Delta f} = \frac{1}{100}\sum_{i=1}^{100}\Delta f_i$, где $\Delta f_i = |f(x^*) - f^i|$; наименьшее значение отклонения $\Delta f_{best} = \min_i \Delta f_i$; среднеквадратическое отклонение $\overline{\sigma}_f = \sqrt{\overline{S}_{100}}$, где $\overline{S}_{100} = \frac{1}{100}\sum_{i=1}^{100}(\Delta f^i - \overline{\Delta f})^2$; количество успехов $n_{\text{усп}}$ (успехом считалось попадание лучшей точки в ε-окрестность точного решения, $\varepsilon = \frac{\max_{i=1,..,n}|b_i - a_i|}{1000}$). Результаты, полученные для каждой функции, представлены в табл. 1.1–1.5.

При выборе функций с большой областью допустимых решений, таких как синусоидальная функция Швефеля, близкий к точному решению результат получается только при увеличении размера популяции *NP*. При работе с функцией Розенброка и корневой функцией метод успешно находит окрестность точного решения и практически всегда сходится к точке глобального минимума. При выборе функции «Кожа» с увеличением числа итераций $I_{\max}$ метод сходится к точке глобального минимума с большей точностью. Алгоритм не всегда может найти изолированный глобальный минимум функции Шаффера или один из глобальных минимумов мульти-функции.

Таблица 1.1. Влияние параметров метода. Корневая функция

Параметры метода							$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{\text{усп}}$
NP	$I_{\max}$	M_1	M_2	*PRT*	*nstep*	b_2				
40	100	3	4	0,01	5	5	0,001248	0	0,004036	98
30	100	3	4	0,01	5	5	0,000072	0	0,000607	100
30	100	1	4	0,01	5	5	0,000011	0	0,000052	100
30	100	5	4	0,01	5	5	0,000095	0	0,000507	100
30	100	1	2	0,01	5	5	0,000030	0	0,000141	100
30	100	1	6	0,01	5	5	0,000008	0	0,000022	100
30	100	1	6	0,008	5	5	0,000044	0	0,000201	100
30	100	1	6	0,03	5	5	0,000004	0	0,000016	100
30	100	1	6	0,05	5	5	0,000001	0	0,000002	100
30	100	1	6	0,05	3	5	0,000001	0	0,000004	100
30	100	1	6	0,05	7	5	0,000001	0	0,000002	100
30	100	1	6	0,05	10	5	0,000007	0	0,000033	100
30	100	1	6	0,05	3	3	0,003446	0	0,010640	94
30	100	1	6	0,05	3	7	0	0	0,000002	100
30	100	1	6	0,05	3	10	0	0	0,000001	100

Таблица 1.2. Влияние параметров метода. Функция Шаффера

Параметры метода							$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	I_{max}	M_1	M_2	*PRT*	*nstep*	b_2				
30	100	3	4	0,01	5	5	0,001396	0	0,002135	87
30	200	3	4	0,01	5	5	0,000359	0	0,001224	87
30	200	1	4	0,01	5	5	0,000617	0	0,001600	87
30	200	2	4	0,01	5	5	0,000362	0	0,001251	90
30	200	3	2	0,01	5	5	0,000784	0	0,001789	84
30	200	2	6	0,01	5	5	0,000690	0	0,001609	80
30	200	2	5	0,01	5	5	0,000314	0	0,001138	92
30	200	2	5	0,03	5	5	0,000640	0	0,001621	85
30	200	2	5	0,009	5	5	0,000421	0	0,001327	89
30	200	2	5	0,01	7	5	0,000435	0	0,001373	89
30	200	2	5	0,01	5	7	0,000377	0	0,001279	92
30	200	2	5	0,01	5	8	0,000197	0	0,000957	96

Таблица 1.3. Влияние параметров метода. Мульти-функция

Параметры метода							$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	I_{max}	M_1	M_2	*PRT*	*nstep*	b_2				
30	100	5	6	0,007	3	4	0,7550	25,0398	2,6628	15
20	100	5	6	0,007	3	4	0,0711	4,3239	0,6330	22
20	200	5	6	0,005	3	4	0,1259	2,9638	0,4851	27
20	200	6	6	0,005	3	4	0,0802	1,9996	0,4662	18
20	200	5	8	0,005	3	4	0,1905	2,9692	0,6111	21
20	200	5	6	0,005	3	6	7,2051	308,138	31,917	12
20	200	5	6	0,005	4	2	0,0110	4,3792	0,7488	16
20	300	5	6	0,004	5	4	0,0458	4,30021	0,5213	25
20	200	5	6	0,004	4	2	0,0121	4,2147	0,6147	19
20	200	5	6	0,005	3	5	0,0954	2,5214	0,3475	29

Таблица 1.4. Влияние параметров метода. Функция «Кожа»

Параметры метода							$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	I_{max}	M_1	M_2	*PRT*	*nstep*	b_2				
30	100	3	4	0,02	7	4	0,020409	0	0,042979	68
40	100	3	4	0,02	7	4	0,019490	0	0,036589	57
30	150	3	4	0,02	7	4	0,012372	0	0,019703	70
30	200	3	4	0,02	7	4	0,005915	0	0,015057	85
30	200	1	4	0,02	7	4	0,004456	0	0,012516	84
30	200	5	4	0,02	7	4	0,036796	0	0,079653	50
30	200	3	2	0,02	7	4	0,009176	0	0,018233	79
30	200	3	6	0,02	7	4	0,019416	0	0,041326	63
30	200	3	4	0,04	7	4	0,004212	0	0,013178	90
30	200	3	4	0,06	7	4	0,003251	0	0,011754	92
30	200	3	4	0,06	7	6	0,000921	0	0,006452	98
30	200	3	4	0,06	8	6	0,000160	0	0,001582	99

Таблица 1.5. Влияние параметров метода. Функция Розенброка

Параметры метода							$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	$I_{\max}$	M_1	M_2	PRT	$nstep$	b_2				
30	100	3	4	0,02	5	5	0,000100	0	0,000470	77
30	200	3	4	0,02	5	5	0,000001	0	0,000003	99
30	200	1	4	0,02	5	5	0,000002	0	0,000014	96
30	200	5	4	0,02	5	5	0,000335	0	0,003314	95
30	200	3	2	0,02	5	5	0,000010	0	0,000074	96
30	200	3	6	0,02	5	5	0	0	0,000001	99
30	200	3	6	0,03	5	5	0,000011	0	0,000092	98
30	200	3	6	0,01	5	5	0,000014	0	0,000078	97
30	200	3	6	0,02	6	5	0,000002	0	0,000012	98
30	200	3	6	0,02	5	6	0	0	0	100

РЕКОМЕНДАЦИИ ПО ВЫБОРУ ПАРАМЕТРОВ

Размер популяции агентов NP определяет количество вычислений целевой функции на каждой итерации. Для задачи с большой областью допустимых решений рекомендуется брать большее значение параметра NP. Рекомендуемые значения параметра $NP \in [30;40]$.

Число итераций $I_{\max}$ определяет, как долго будет продолжаться поиск новых решений. Чем больше величина $I_{\max}$, тем более точным будет решение. Для рассмотренного набора стандартных функций рекомендуемые значения в зависимости от сложности функции $I_{\max} \in [100;300]$.

Количество точек M_1**, полученных с использованием кривых Безье** на фазе интерполяционного поиска. При минимизации вдоль кривой образуются M_1 агентов. Рекомендованное значение этого параметра $M_1 \in [2;5]$.

Количество точек M_2**, полученных с использованием B-сплайна** на фазе интерполяционного поиска. При минимизации вдоль кривой образуются M_2 агентов. Кривая применяется для исследования новых областей. Рекомендованное значение этого параметра $M_2 \in [4;6]$.

Параметр PRT, определяющий активность поиска по координате во время миграции популяции. Параметр определяет направление поиска. Рекомендованное значение этого параметра $PRT \in [0,005;0,06]$.

Количество возможных положений агентов популяции $nstep$ во время миграции популяции. Параметр определяет, сколько шагов сделает агент по направлению к лидеру. Рекомендованное значение этого параметра $nstep \in [3;7]$.

Количество худших агентов для сокращения популяции b_2. Из популяции удаляются b_2 худших агентов и добавляется b_2 новых агентов для поддержания одного и того же количества NP агентов. Рекомендованное значение этого параметра $b_2 \in [4;8]$.

1.4. МУЛЬТИАГЕНТНЫЙ МЕТОД, ИСПОЛЬЗУЮЩИЙ ЛИНЕЙНЫЕ РЕГУЛЯТОРЫ ДЛЯ УПРАВЛЕНИЯ ДВИЖЕНИЕМ АГЕНТОВ

1.4.1. Стратегия поиска решения

Рассматривается задача (1.2) поиска условного глобального максимума целевой функции $f(x)$ на множестве допустимых решений D, т.е. такой точки $x^* \in D$, что

$$f(x^*) = \min_{x \in D} f(x),$$

где $x = (x_1, x_2, ..., x_n)^T$, $D = \{x \mid x_i \in [a_i, b_i], i = 1, 2, ..., n\}$.

Метод основан на процессах, происходящих в среде, содержащей множество *агентов*. Агенты имеют возможность обмениваться информацией для того, чтобы найти решение задачи. Стратегия алгоритма основана на использовании линейных регуляторов управления движением агентов [1, 35, 40, 110].

На начальном этапе требуется сгенерировать популяцию из NP агентов на множестве допустимых решений D, используя равномерный закон распределения.

Поиск экстремума реализуется за заданное число проходов $P_{\max}$. В рамках очередного прохода все агенты двигаются под действием соответствующего управления в течение определенного числа итераций.

Уравнение движения агентов

$$\frac{dx}{dt} = v,\ x(t_0) = x_0,$$

$$\frac{dv}{dt} = u,\ v(t_0) = v_0, \tag{1.3}$$

где x – n-мерный вектор положения агента, v – n-мерный вектор скорости агента, t – время, t_0 – начальный момент времени в рамках очередного прохода, x_0 – начальное положение, v_0 – начальная скорость, u – n-мерный вектор управления агентом. При $t_0 = 0$ можно положить $v_0 = o$ (o – нулевой n-мерный вектор).

Обозначим $X = \begin{pmatrix} x \\ v \end{pmatrix}$ – вектор состояния агента и перепишем уравнение (1.3) в форме

$$\frac{d\begin{pmatrix} x \\ v \end{pmatrix}}{dt} = \begin{pmatrix} O_n & E_n \\ O_n & O_n \end{pmatrix}\begin{pmatrix} x \\ v \end{pmatrix} + \begin{pmatrix} O_n \\ E_n \end{pmatrix}u,$$

или

$$\frac{dX}{dt} = AX(t) + Bu(t),\ X(t_0) = X_0 = \begin{pmatrix} x_0 \\ v_0 \end{pmatrix}, \tag{1.4}$$

где O_n, E_n – нулевая и единичная матрицы порядка n, $A = \begin{pmatrix} O_n & E_n \\ O_n & O_n \end{pmatrix}$, $B = \begin{pmatrix} O_n \\ E_n \end{pmatrix}$ – матрицы размеров $(2n \times 2n)$, $(2n \times n)$ соответственно; на управление ограничений не наложено, т.е. $u \in R^n$.

В начальной популяции ($k=0$), а также в конце каждого k-го прохода, определить положение лидера среди агентов популяции и соответствующее ему максимальное значение целевой функции: $x^{best,k}, f(x^{best,k})$. В течение следующего прохода он не изменяет своего положения:

$$\frac{dx^{best}}{dt}=o,\ x^{best}(t_k)=x^{best,k},$$

$$t\in[t_k,t_{k+1}),k=0,...,P_{\max}-1, \qquad (1.5)$$

$$\frac{dv^{best}}{dt}=o,\ v^{best}(t_k)=o,$$

где управление лидером $u^{best}=o$, или $\frac{dX^{best}}{dt}=o$, вектор состояния лидера $X^{best}=(x^{best},v^{best})^T$.

Остальные $(NP-1)$ агентов делятся на четыре равные группы:

- агенты, использующие минимизацию критерия качества управления агентами на конечном промежутке времени;
- агенты, использующие минимизацию критерия качества управления агентами на бесконечном промежутке времени;
- агенты, использующие минимизацию полуопределенного критерия качества (критерия обобщенной работы) управления агентами на конечном промежутке времени;
- агенты, использующие минимизацию приращения критерия качества управления агентами (локально-оптимальный подход).

Для всех агентов каждой группы позиции и векторы скорости разные, но находится и применяется одно и то же управление с полной обратной связью по вектору состояния, определяемое соотношением для линейного регулятора, матрица коэффициентов усиления которого находится из условия минимизации квадратичного критерия качества управления, характеризующего характер приближения агента к агенту-лидеру на текущей итерации, а также интенсивность применяемого управления.

Введем отклонение от лидера $\Delta X=X-X^{best}$, изменение которого описывается уравнением (вычитая из (1.4) уравнение (1.5)):

$$\frac{d\Delta X}{dt}=A\Delta X(t)+Bu(t),\quad \Delta X(t_k)=\begin{pmatrix}x^k-x^{best,k}\\ v^k\end{pmatrix},\ t\in[t_k,t_{k+1}],k=0,...,P_{\max}-1, \qquad (1.6)$$

где x^k,v^k – положение и скорость агента в конце предыдущего прохода.

Движение первой группы агентов. Для всех агентов группы применяется оптимальное управление с конечным горизонтом, т.е. линейно-квадратичный регулятор (ЛК-регулятор). Критерий качества управления траекториями агентов первой группы:

$$I=\frac{1}{2}\int_{t_k}^{t_{k+1}}\left[\Delta X^T(t)S(t)\Delta X(t)+u^T(t)Q(t)u(t)\right]dt+\frac{1}{2}\Delta X^T(t_{k+1})\Lambda\Delta X(t_{k+1}),$$

где $S(t),\Lambda$ – неотрицательно определенные симметрические матрицы размеров $(2n\times 2n)$, а $Q(t)$ – положительно определенная симметрическая матрица $(n\times n)$.

Оптимальное управление $u^*(t,\Delta X)$ с полной обратной связью для любых начальных состояний имеет вид

$$u^*(t,\Delta X)=-Q^{-1}(t)B^T(t)P(t)\Delta X=-F(t)\Delta X,$$

где матрица коэффициентов усиления линейного регулятора $F(t) = Q^{-1}(t)B^T(t)P(t)$, а $P(t)$ – симметрическая матрица размеров $(n \times n)$, удовлетворяющая дифференциальному уравнению Риккати:

$$\dot{P}(t) = -A^T(t)P(t) - P(t)A(t) + P(t)B(t)Q^{-1}(t)B^T(t)P(t) - S(t),\ P(t_{k+1}) = \Lambda.$$

Здесь $t \in [t_k, t_{k+1}]$, $t_0 = 0$, значение $t_{k+1} = t_k + NMAX \cdot h$, где $NMAX$ – заданное число итераций, h – шаг интегрирования дифференциального уравнения (1.5). Для упрощения решения можно везде положить $A(t) = A, B(t) = B, Q(t) = Q, S(t) = S$, так как модели (1.5), (1.6) линейные стационарные.

Движение второй группы агентов. Для всех агентов группы применяется оптимальное управление с бесконечным горизонтом, определяемое процедурой аналитического конструирования оптимальных регуляторов (АКОР). Критерий качества управления траекториями агентов второй группы:

$$I = \frac{1}{2}\int_{t_k}^{+\infty} \left[\Delta X^T(t) S \Delta X(t) + u^T(t) Q u(t) \right] dt,$$

где S – неотрицательно определенная симметрическая матрица размеров $(2n \times 2n)$, а Q – положительно определенная симметрическая матрица размеров $(n \times n)$.

Оптимальное управление $u^*(\Delta X)$ с полной обратной связью для любых начальных состояний имеет вид:

$$u^*(\Delta X) = -Q^{-1}B^T P \Delta X = -F \Delta X,$$

где матрица коэффициентов усиления линейного регулятора $F = Q^{-1}B^T P$, а P – положительно определенная симметрическая матрица, удовлетворяющая алгебраическому уравнению Риккати:

$$-A^T P - P A + P B Q^{-1} B^T P - S = 0.$$

Решение этого уравнения, удовлетворяющее критерию Сильвестра, единственное.

Движение третьей группы агентов (для всех агентов группы применяется оптимальное управление по функционалу обобщенной работы (ФОР)). Критерий качества управления траекториями агентов третьей группы:

$$I^{\text{o.p.}} = \frac{1}{2}\int_{t_k}^{t_{k+1}} \left[\Delta X^T(t) S(t) \Delta X(t) + u^T(t) Q(t) u(t) + \Delta X^T(t) P(t) B(t) Q^{-1}(t) B^T(t) P(t) \Delta X(t) \right] dt +$$

$$+ \frac{1}{2} \Delta X^T(t_{k+1}) \Lambda \Delta X(t_{k+1}),$$

где S, Λ – неотрицательно определенные симметрические матрицы размеров $(2n \times 2n)$, а $Q(t)$ – положительно определенная симметрическая матрица размеров $(n \times n)$.

Оптимальное управление $u^{\text{o.p.}}(t, \Delta X)$ с полной обратной связью для любых начальных состояний имеет вид:

$$u^{\text{o.p.}}(t, \Delta X) = -Q^{-1}(t)B^T(t)P(t)\Delta X = -F(t)\Delta X,$$

где матрица коэффициентов усиления линейного регулятора $F(t) = Q^{-1}(t)B^T(t)P(t)$, а $P(t)$ – симметрическая матрица, удовлетворяющая линейному дифференциальному уравнению:

$$\dot{P}(t) = -A^T(t)P(t) - P(t)A(t) - S(t), \quad P(t_{k+1}) = \Lambda.$$

Движение четвертой группы агентов (для всех агентов группы применяется локально-оптимальное управление (ЛО-регулятор)). Критерий качества управления траекториями агентов четвертой группы:

$$I^{loc} = \frac{1}{2}\int_{t_k}^{t} \left[\Delta X^T(\tau) S(\tau) \Delta X(\tau) + u^T(\tau) Q(\tau) u(\tau) \right] d\tau + \frac{1}{2} \Delta X^T(t) \Lambda(t) \Delta X(t),$$

где $S(t), \Lambda(t)$ – неотрицательно определенные симметрические матрицы размеров $(2n \times 2n)$, а $Q(t)$ – положительно определенная симметрическая матрица $(n \times n)$.

Локально-оптимальное управление $u^{loc}(t, \Delta X)$ с полной обратной связью для любых начальных состояний имеет вид:

$$u^{loc}(t, \Delta X) = -\ Q^{-1}(t) B^T(t) \Lambda(t) \Delta X = -F(t) \Delta X,$$

где матрица коэффициентов усиления линейного регулятора $F(t) = Q^{-1}(t) B^T(t) \Lambda(t)$.

Заметим, что матрицы, входящие в функционалы качества управления для всех четырех групп агентов могут быть заданы независимо, несмотря на одинаковые обозначения.

После выполнения *NMAX* итераций каждый агент предъявляет наилучший из полученных во время движения результатов. Тем самым завершается очередной проход. Затем среди всех агентов популяции снова выбирается лидер и производится новое деление на группы. Процесс повторения проходов завершается при достижении максимального числа проходов.

Общая схема работы мультиагентного метода, использующего линейные регуляторы для управления движением агентов, представлена на рис. 1.14.

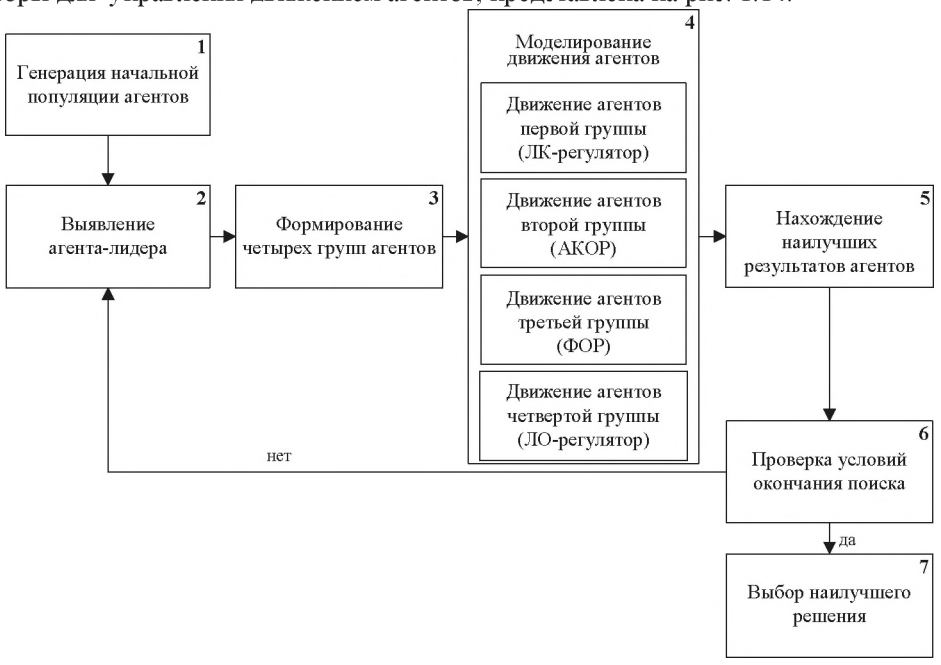


Рис. 1.14. Общая схема работы мультиагентного метода, использующего линейные регуляторы для управления движением агентов

1.4.2. Алгоритм решения задачи

Шаг 1. Задать параметры метода:

- размер популяции NP;
- максимальное число проходов $P_{\max}$;
- число итераций за время прохода $NMAX$;
- шаг интегрирования дифференциальных уравнений h;
- матрицы S, Q, Λ, определяющие критерии качества движения агентов каждой группы.

Положить $k = 0$ (счетчик числа итераций), $v^0 = o$, $t_0 = 0$.

Шаг 2. *Формирование начальной популяции агентов.* Генерировать начальную популяцию на множестве D, используя равномерный закон распределения: $x^1, \ldots, x^{NP}$. Вычислить значения целевой функции $f(x^1), \ldots, f(x^{NP})$.

Шаг 3. *Упорядочение агентов в популяции.* Упорядочить популяцию, состоящую из NP особей, по величине целевой функции.

Шаг 4. *Выбор агента-лидера.* Выбрать агента-лидера и соответствующее наибольшее значение целевой функции: $x^{best,k}$, $f^{best,k}$.

Шаг 5. *Формирование групп агентов.* Поделить всех остальных $(NP-1)$ агентов произвольно на четыре группы.

Для каждого агента составить дифференциальное уравнение

$$\frac{d\Delta X}{dt} = A\,\Delta X(t) + B u(t),\ \Delta X(t_k) = \begin{pmatrix} x^k - x^{best,k} \\ v^k \end{pmatrix},$$

где $A = \begin{pmatrix} O_n & E_n \\ O_n & O_n \end{pmatrix}$, $B = \begin{pmatrix} O_n \\ E_n \end{pmatrix}$.

Шаг 6. *Движение агентов первой группы* (применяется оптимальное управление с конечным горизонтом).

Шаг 6.1. Найти решение дифференциального уравнения Риккати $P(t)$ $\forall t \in [t_k, t_{k+1}]$:

$$\dot{P}(t) = -A^T(t)P(t) - P(t)A(t) + P(t)B(t)Q^{-1}(t)B^T(t)P(t) - S(t),\quad P(t_{k+1}) = \Lambda,$$

где $t_{k+1} = t_k + NMAX \cdot h$.

Шаг 6.2. Найти оптимальное управление $u^*(t, \Delta X)$ с полной обратной связью:

$$u^*(t, \Delta X) = -\,Q^{-1}(t)B^T(t)P(t)\Delta X = -F(t)\Delta X, \text{ где } F(t) = Q^{-1}(t)B^T(t)P(t).$$

Шаг 6.3. Для каждого агента первой группы выполнить следующие действия.

Шаг 6.3.1. Найти решение дифференциального уравнения:

$$\frac{d\Delta X}{dt} = A\,\Delta X(t) + B u^*(t, \Delta X(t)),\ \Delta X(t_k) = \begin{pmatrix} x^k - x^{best,k} \\ v^k \end{pmatrix}$$

на промежутке $[t_k, t_{k+1}]$: $\Delta X(t_i), t_i = t_k + ih, i = 0,1,\ldots, NMAX - 1$.

Шаг 6.3.2. Найти векторы состояния агента в течение прохода:

$$X^{I,New}(t_i) = X^{best} + \Delta X(t_i),\ i = 0,1,...,NMAX - 1.$$

Шаг 6.3.3. Среди всех положений агента в течение прохода выбрать наилучшее за весь период движения, которому соответствует наибольшее значение целевой функции.

Шаг 7. *Движение агентов второй группы* (применяется оптимальное управление с бесконечным горизонтом).

Шаг 7.1. Найти решение P алгебраического уравнения Риккати:

$$-A^T P - P A + P B Q^{-1} B^T P - S = 0.$$

Шаг 7.2. Найти оптимальное управление $u^*(\Delta X)$ с полной обратной связью:

$$u^*(\Delta X) = -Q^{-1} B^T P \Delta X = -F \Delta X,$$

где $F = Q^{-1} B^T P.$

Шаг 7.3. Для каждого агента второй группы выполнить следующие действия.

Шаг 7.3.1. Найти решение дифференциального уравнения:

$$\frac{d\Delta X}{dt} = A\Delta X(t) + B u^*(\Delta X(t)),\ \Delta X(t_k) = \begin{pmatrix} x^k - x^{best,k} \\ v^k \end{pmatrix}.$$

Шаг 7.3.2. Найти векторы состояния агента в течение прохода:

$$X^{II,New}(t_i) = X^{best} + \Delta X(t_i),\ i = 0,1,...,NMAX - 1.$$

Шаг 7.3.3. Среди всех положений агента в течение прохода выбрать наилучшее за весь период движения, которому соответствует наибольшее значение целевой функции.

Шаг 8. *Движение агентов третьей группы* (применяется оптимальное управление по критерию обобщенной работы).

Шаг 8.1. Найти решение линейного дифференциального уравнения $P(t)$ $\forall t \in [t_k, t_{k+1}]$:

$$\dot{P}(t) = -A^T(t) P(t) - P(t) A(t) - S(t),\quad P(t_{k+1}) = \Lambda.$$

Шаг 8.2. Найти оптимальное управление $u^{\text{o.p.}}(t, \Delta X)$ с полной обратной связью:

$u^{\text{o.p.}}(t, \Delta X) = -Q^{-1}(t) B^T(t) P(t) \Delta X = -F(t) \Delta X$, где $F(t) = Q^{-1}(t) B^T(t) P(t).$

Шаг 8.3. Для каждого агента третьей группы выполнить следующие действия.

Шаг 8.3.1. Найти решение дифференциального уравнения:

$$\frac{d\Delta X}{dt} = A\Delta X(t) + B u^{\text{o.p.}}(t, \Delta X(t)),\ \Delta X(t_k) = \begin{pmatrix} x^k - x^{best,k} \\ v^k \end{pmatrix}.$$

Шаг 8.3.2. Найти векторы состояния агента в течение прохода:

$$X^{III,New}(t_i) = X^{best} + \Delta X(t_i),\ i = 0,1,...,NMAX - 1.$$

Шаг 8.3.3. Среди всех положений агента в течение прохода выбрать наилучшее за весь период движения, которому соответствует наибольшее значение целевой функции.

Шаг 9. *Движение агентов четвертой группы* (применяется локально-оптимальное управление).

Шаг 9.1. Найти локально-оптимальное управление $u^{loc}(t,\Delta X)$ с полной обратной связью:

$$u^{loc}(t,\Delta X) = -\,Q^{-1}(t)\,B^T(t)\,\Lambda(t)\,\Delta X = -F(t)\,\Delta X\,,\ \text{где}\ F(t) = Q^{-1}(t)B^T(t)\Lambda(t)\,.$$

Шаг 9.2. Для каждого агента четвертой группы выполнить следующие действия.

Шаг 9.2.1. Найти решение дифференциального уравнения:

$$\frac{d\Delta X}{dt} = A\,\Delta X(t) + B u^{loc}(t,\Delta X(t)),\ \Delta X(t_k) = \begin{pmatrix} x^k - x^{best,k} \\ v^k \end{pmatrix}.$$

Шаг 9.2.2. Найти векторы состояния агента в течение прохода:

$$X^{k,New}(t_i) = X^{best} + \Delta X(t_i)\,,\ i = 0,1,\ldots,NMAX-1.$$

Шаг 9.2.3. Среди всех положений агента в течение прохода выбрать наилучшее за весь период движения, которому соответствует наибольшее значение целевой функции.

Результатом шагов 6–9 являются положения NP агентов популяции, из которых $(NP-1)$ агентов с новым положением в результате движения под действием управления в рамках фиксированной группы и агент-лидер x^{best}, который не изменял своего положения за время последнего прохода.

Шаг 10. *Проверка условий окончания глобального поиска.*

Если $k < P_{\max} - 1$, то поиск продолжить, перейти к шагу 3 (упорядочить популяцию агентов, выявить текущего лидера, перетасовать группы) и положить $k = k+1$.

Если $k \geq P_{\max} - 1$, то поиск завершить, перейти к шагу 11.

Шаг 11. *Выбор решения из последней популяции.*

Закончить работу алгоритма. В качестве приближенного решения задачи $f(x^*) = \max\limits_{x\in D} f(x)$ выбрать агента с наибольшим значением целевой функции из последней популяции: $x^* \cong \tilde{x}^* = \arg\max\limits_{j=1,\ldots,NP} f(x^j)$.

Замечание. Можно положить $S = \begin{pmatrix} k_s E_n & O_n \\ O_n & O_n \end{pmatrix}$, $Q = E_n$, $\Lambda = \begin{pmatrix} k_l E_n & O_n \\ O_n & O_n \end{pmatrix}$, $k_s > 0,\ k_l > 0$.

МОДИФИКАЦИЯ МЕТОДА

После шага 9 в исходном алгоритме выполнить следующие шаги.

Шаг 10. *Сокращение популяции.*

Расположить агентов текущей популяции по убыванию соответствующего им значения целевой функции: $I_t = \{x^1,\ldots,x^{NP}\}$, $f(x^1) = f_{\max}$, и удалить d последних агентов.

Результатом шага 10 является сокращенная популяция.

Шаг 11. *Проверка условий окончания глобального поиска.*

Если $k < P_{\max} - 1$, то поиск продолжить, перейти к шагу 3 (упорядочить популяцию агентов, выявить текущего лидера, перетасовать группы) и положить $k = k+1$.

Если $k \geq P_{\max} - 1$, то поиск завершить, перейти к шагу 13.

Шаг 12. *Пополнение популяции.*

Шаг 12.1. Генерировать d агентов на множестве D, используя равномерный закон распределения, и заменить ими удаленных на шаге 10 агентов.

Шаг 12.2. Расположить агентов популяции по убыванию соответствующего им значения целевой функции. Перейти к шагу 4 исходного алгоритма.

Шаг 13. *Выбор решения из последней популяции.*

Закончить работу алгоритма. В качестве приближенного решения задачи $f(x^*) = \max\limits_{x \in D} f(x)$ выбрать агента с наибольшим значением целевой функции из текущей популяции: $x^* \cong x^1$.

1.4.3. Программное обеспечение

На основе изложенного алгоритма разработана программа поиска глобального экстремума функций [16]. Для ее создания использовалась среда разработки Microsoft Visual Studio, язык программирования C#.

Возможности программы позволяют изучить алгоритм метода, а также влияние параметров метода на результат его работы. В список выбираемых функций включены стандартные многоэкстремальные тестовые функции двух переменных, для которых известно точное решение (табл. П.1).

На рис. 1.15 представлено главное окно мультиагентного метода, использующего линейные регуляторы для управления движением агентов, где пользователь может выбрать вид целевой функции, задать множество допустимых решений и параметры метода, просматривать результаты работы.

Разработанная программа предусматривает возможность анализа работы метода по шагам. На рис. 1.16 представлено окно пошаговой работы метода, где изображена общая схема метода, отображаются результаты работы после выполнения каждого шага (график изменения наилучшего значения целевой функции, а также графическое изображение популяции) и после завершения работы метода.

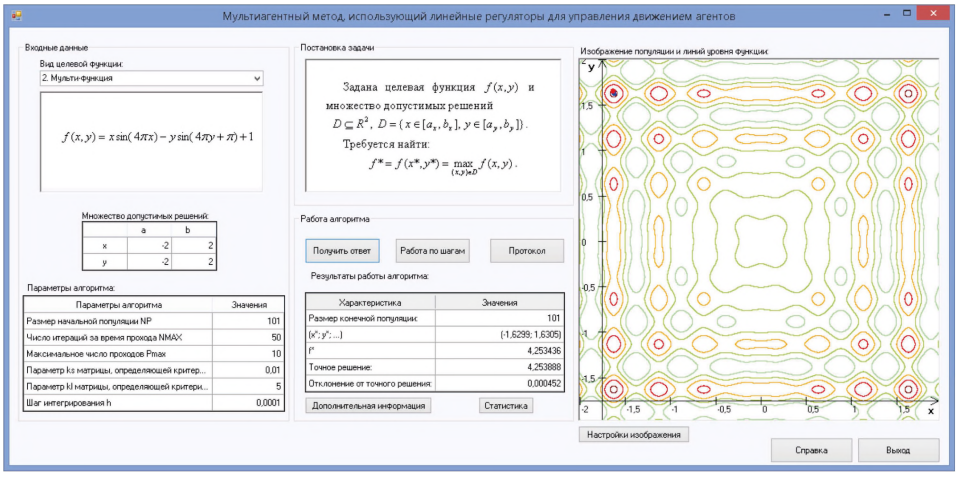


Рис. 1.15. Главное окно программы мультиагентного метода, использующего линейные регуляторы для управления движением агентов

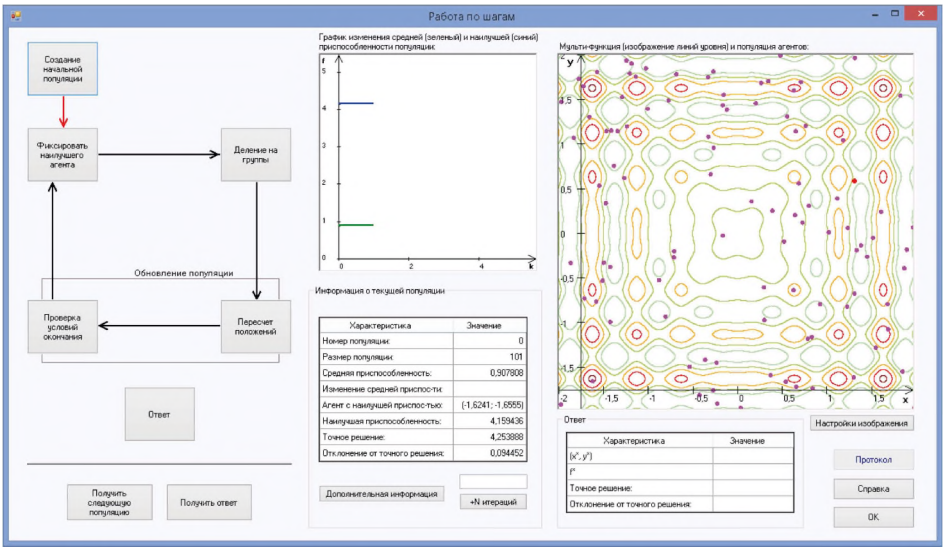


Рис. 1.16. Окно пошаговой работы мультиагентного метода, использующего линейные регуляторы для управления движением агентов

1.4.4. Тестовые примеры

Пример 1.3. Найдем глобальный максимум корневой функции (табл. П.1). Зададим множество допустимых решений $x, y \in [-2; 2]$.

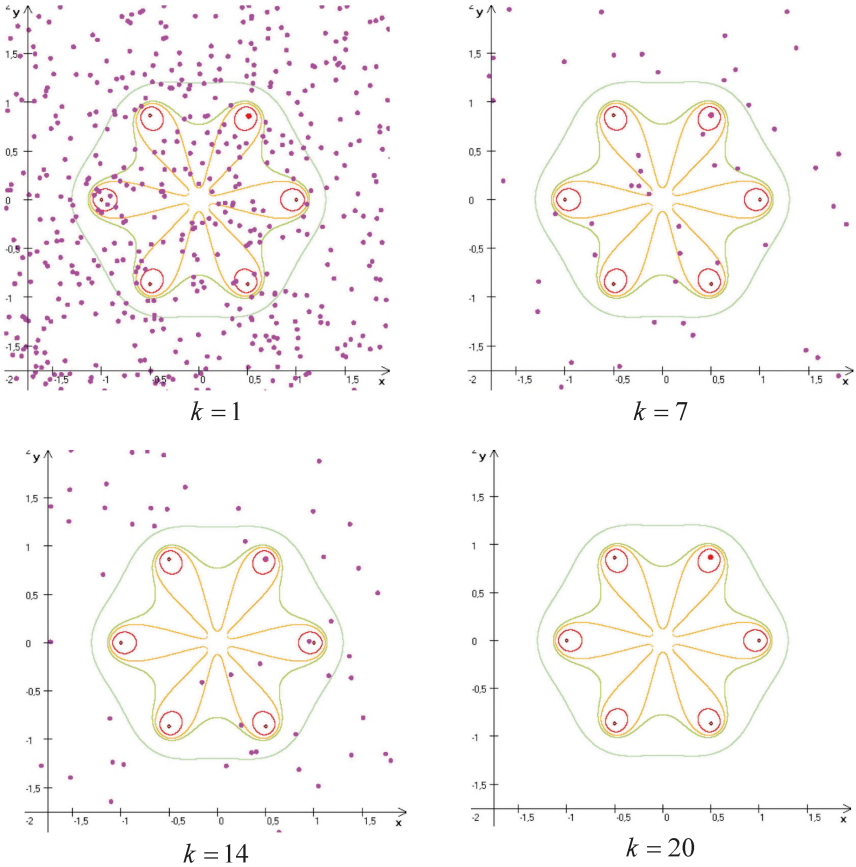
Выберем следующие параметры алгоритма:

- размер популяции $NP = 501$;
- максимальное число проходов $P_{\max} = 20$;
- число итераций за время прохода $NMAX = 40$;
- шаг интегрирования дифференциальных уравнений $h = 0,0001$;
- параметр матрицы S, определяющей критерий качества, $k_s = 1$;
- параметр матрицы Λ, определяющей критерий качества, $k_l = 3$.

На рис. 1.17 представлена популяция на начальной ($k = 1$), промежуточных ($k = 7, k = 14$) и конечной ($k = 20$) итерациях. Красным цветом обозначено наилучшее решение в популяции на текущей итерации. График изменения наилучшего значения целевой функции представлен на рис. 1.18. Результаты свидетельствуют о том, что решение достаточно хорошего качества может быть получено за небольшое число итераций.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (0,499945; 0,866149)$;
- значение целевой функции $f(x^*, y^*) = 0,99956$;
- отклонение от точного решения $\Delta = 0,00044$.



$k = 1$ $k = 7$

$k = 14$ $k = 20$

Рис. 1.17. Начальная, промежуточные и конечная итерации

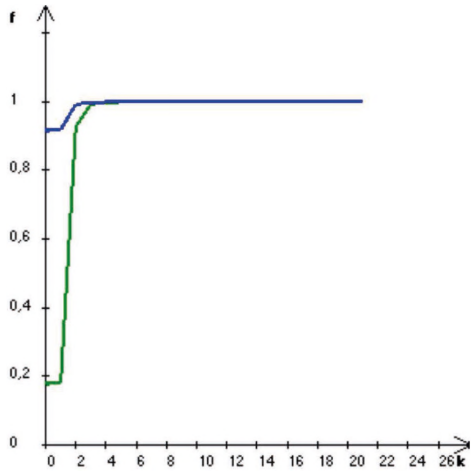


Рис. 1.18. График изменения среднего (зеленый) и наилучшего (синий) значения целевой функции

Пример 1.4. Найдем глобальный максимум функции Экли (табл. П.1) на множестве допустимых решений $x, y \in [-10;10]$. Параметры алгоритма:

- размер популяции $NP = 501$;
- максимальное число проходов $P_{\max} = 30$;
- число итераций за время прохода $NMAX = 100$;
- шаг интегрирования дифференциальных уравнений $h = 0,001$;
- параметр матрицы S, определяющей критерий качества, $k_s = 4$;
- параметр матрицы Λ, определяющей критерий качества, $k_l = 5$.

На рис. 1.19 представлена популяция на начальной ($k = 1$), промежуточных ($k = 10, k = 20$) и конечной ($k = 30$) итерациях. График изменения наилучшего значения целевой функции (ему соответствует красная точка) представлен на рис. 1.20.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (6 \cdot 10^{-6}; -1 \cdot 10^{-6})$;
- значение целевой функции $f(x^*, y^*) = 19,999983$;
- отклонение от точного решения $\Delta = 1,7 \cdot 10^{-5}$.

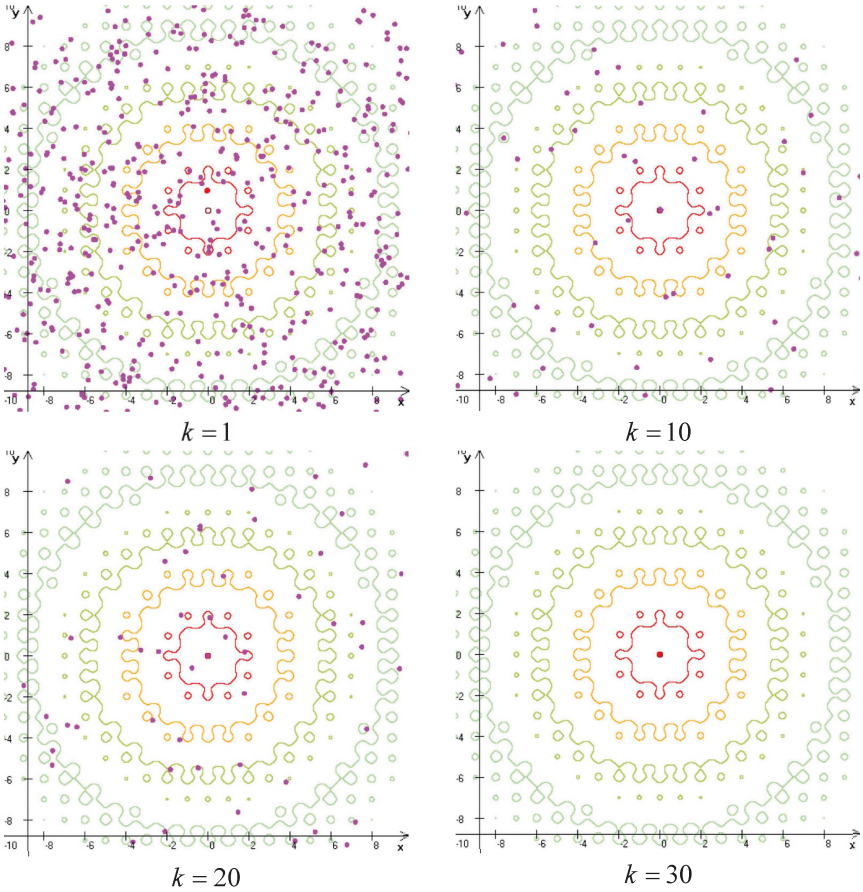


Рис. 1.19. Начальная, промежуточные и конечная популяции

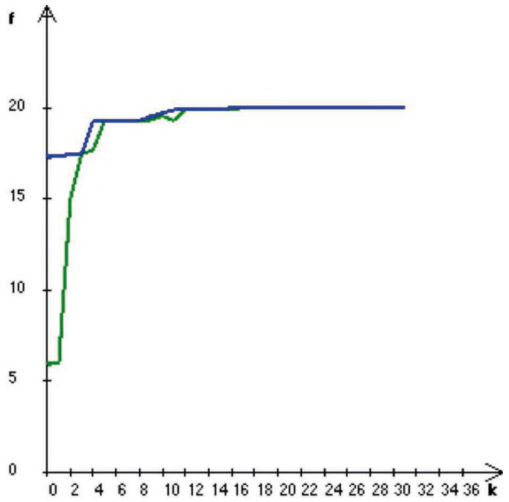


Рис. 1.20. График изменения среднего (зеленый) и наилучшего (синий) значения целевой функции

1.4.5. Анализ эффективности метода

АНАЛИЗ РАБОТЫ МЕТОДА ПРИ РАЗЛИЧНЫХ ЗНАЧЕНИЯХ ПАРАМЕТРОВ

В данном разделе приводится статистический анализ и сравнение работы мультиагентного метода, использующего линейные регуляторы для управления движением агентов, при различных значениях его параметров. Исследуемый метод применялся к некоторым функциям из набора тестовых функций (табл. П.1). При этом искался минимум целевой функции с учетом описанного в разд. 1.1 перехода к задаче поиска максимума.

Для каждой функции проводились серии из 100 решений одной и той же задачи с одними и теми же значениями параметров. Для полученной выборки $\{f^1, f^2, \ldots, f^{100}\}$ вычислялись среднее значение отклонения полученного решения от точного $\overline{\Delta f} = \frac{1}{100}\sum_{i=1}^{100} \Delta f_i$, где $\Delta f_i = \left| f(x^*) - f^i \right|$; наименьшее значение отклонения $\Delta f_{best} = \min_i \Delta f_i$; среднеквадратическое отклонение $\overline{\sigma}_f = \sqrt{S_{100}}$, где $\overline{S}_{100} = \frac{1}{100}\sum_{i=1}^{100} (\Delta f^i - \overline{\Delta f})^2$; количество успехов $n_{\text{усп}}$ (успехом считалось попадание лучшей точки в ε-окрестность точного решения, $\varepsilon = \frac{\max\limits_{i=1,\ldots,n} \left| b_i - a_i \right|}{1000}$). Результаты, полученные для каждой функции, представлены в табл. 1.6–1.9.

Анализ работы мультиагентного метода, использующего линейные регуляторы для управления движением агентов, показал, что с большинством тестовых функций метод справляется успешно. В серии из 100 запусков удается найти глобальный минимум функций с большой точностью или точное решение.

Таблица 1.6. Влияние параметров метода. Корневая функция

Параметры метода						$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	*NMAX*	P_{max}	k_s	k_l	*h*				
501	20	10	0,1	1	0,0001	0,004736	0,000004	0,005977	97
501	50	10	0,1	1	0,0001	0,005330	0,000002	0,006751	95
501	20	20	0,1	1	0,0001	0,005181	0,000004	0,005917	98
501	40	20	0,1	1	0,0001	0,004378	0,000039	0,005380	98
501	40	20	0,01	1	0,0001	0,007502	0,000008	0,008980	91
501	40	20	1	1	0,0001	0,002395	0,000007	0,003179	100
501	40	20	1	0,5	0,0001	0,002608	0,000011	0,003197	100
501	40	20	1	0,1	0,0001	0,002897	0,000012	0,004087	99
501	40	20	1	3	0,0001	0,002810	0,000004	0,003012	100
501	40	20	1	1	0,001	0,003389	0,000005	0,003778	100

Таблица 1.7. Влияние параметров метода. Функция Розенброка

Параметры метода						$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	*NMAX*	P_{max}	k_s	k_l	*h*				
5001	50	10	0,1	1	0,0001	0,000385	0,000025	0,001561	79
2001	50	10	0,1	1	0,0001	0,003127	0,000025	0,009733	55
1601	50	10	0,1	0,5	0,0001	0,005495	0,000025	0,021443	99
1601	50	10	0,1	1,5	0,0001	0,005399	0,000025	0,021717	99
1601	50	15	0,1	1,5	0,0001	1,188174	0,000025	11,784093	99
1601	50	10	0,1	0,5	0,0001	1,187471	0,000025	11,784456	99
1601	50	10	1	1	0,0001	2,370793	0,000026	16,581389	98
1601	50	10	3	1	0,0001	2,475812	0,000042	3,999449	27

Таблица 1.8. Влияние параметров метода. Параболическая функция

Параметры метода						$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	*NMAX*	P_{max}	k_s	k_l	*h*				
33	250	200	6	2	0,001	0,031485	0	0,065634	34
1001	50	10	5	1	0,001	0,000993	0	0,003289	63
1001	50	10	5	1	0,0001	0,002946	0	0,009647	44
1001	50	10	0,5	1	0,0001	0,000006	0	0,000014	100
1001	50	10	0,5	4	0,0001	0,000015	0	0,000051	95
1001	50	10	0,5	0,5	0,0001	0,000014	0	0,000050	95
1001	25	5	0,5	1	0,0001	0,000018	0	0,000054	94
801	50	10	0,5	1	0,0001	0,000016	0	0,000044	95
801	100	20	0,5	1	0,0001	0,000012	0	0,000029	96

Таблица 1.9. Влияние параметров метода. Функция Растригина

Параметры метода						$\overline{\Delta f}$	Δf_{best}	$\bar{\sigma}_f$	$n_{усп}$
NP	$NMAX$	$P_{\max}$	k_s	k_l	h				
1001	50	10	5	1	0,001	0,248143	0	0,398301	35
801	50	10	5	1	0,001	0,159725	0	0,301318	44
801	30	10	5	1	0,001	0,303150	0	0,424992	31
801	70	10	5	1	0,001	0,251336	0	0,397934	41
801	50	20	5	1	0,001	0,124469	0	0,281953	61
801	50	30	5	1	0,001	0,078042	0	0,202496	66
801	50	30	0,5	1	0,001	0,030643	0	0,125429	82
801	50	30	0,5	0,1	0,001	0,029388	0	0,131955	84
801	50	30	0,5	0,01	0,001	0,035308	0	0,153847	87

При анализе работы алгоритма можно увидеть большую зависимость от числа агентов NP в популяции. Увеличивая этот параметр, наблюдается повышение точности получаемого результата при выборе корневой, параболической функции и функции Растригина.

При работе с функцией Растригина метод иногда «застревает» в окрестности локального минимума и не успевает сойтись к точке глобального минимума.

РЕКОМЕНДАЦИИ ПО ВЫБОРУ ПАРАМЕТРОВ

Размер популяции NP определяет количество вычислений целевой функции на каждой итерации. Значение $NP-1$ должно быть кратно четырем. Для задачи с большой областью допустимых решений рекомендуется брать большее значение параметра NP. Рекомендуемые значения параметра $NP \in [501;1601]$.

Число итераций $NMAX$ за время прохода определяет, как долго будет продолжаться поиск новых решений за проход. При увеличении $NMAX$ точность решения увеличивается. Для рассмотренного набора стандартных функций рекомендуемые значения в зависимости от сложности функции $NMAX \in [20;80]$.

Максимальное число проходов $P_{\max}$ определяет, как долго будет продолжаться поиск новых решений. При увеличении количества проходов точность решения увеличивается. Рекомендованное значение этого параметра $P_{\max} \in [10;30]$.

Коэффициент k_s **для матрицы** $\boldsymbol{S}$, определяющей критерий качества управления траекториями агентов. Рекомендованное значение этого параметра $k_s \in [0,1;1]$.

Коэффициент k_l **для матрицы** $\boldsymbol{\Lambda}$, определяющей критерий качества управления траекториями агентов. Рекомендованное значение этого параметра $k_l \in [1;5]$.

Шаг интегрирования дифференциальных уравнений Риккати h. Рекомендованное значение этого параметра $h = 0,0001$.

1.5. МУЛЬТИАГЕНТНЫЙ МЕТОД, ИСПОЛЬЗУЮЩИЙ ПИД-РЕГУЛЯТОРЫ ДЛЯ УПРАВЛЕНИЯ ДВИЖЕНИЕМ АГЕНТОВ

1.5.1. Стратегия поиска решения

Рассматривается задача (1.1) поиска условного глобального минимума целевой функции $f(x)$ на множестве допустимых решений D, т.е. такой точки $x^* \in D$, что

$$f(x^*) = \min_{x \in D} f(x),$$

где $x = (x_1, x_2, ..., x_n)^T$, $D = \{x \mid x_i \in [a_i, b_i], i = 1, 2, ..., n\}$.

Метод основан на процессах, происходящих в среде, имеющей множество *агентов*. Агенты имеют возможность обмениваться информацией для того, чтобы найти решение задачи. Стратегия поиска решения основана на использовании ПИД-регуляторов (пропорционально-интегрально-дифференциальных регуляторов) управления движением агентов [132, 135].

На начальном этапе требуется сгенерировать популяцию из NP агентов на множестве допустимых решений D, используя равномерный закон распределения.

Поиск экстремума реализуется за заданное число проходов $P_{\max}$. В рамках очередного прохода все агенты двигаются под действием соответствующего управления в течение определенного числа итераций.

Уравнение движения агентов

$$\begin{aligned} \frac{dx}{dt} &= v,\ x(t_0) = x_0, \\ \frac{dv}{dt} &= u,\ v(t_0) = v_0, \end{aligned} \tag{1.7}$$

где x – n-мерный вектор положения агента, v – n-мерный вектор скорости агента, t – время, t_0 – начальный момент времени в рамках очередного прохода, x_0 – начальное положение, v_0 – начальная скорость, u – n-мерный вектор управления агентом. При $t_0 = 0$ можно положить $v_0 = o$ (o – нулевой n-мерный вектор).

Обозначим $X = \begin{pmatrix} x \\ v \end{pmatrix}$ – вектор состояния агента и перепишем уравнение (1.7) в форме

$$\frac{d\begin{pmatrix} x \\ v \end{pmatrix}}{dt} = \begin{pmatrix} O_n & E_n \\ O_n & O_n \end{pmatrix} \begin{pmatrix} x \\ v \end{pmatrix} + \begin{pmatrix} O_n \\ E_n \end{pmatrix} u,$$

или

$$\frac{dX}{dt} = AX(t) + Bu(t),\ X(t_0) = X_0 = \begin{pmatrix} x_0 \\ v_0 \end{pmatrix}, \tag{1.8}$$

где O_n, E_n – нулевая и единичная матрицы порядка n, $A = \begin{pmatrix} O_n & E_n \\ O_n & O_n \end{pmatrix}$, $B = \begin{pmatrix} O_n \\ E_n \end{pmatrix}$ – матрицы размеров $(2n \times 2n)$, $(2n \times n)$ соответственно; на управление ограничений не наложено, т.е. $u \in R^n$.

В начальной популяции ($k = 0$), а также в конце каждого k-го прохода, определить положение лидера среди агентов популяции и соответствующее ему минимальное значение целевой функции: $x^{best,k}, f(x^{best,k})$. В течение следующего прохода он не изменяет своего положения:

$$\begin{aligned} &\frac{dx^{best}}{dt} = o,\ x^{best}(t_k) = x^{best,k}, \\ &\frac{dv^{best}}{dt} = o,\ v^{best}(t_k) = o, \end{aligned} \qquad t \in [t_k, t_{k+1}), k = 0, \ldots, P_{\max} - 1, \tag{1.9}$$

где управление лидером $u^{best} = o$, или $\frac{dX^{best}}{dt} = o$, вектор состояния лидера $X^{best} = (x^{best}, v^{best})^T$.

Остальные $(NP-1)$ агентов делятся на четыре равные группы:

- агенты, использующие расширенный ПИД-регулятор (Extended Proportional-Integral-Derivative, EPID) для управления агентами;
- агенты, использующие классический ПИД-регулятор для управления агентами;
- агенты, использующие ПИ-регулятор (пропорционально-интегральный регулятор) для управления агентами;
- агенты, использующие ПД-регулятор (пропорционально-дифференциальный регулятор) для управления агентами.

Для всех агентов каждой группы позиции и векторы скорости разные, но находится и применяется одно и то же управление с полной обратной связью по отклонению вектора состояния агента от вектора состояния лидера.

Введем отклонение от лидера $\Delta X = X - X^{best}$, изменение которого описывается уравнением (вычитая из (1.8) уравнение (1.9)):

$$\frac{d\Delta X}{dt} = A\Delta X(t) + Bu(t), \quad \Delta X(t_k) = \begin{pmatrix} x^k - x^{best,k} \\ v^k \end{pmatrix}, t \in [t_k, t_{k+1}], k = 0, \ldots, P_{\max} - 1, \tag{1.10}$$

где x^k, v^k – положение и скорость агента в конце предыдущего прохода.

Движение первой группы агентов (используется расширенный ПИД-регулятор). Управление $u(t, \Delta X)$ с полной обратной связью для любых начальных состояний имеет вид

$$u_{EPID}(t, \Delta X) = K_P \Delta X + K_{D_1} \dot{\Delta X} + K_{D_2} \ddot{\Delta X} + K_I \int_{t_k}^{t} \Delta X(\tau) d\tau, \tag{1.11}$$

где $K_P, K_{D_1}, K_{D_2}, K_I$ – матрицы размеров $(n \times 2n)$ коэффициентов усиления регулятора, соответствующие пропорциональной, дифференциальным и интегральной составляющим. В отличие от классического ПИД-регулятора имеется слагаемое, учитывающее значения вторых производных отклонения состояния агента от лидера.

Движение второй группы агентов (для всех агентов группы используется классический ПИД-регулятор). Управление $u(t, \Delta X)$ с полной обратной связью для любых начальных состояний имеет вид

$$u_{PID}(t,\Delta X) = K_P\,\Delta X + K_{D_1}\,\dot{\Delta X} + K_I \int_{t_k}^{t} \Delta X(\tau)d\tau. \qquad (1.12)$$

Движение третьей группы агентов (для всех агентов группы применяется ПИ-регулятор). Управление $u(t,\Delta X)$ с полной обратной связью для любых начальных состояний имеет вид

$$u_{PI}(t,\Delta X) = K_P\,\Delta X + K_I \int_{t_k}^{t} \Delta X(\tau)d\tau. \qquad (1.13)$$

Движение четвертой группы агентов (для всех агентов группы применяется ПД-регулятор). Управление $u(t,\Delta X)$ с полной обратной связью для любых начальных состояний имеет вид

$$u_{PD}(t,\Delta X) = K_P\,\Delta X + K_{D_1}\,\dot{\Delta X}. \qquad (1.14)$$

После выполнения $NMAX$ итераций каждый агент предъявляет наилучший из полученных во время движения результатов. Тем самым завершается очередной проход. Затем среди всех агентов популяции снова выбирается лидер и производится новое деление на группы. Процесс повторения проходов завершается при достижении максимального числа проходов.

Здесь $t \in [t_k, t_{k+1}]$, $t_0 = 0$, значение $t_{k+1} = t_k + NMAX \cdot h$, где $NMAX$ – заданное число итераций, h – шаг интегрирования дифференциального уравнения (1.10). Определим моменты времени на промежутке $[t_k, t_{k+1}]$:

$$t_{kj} = t_k + jh,\ j = 0,1,\ldots,NMAX;\ t_{k0} = 0,\ t_{kNMAX} = t_{k+1}.$$

С целью сокращения громоздкости записи обозначим $\Delta X(t_{kj}) = \Delta X_j$.

Для нахождения первых, вторых производных и интегралов в (1.11)–(1.14) используются конечно-разностные формулы [21].

Тогда для нахождения первой производной в правой точке двухточечного шаблона можно использовать формулу

$$\dot{\Delta X}(t_{kj}) = \frac{\Delta X_j - \Delta X_{j-1}}{h},\ j = 1,2,\ldots,NMAX;\quad \dot{\Delta X}(t_{k0}) = 0, \qquad (1.15)$$

имеющую первый порядок аппроксимации относительно шага h.

Для нахождения первой производной в правой точке трехточечного шаблона можно применять формулу

$$\dot{\Delta X}(t_{kj}) = \frac{3\Delta X_j - 4\Delta X_{j-1} + \Delta X_{j-2}}{2h},\ j = 2,3,\ldots,NMAX;\quad \dot{\Delta X}(t_{k0}) = \dot{\Delta X}(t_{k1}) = 0, \qquad (1.16)$$

имеющую второй порядок аппроксимации относительно шага h.

Для нахождения второй производной в правой точке четырехточечного шаблона можно применять формулу

$$\ddot{\Delta X}(t_{kj}) = \frac{2\Delta X_j - 5\Delta X_{j-1} + 4\Delta X_{j-2} - \Delta X_{j-3}}{h^2},\ j = 3,4,\ldots,NMAX;$$
$$\ddot{\Delta X}(t_{k0}) = \ddot{\Delta X}(t_{k1}) = 0, \ddot{\Delta X}(t_{k2}) = 0, \qquad (1.17)$$

имеющую второй порядок аппроксимации относительно шага h.

Для подсчета величины интеграла воспользуемся интегральной формулой трапеций, имеющей второй порядок аппроксимации относительно шага h:

$$\int_{t_k}^{t_{k_2}} \Delta X(\tau) d\tau = \frac{h}{2}[\Delta X_0 + 2\Delta X_1 + \ldots + 2\Delta X_{j-1} + \Delta X_j]. \qquad (1.18)$$

Общая схема работы мультиагентного метода, использующего ПИД-регуляторы для управления движением агентов, представлена на рис. 1.21.

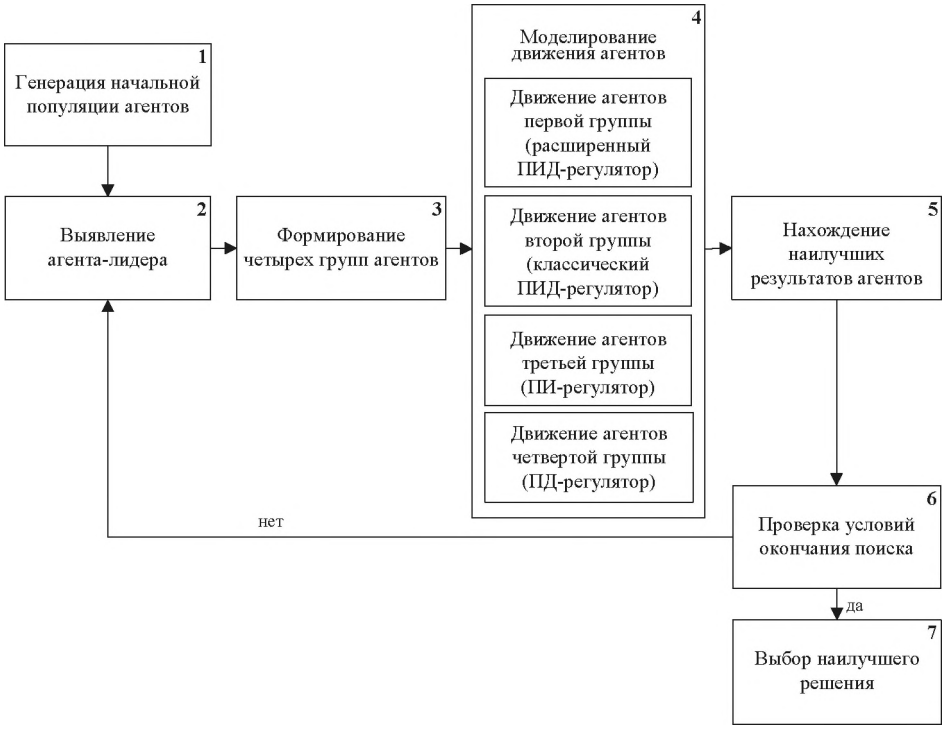


Рис. 1.21. Общая схема работы мультиагентного метода, использующего ПИД-регуляторы для управления движением агентов

1.5.2. Алгоритм решения задачи

Шаг 1. Задать параметры метода:

- размер популяции NP;
- максимальное число проходов $P_{\max}$;
- число итераций за время прохода $NMAX$;
- шаг интегрирования дифференциальных уравнений h;
- $K_P, K_{D_1}, K_{D_2}, K_I$ – матрицы размеров $(n \times 2n)$ коэффициентов усиления регулятора для каждой из четырех групп агентов.

Положить $k = 0$ (счетчик числа итераций), $v^0 = o$, $t_0 = 0$.

Шаг 2. *Формирование начальной популяции агентов.* Генерировать начальную популяцию на множестве D, используя равномерный закон распределения: $x^1,\ldots,x^{NP}$. Вычислить значения целевой функции $f(x^1),\ldots,f(x^{NP})$.

Шаг 3. *Упорядочение агентов в популяции.* Упорядочить популяцию, состоящую из NP агентов, по величине целевой функции.

Шаг 4. *Выбор агента-лидера.* Выбрать агента-лидера и соответствующее наименьшее значение целевой функции: $x^{best,k}$, $f^{best,k}$.

Шаг 5. *Формирование групп агентов.* Поделить всех остальных $(NP-1)$ агентов произвольно на 4 группы.

Для каждого агента составить дифференциальное уравнение

$$\frac{d\Delta X}{dt} = A\,\Delta X(t) + B\,u(t),\ \ \Delta X(t_k) = \begin{pmatrix} x^k - x^{best,k} \\ v^k \end{pmatrix},$$

где $A = \begin{pmatrix} O_n & E_n \\ O_n & O_n \end{pmatrix}$, $B = \begin{pmatrix} O_n \\ E_n \end{pmatrix}$.

Шаг 6. *Движение агентов первой группы* (применяется расширенный ПИД-регулятор).

Шаг 6.1. Сформировать структуру расширенного ПИД-регулятора

$$u_{EPID}(t,\Delta X) = K_P\,\Delta X + K_{D_1}\,\dot{\Delta X} + K_{D_2}\,\ddot{\Delta X} + K_I \int_{t_k}^{t} \Delta X(\tau)d\tau.$$

Для реализации вычислений первой, второй производных и интеграла использовать конечно-разностные формулы (1.15)–(1.18).

Шаг 6.2. Для каждого агента первой группы выполнить следующие действия.

Шаг 6.2.1. Найти решение дифференциального уравнения:

$$\frac{d\Delta X}{dt} = A\,\Delta X(t) + B\,u_{EPID}(t,\Delta X(t)),\ \ \Delta X(t_k) = \begin{pmatrix} x^k - x^{best,k} \\ v^k \end{pmatrix}$$

на промежутке $[t_k, t_{k+1}]$: $\Delta X(t_i), t_i = t_k + ih, i = 0,1,\ldots,NMAX-1$.

Шаг 6.2.2. Найти векторы состояния агента в течение прохода:

$$X^{I,New}(t_i) = X^{best} + \Delta X(t_i),\ i = 0,1,\ldots,NMAX-1.$$

Шаг 6.2.3. Среди всех положений агента в течение прохода выбрать наилучшее за весь период движения, которому соответствует наименьшее значение целевой функции.

Шаг 7. *Движение агентов второй группы* (применяется классический ПИД-регулятор).

Шаг 7.1. Сформировать структуру классического ПИД-регулятора

$$u_{PID}(t,\Delta X) = K_P\,\Delta X + K_{D_1}\,\dot{\Delta X} + K_I \int_{t_k}^{t} \Delta X(\tau)d\tau.$$

Для реализации вычислений первой производной и интеграла использовать конечно-разностные формулы (1.15),(1.16),(1.18).

Шаг 7.2. Для каждого агента второй группы выполнить следующие действия.

Шаг 7.2.1. Найти решение дифференциального уравнения:

$$\frac{d\Delta X}{dt} = A\,\Delta X(t) + B u_{PID}(t, \Delta X(t)),\ \ \Delta X(t_k) = \begin{pmatrix} x^k - x^{best,k} \\ v^k \end{pmatrix}.$$

Шаг 7.2.2. Найти векторы состояния агента в течение прохода:

$$X^{II,New}(t_i) = X^{best} + \Delta X(t_i),\ i = 0,1,...,NMAX - 1.$$

Шаг 7.2.3. Среди всех положений агента в течение прохода выбрать наилучшее за весь период движения, которому соответствует наименьшее значение целевой функции.

Шаг 8. *Движение агентов третьей группы* (применяется ПИ-регулятор).

Шаг 8.1. Сформировать структуру ПИ-регулятора

$$u_{PI}(t, \Delta X) = K_P\,\Delta X + K_I \int_{t_k}^{t} \Delta X(\tau) d\tau.$$

Для нахождения величины интеграла использовать формулу (1.18).

Шаг 8.2. Для каждого агента третьей группы выполнить следующие действия.

Шаг 8.2.1. Найти решение дифференциального уравнения:

$$\frac{d\Delta X}{dt} = A\,\Delta X(t) + B u_{PI}(t, \Delta X(t)),\ \ \Delta X(t_k) = \begin{pmatrix} x^k - x^{best,k} \\ v^k \end{pmatrix}.$$

Шаг 8.2.2. Найти векторы состояния агента в течение прохода:

$$X^{III,New}(t_i) = X^{best} + \Delta X(t_i),\ i = 0,1,...,NMAX - 1.$$

Шаг 8.2.3. Среди всех положений агента в течение прохода выбрать наилучшее за весь период движения, которому соответствует наименьшее значение целевой функции.

Шаг 9. *Движение агентов четвертой группы* (применяется ПД-регулятор).

Шаг 9.1. Сформировать структуру ПД-регулятора

$$u_{PD}(t, \Delta X) = K_P\,\Delta X + K_{D_1}\,\dot{\Delta X}.$$

Для реализации вычислений первой производной использовать конечно-разностные формулы (1.15) или (1.16).

Шаг 9.2. Для каждого агента четвертой группы выполнить следующие действия.

Шаг 9.2.1. Найти решение дифференциального уравнения:

$$\frac{d\Delta X}{dt} = A\,\Delta X(t) + B u_{PD}(t, \Delta X(t)),\ \ \Delta X(t_k) = \begin{pmatrix} x^k - x^{best,k} \\ v^k \end{pmatrix}.$$

Шаг 9.2.2. Найти векторы состояния агента в течение прохода:

$$X^{IV,New}(t_i) = X^{best} + \Delta X(t_i),\ i = 0,1,...,NMAX - 1.$$

Шаг 9.2.3. Среди всех положений агента в течение прохода выбрать наилучшее за весь период движения, которому соответствует наименьшее значение целевой функции.

Результатом шагов 6–8 являются положения NP агентов популяции, из которых $(NP-1)$ агентов с новым положением в результате движения под действием управления в рамках фиксированной группы и агент-лидер x^{best}, который не изменял своего положения за время последнего прохода.

Шаг 10. *Проверка условий окончания глобального поиска.*

Если $k < P_{\max} - 1$, то поиск продолжить, перейти к шагу 3 (упорядочить популяцию агентов, выявить текущего лидера, перетасовать группы) и положить $k = k + 1$.

Если $k \geq P_{\max} - 1$, то поиск завершить, перейти к шагу 11.

Шаг 11. *Выбор решения из последней популяции.*

Закончить работу алгоритма. В качестве приближенного решения задачи $f(x^*) = \min\limits_{x \in D} f(x)$ выбрать агента с наименьшим значением целевой функции из последней популяции: $x^* \cong \bar{x}^* = \arg\min\limits_{j=1,\ldots,NP} f(x^j)$.

Замечания.

1. Можно положить $K_P = (K_{px} \quad O_n)$, где $K_{px} = k_p E_n$ – диагональная матрица, чтобы учитывать только отклонение от лидера по положению. Допускается аналогично задать структуры матриц K_{D_1}, K_{D_2}, K_I.

2. Для настройки регуляторов можно применить процедуру минимизации по параметрам среднего значения целевой функции по 100 запускам для выбранного набора тестовых функций. Для этого можно применять другие мультиагентные алгоритмы оптимизации.

МОДИФИКАЦИЯ МЕТОДА

После шага 9 в исходном алгоритме предлагается выполнить следующие шаги.

Шаг 10. *Сокращение популяции.*

Расположить агентов текущей популяции по возрастанию соответствующего им значения целевой функции: $I_t = \{x^{(1)}, \ldots, x^{(NP)}\}$, $f(x^{(1)}) = f_{\min}$, и удалить s последних агентов, которым соответствует наихудшие значения целевой функции.

Результатом шага 10 является сокращенная популяция.

Шаг 11. *Проверка условий окончания глобального поиска.*

Если $k < P_{\max} - 1$, то поиск продолжить, перейти к шагу 3 (упорядочить популяцию агентов, выявить текущего лидера, перетасовать группы) и положить $k = k + 1$.

Если $k \geq P_{\max} - 1$, то поиск завершить, перейти к шагу 11.

Шаг 12. *Пополнение популяции.*

Шаг 12.1. Генерировать популяцию, состоящую из s агентов, на множестве D, используя равномерный закон распределения: $x^1, \ldots, x^s$.

Шаг 12.2. Расположить агентов популяции по возрастанию соответствующего им значения целевой функции: $I_t = \{x^{(1)}, \ldots, x^{(NP)}\}$, $f(x^{(1)}) = f_{\min}$. Перейти к шагу 4 исходного алгоритма.

Шаг 13. *Выбор решения из последней популяции.*

Закончить работу алгоритма. В качестве приближенного решения задачи $f(x^*) = \min\limits_{x \in D} f(x)$ выбрать агента с наименьшим значением целевой функции из текущей популяции: $x^* \cong x^{(1)}$.

1.5.3. Программное обеспечение

На основе изложенного алгоритма разработана программа поиска глобального экстремума функций [16]. Для ее создания использовалась среда разработки Microsoft Visual Studio, язык программирования C#. На рис. 1.22 представлено главное окно мультиагентного метода, использующего ПИД-регуляторы для управления движением агентов, а на рис. 1.23 окно его пошаговой работы.

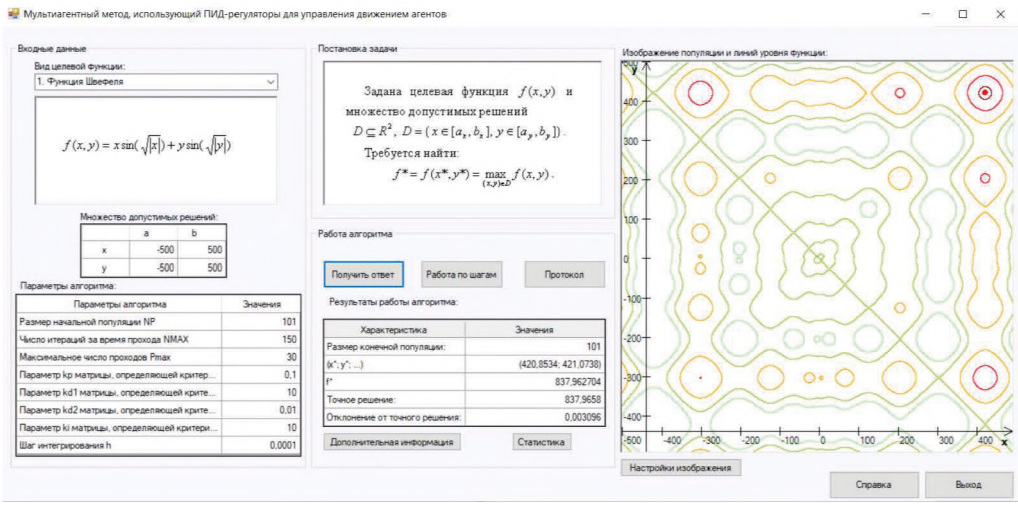


Рис. 1.22. Главное окно программы мультиагентного метода, использующего ПИД-регуляторы для управления движением агентов

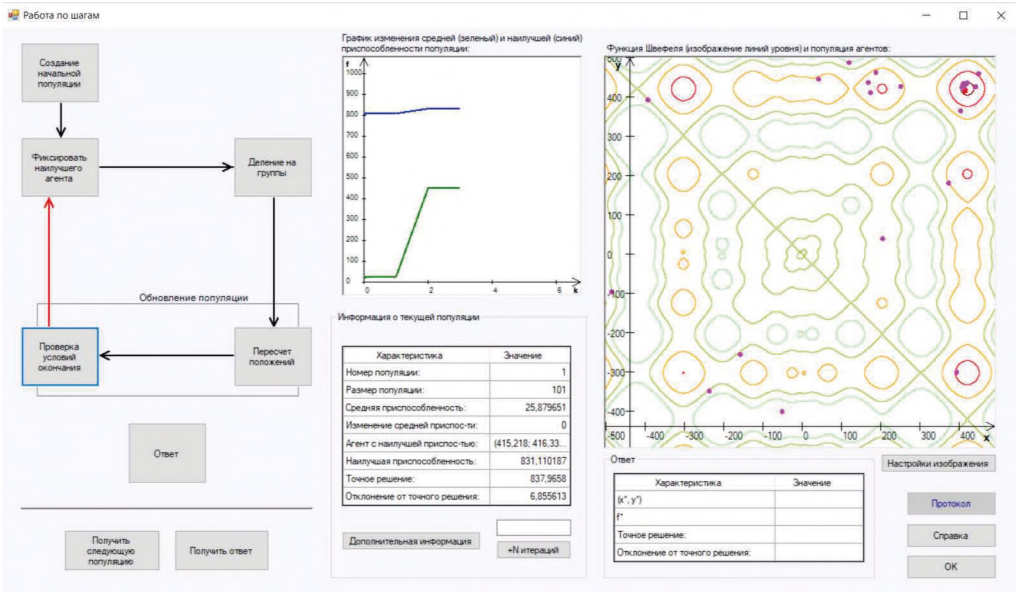


Рис. 1.23. Окно пошаговой работы мультиагентного метода, использующего ПИД-регуляторы для управления движением агентов

Возможности программы позволяют изучить алгоритм метода, а также влияние параметров метода на результат его работы. В список выбираемых функций включены стандартные многоэкстремальные тестовые функции двух переменных, для которых известно точное решение (табл. П.1).

Описанный в данном разделе метод соответствуют задаче поиска минимума целевой функции. Так как заданные тестовые функции имеют глобальный максимум, при разработке программного обеспечения был заменен знак перед целевыми функциями на противоположный.

В главном окне пользователь может выбрать вид целевой функции, задать множество допустимых решений и параметры метода, просматривать результаты работы.

В окне пошаговой работы метода изображена общая схема метода, отображаются результаты работы после выполнения каждого шага (график изменения наилучшего значения целевой функции, графическое изображение популяции) и после завершения работы метода.

1.5.4. Тестовые примеры

Пример 1.5. Найдем глобальный максимум функции Швефеля (табл. П.1). Зададим множество допустимых решений $x, y \in [-500; 500]$. Выберем следующие параметры алгоритма:

- размер популяции $NP = 501$;
- максимальное число проходов $P_{\max} = 30$;
- число итераций за время прохода $NMAX = 20$;
- шаг интегрирования дифференциальных уравнений $h = 0,0001$;
- параметр матрицы, определяющей структуру регуляторов, $k_P = 0,1$;
- параметр матрицы, определяющей структуру регуляторов, $k_{D_1} = 10$;
- параметр матрицы, определяющей структуру регуляторов, $k_{D_2} = 0,01$;
- параметр матрицы, определяющей структуру регуляторов, $k_I = 10$.

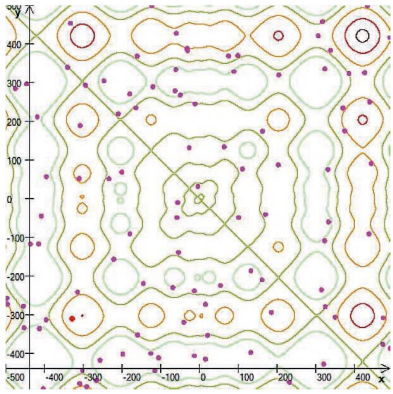
На рис. 1.24 представлена популяция на начальном ($k = 1$), промежуточных ($k = 10$, $k = 20$) и конечном ($k = 30$) проходе. Красным цветом обозначено наилучшее решение в популяции на текущем проходе.

Результаты работы алгоритма:

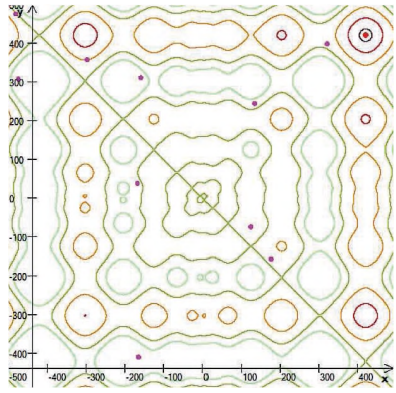
- наилучшее решение $(x^*; y^*) = (420,980078; 420,980078)$;
- значение целевой функции $f(x^*, y^*) = 837,965738$;
- отклонение от точного решения $\Delta = 6,2 \cdot 10^{-5}$.

Полученные результаты свидетельствуют о способности разработанного метода находить точку глобального экстремума целевой функции со сложной структурой линий уровня. Для получения решения достаточно хорошего качества требуется небольшое число итераций метода.

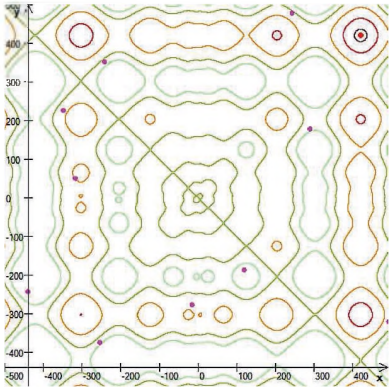
График изменения наилучшего значения целевой функции с ростом числа итераций представлен на рис. 1.25.



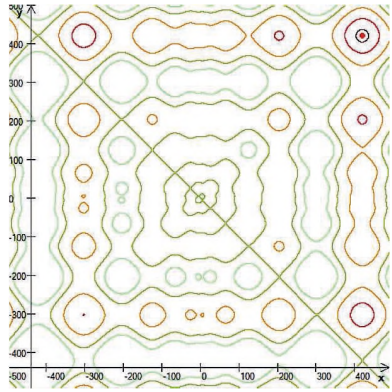
$k = 1$



$k = 10$



$k = 20$



$k = 30$

Рис. 1.24. Начальная, промежуточные и конечная популяции

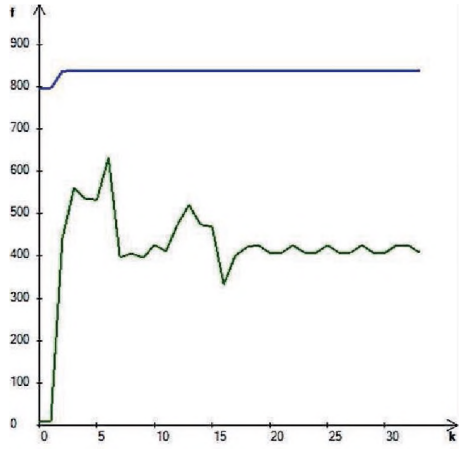


Рис. 1.25. График изменения среднего (зеленый) и наилучшего (синий) значения целевой функции

Пример 1.6. Найдем глобальный максимум функции Экли (табл. П.1). Зададим множество допустимых решений $x, y \in [-5;5]$. Выберем следующие параметры:

- размер популяции $NP = 501$;
- максимальное число проходов $P_{\max} = 30$;
- число итераций за время прохода $NMAX = 100$;
- шаг интегрирования дифференциальных уравнений $h = 0,0001$;
- параметр матрицы, определяющей структуру регуляторов, $k_P = 0,1$;
- параметр матрицы, определяющей структуру регуляторов, $k_{D_1} = 1$;
- параметр матрицы, определяющей структуру регуляторов, $k_{D_2} = 0,05$;
- параметр матрицы, определяющей структуру регуляторов, $k_I = 0,1$.

На рис. 1.26 представлена популяция на начальном ($k = 1$), промежуточных ($k = 10, k = 20$) и конечном ($k = 30$) проходе. Красным цветом обозначено наилучшее решение в популяции на данном проходе.

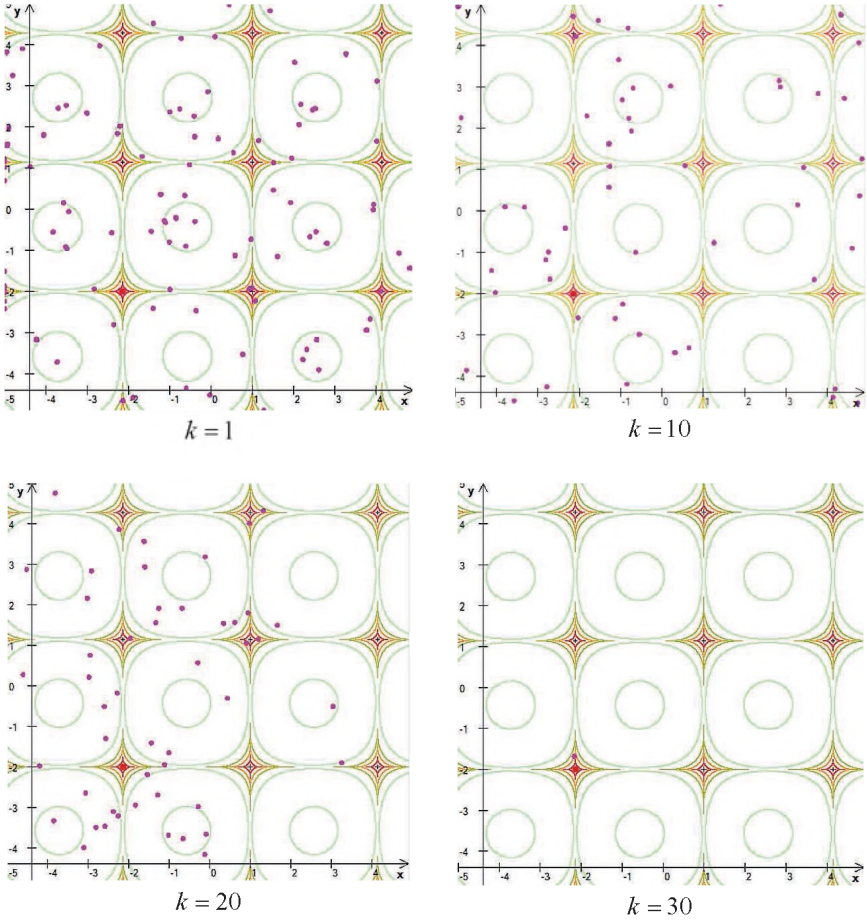


Рис. 1.26. Начальная, промежуточные и конечная популяции

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (-2{,}1416; -2)$;
- значение целевой функции $f(x^*, y^*) = 0{,}952341$;
- отклонение от точного решения $\Delta = 0{,}047659$.

График изменения наилучшего значения целевой функции представлен на рис. 1.27.

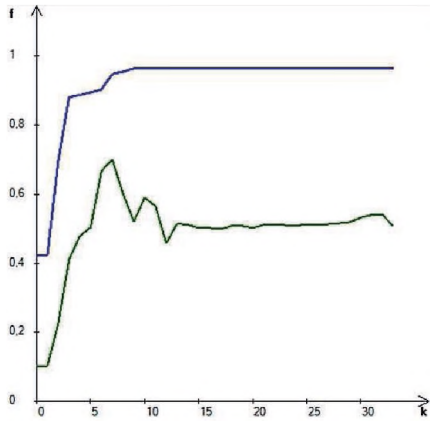


Рис. 1.27. График изменения среднего (зеленый) и наилучшего (синий) значения целевой функции

1.5.5. Анализ эффективности метода

АНАЛИЗ РАБОТЫ МЕТОДА ПРИ РАЗЛИЧНЫХ ЗНАЧЕНИЯХ ПАРАМЕТРОВ

В данном разделе приводится статистический анализ и сравнение работы мультиагентного метода, использующего ПИД-регуляторы для управления движением агентов, при различных значениях его параметров. Исследуемый метод применялся к некоторым функциям из набора тестовых функций (табл. П.1). Для каждой функции проводились серии из 100 решений одной и той же задачи с одними и теми же значениями параметров. Для полученной выборки $\{f^1, f^2, ..., f^{100}\}$ вычислялись среднее значение отклонения полученного решения от точного $\overline{\Delta f} = \frac{1}{100}\sum_{i=1}^{100} \Delta f_i$, где $\Delta f_i = \left| f(x^*) - f^i \right|$; наименьшее значение отклонения $\Delta f_{best} = \min_i \Delta f_i$; среднеквадратическое отклонение $\overline{\sigma}_f = \sqrt{S_{100}}$, где $\overline{S}_{100} = \frac{1}{100}\sum_{i=1}^{100} (\Delta f^i - \overline{\Delta f})^2$; количество успехов $n_{\text{усп}}$ (успехом считалось попадание лучшей точки в ε-окрестность точного решения, $\varepsilon = \frac{\max\limits_{i=1,\ldots,n} |b_i - a_i|}{1000}$). Результаты, полученные для каждой функции, представлены в табл. 1.10–1.12.

Таблица 1.10. Влияние параметров метода. Функция Шафера

Параметры метода								$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	*NMAX*	P_{max}	k_P	k_{D_1}	k_{D_2}	k_I	*h*				
101	10	5	0,0001	0,0001	0,05	0,1	0,0001	0,002693	0,00000	0,002928	47
501	10	5	0,0001	0,0001	0,05	0,1	0,0001	0,001678	0,00000	0,002306	65
501	10	5	0,00001	0,0001	0,05	0,1	0,0001	0,001570	0,00000	0,002276	67
501	10	5	0,001	0,0001	0,05	0,1	0,0001	0,001692	0,00000	0,002203	49
501	10	5	0,0001	0,00001	0,05	0,1	0,0001	0,004593	0,00000	0,001152	6
501	10	5	0,0001	0,0005	0,05	0,1	0,0001	0,004682	0,00000	0,000912	3
501	10	5	0,0001	0,0001	0,005	0,1	0,0001	0,001571	0,00000	0,002274	67
501	10	5	0,0001	0,0001	0,0005	0,1	0,0001	0,001629	0,00000	0,002289	66
501	10	5	0,0001	0,0001	0,005	0,05	0,0001	0,001858	0,00000	0,002369	62
501	10	5	0,0001	0,0001	0,005	0,2	0,0001	0,001615	0,00000	0,002294	67
501	30	5	0,0001	0,0001	0,005	0,2	0,0001	0,001713	0,00000	0,002327	65
501	10	10	0,0001	0,0001	0,005	0,2	0,0001	0,000440	0,00000	0,001398	91
101	10	15	0,0001	0,0001	0,005	0,2	0,0001	0,000977	0,00000	0,001954	80

Таблица 1.11. Влияние параметров метода. Функция Кожа

Параметры метода								$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	*NMAX*	P_{max}	k_P	k_{D_1}	k_{D_2}	k_I	*h*				
501	100	15	0,1	1	0,01	10	0,0001	0,104567	0,000000	0,188489	35
501	10	15	0,1	1	0,01	10	0,0001	0,075384	0,000001	0,131991	33
501	10	15	0,01	1	0,01	10	0,0001	0,241678	0,000011	0,213662	8
501	10	15	1	1	0,01	10	0,0001	0,451651	0,009620	0,258849	0
501	10	15	0,1	0,0001	0,01	10	0,0001	0,010301	– 0,000001	0,020437	78
501	10	15	0,1	0,0001	1	10	0,0001	0,016208	– 0,000001	0,027474	68
501	10	15	0,1	0,0001	0,0001	10	0,0001	0,005095	– 0,000001	0,014415	88
501	10	15	0,1	0,0001	0,0001	0.1	0,0001	0,020667	– 0,000001	0,068058	74
501	10	15	0,1	0,0001	0,0001	5	0,0001	0,009579	– 0,000001	0,019669	80
501	10	30	0,1	0,0001	0,0001	10	0,0001	0,019891	– 0,000001	0,075995	80

Таблица 1.12. Влияние параметров метода. Параболическая функция

Параметры метода								$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	*NMAX*	P_{max}	k_P	k_{D_1}	k_{D_2}	k_I	*h*				
101	10	10	0,1	10	0,1	10	0,0001	0,000010	0,000000	0,000032	97
41	10	10	0,1	10	0,1	10	0,0001	0,000636	0,000000	0,002839	82
41	10	10	0,01	10	0,1	10	0,0001	0,139466	0,000004	0,271471	12
41	10	10	1	10	0,1	10	0,0001	0,385972	0,001888	0,420179	0
41	10	10	0,1	1	0,1	10	0,0001	0,000461	0,000000	0,002025	85
41	10	10	0,1	0,001	0,1	10	0,0001	0,000059	0,000000	0,000303	92
41	10	10	0,1	0,0001	0,1	10	0,0001	0,000000	0,000000	0,000000	100
41	10	10	0,1	0,0001	0,001	10	0,0001	0,000000	0,000000	0,000000	100
41	10	10	0,1	0,0001	10	10	0,0001	0,000000	0,000000	0,000001	100
41	10	10	0,1	0,0001	10	0,1	0,0001	0,000001	0,000000	0,000005	100

Анализ работы мультиагентного алгоритма, основанного на использовании ПИД-регуляторов управления движением агентов, подтвердил его эффективность на стандартном наборе тестовых функций двух переменных. Проведена серия, состоящая из 100 запусков, в ходе которой были найдены точки глобального экстремума некоторых функций с достаточно высокой точностью.

В ходе анализа работы алгоритма наблюдалась чувствительность метода к параметрам k_P, k_{D_1}, k_{D_2}, k_I, поэтому требуется их тщательная настройка. В процессе тестирования алгоритма были подобраны наилучшие значения параметров, при которых получается близкое к точному решение в большинстве запусков.

РЕКОМЕНДАЦИИ ПО ВЫБОРУ ПАРАМЕТРОВ

Размер популяции NP определяет количество вычислений целевой функции на каждой итерации. Значение $NP-1$ должно быть кратно четырем. Для задачи с большой областью допустимых решений или со сложной структурой линий уровня рекомендуется брать большее значение параметра NP. Рекомендуемые значения параметра $NP \in [41;501]$.

Число итераций $NMAX$ за время прохода определяет, как долго будет продолжаться поиск новых решений за проход. При увеличении $NMAX$ точность решения не слишком возрастает. Для рассмотренного набора стандартных функций рекомендуемые значения в зависимости от сложности функции $NMAX \in [10;50]$.

Максимальное число проходов $P_{\max}$ определяет, как долго будет продолжаться поиск новых решений. При увеличении количества проходов точность решения увеличивается, но также возрастают и вычислительные затраты. Рекомендованное значение этого параметра $P_{\max} \in [5;30]$.

Коэффициент k_P **для матрицы** K_P, определяющей структуру регуляторов для управления движением агентов. Рекомендованное значение этого параметра $k_P \in [0,0001;0,1]$.

Коэффициент k_{D_1} **для матрицы** K_{D_1}, определяющей структуру регуляторов для управления движением агентов. Рекомендованное значение этого параметра $k_{D_1} = 0,0001$.

Коэффициент k_{D_2} **для матрицы** K_{D_2}, определяющей структуру регуляторов для управления движением агентов. Рекомендованное значение этого параметра $k_{D_2} \in [0,0001;0,1]$.

Коэффициент k_I **для матрицы** K_I, определяющей структуру регуляторов для управления движением агентов. Рекомендованное значение этого параметра $k_I \in [0,1;10]$.

Шаг интегрирования дифференциальных уравнений Риккати h. Рекомендованное значение этого параметра $h = 0,0001$.

1.6. ПОСЛЕДОВАТЕЛЬНО-ПАРАЛЛЕЛЬНЫЙ ГИБРИДНЫЙ МУЛЬТИАГЕНТНЫЙ МЕТОД, ОСНОВАННЫЙ НА АЛГОРИТМАХ, ИМИТИРУЮЩИХ ИМПЕРИАЛИСТИЧЕСКУЮ КОНКУРЕНЦИЮ, ПОВЕДЕНИЕ СТАЙ РЫБ И КРИЛЯ

1.6.1. Стратегия поиска решения

Рассматривается задача (1.1) поиска условного глобального минимума целевой функции $f(x)$ на множестве допустимых решений D, т.е. такой точки $x^* \in D$, что

$$f(x^*) = \min_{x \in D} f(x),$$

где $x = (x_1, x_2, ..., x_n)^T$, $D = \{x \mid x_i \in [a_i, b_i], i = 1, 2, ..., n\}$.

СТРАТЕГИЯ ПОСЛЕДОВАТЕЛЬНО-ПАРАЛЛЕЛЬНОГО ГИБРИДНОГО МУЛЬТИАГЕНТНОГО МЕТОДА

Среди мультиагентных методов оптимизации можно выделить два метода, относящихся одновременно к биоинспирированным алгоритмам: методы, имитирующие поведение стай рыб и стай криля [64, 66–68, 89, 122]. Они хорошо зарекомендовали себя при решении различных технических задач [41, 42]. Каждый из них, как и все метаэвристические алгоритмы, обладает своими преимуществами и недостатками. Среди алгоритмов, порожденных наблюдением за социальными процессами, можно выделить алгоритм, имитирующий империалистическую конкуренцию. Анализ его работы показал, что он успешно находит область глобального экстремума, но не всегда попадает в точку глобального экстремума за фиксированное число итераций с достаточной точностью. Поэтому предлагается использовать данный алгоритм для поиска хорошего начального приближения, а для уточнения решения применять два метода, упомянутые выше.

Предлагаемая стратегия включает в себя следующие этапы:

- формирование начальной популяции решений с помощью равномерного закона распределения на множестве допустимых решений;
- получение начального приближения с помощью алгоритма, имитирующего империалистическую конкуренцию. Среди решений, образующих начальную популяцию, выделяются несколько лидеров (они называются империями). Остальные решения называются колониями. В процессе конкурентной борьбы империи стремятся увеличить число своих колоний так, что в конце остается только одна империя с принадлежащими ей колониями. При этом поиск завершается;
- полученная популяция решений используется в качестве начальной для одного из двух алгоритмов, имитирующих поведение стай криля или стай рыб. Выбор алгоритма производится случайно с некоторой вероятностью, отражающей степень предпочтения при выборе. При помощи выбранного алгоритма производится уточнение результата поиска экстремума в течение заданного числа итераций, образующих проход;
- после завершения прохода случайным образом выбирается алгоритм для продолжения эволюционного процесса. Таким образом, может быть приня-

то решение о продолжении итерационного процесса тем же алгоритмом или о смене алгоритма (при этом полученная популяция агентов передается в качестве начальной в соответствующий алгоритм);

- по завершении заданного числа проходов процесс завершается, и из последней популяции выбирается наилучшее решение.

Замечания.

1. Проход может быть завершен при выполнении условия текущей неэффективности метода, когда в течение заданного числа итераций наилучшее решение в популяции не обновляется.

2. При переходе от одного алгоритма к другому необязательно требуется сохранять то же самое число членов популяции. Можно исключить фиксированное число наихудших решений, а на их место генерировать новые способом, применяемым при формировании начальной популяции.

3. Одним из вариантов после реализации первого этапа методом, имитирующим империалистическую конкуренцию, может быть использование полученной популяции решений в качестве начальной популяции для параллельного применения методов имитации поведения стаи криля и стаи рыб. По окончании прохода полученные обоими методами популяции объединяются в одну. В ней решения ранжируются по величине целевой функции и отбираются NP наилучших решений для продолжения процесса поиска. В процессе отбора допускается организовать последовательное попарное сравнение решений для исключения расположенных слишком близко друг к другу.

Общая схема работы последовательно-параллельного гибридного метода, основанного на алгоритмах, имитирующих империалистическую конкуренцию и поведение стай криля и рыб, представлена на рис. 1.28.

Рис. 1.28. Общая схема работы последовательно-параллельного гибридного мультиагентного метода, основанного на алгоритмах, имитирующих империалистическую конкуренцию и поведение стай криля и рыб

В стратегии (*Imperialist Competitive Algorithm* – ICA) используются наблюдения за поведением империй в борьбе за сферы влияния [63]. Империализм – это политика расширения управляющего воздействия правительства за пределы границ страны, реализуемая как посредством непосредственного управления, так и косвенного через влияние на рынки продовольствия, товаров, материалов и т.д. Все страны делятся на империи и колонии. Империи стремятся использовать ресурсы колоний, противодействуя другим империям, и распространить свое влияние на весь мир.

Метод использует идеи, как эволюционных алгоритмов, так и методов «роевого интеллекта». Он начинается с формирования начальной популяции – стран в мире (решений на множестве допустимых решений). Несколько лучших стран (по величине целевой функции) отбираются на роль *империалистических стран*, а остальные образуют *колонии*. Все колонии закрепляются за империалистическими государствами, причем их количество определяется силой такого государства, обратно пропорциональной величине целевой функции. Так образуются *империи*: империалистические государства и их колонии. Наиболее сильному империалистическому государству соответствует наибольшее число колоний. Затем каждая колония начинает движение к своему империалистическому государству. Сила империи определяется силой империалистического государства и его колоний (к силе государства добавляется доля от средней силы колоний). Конкуренция между империями приводит либо к возрастанию (по крайней мере, неубыванию) силы империи, либо к ее уменьшению. Слабые империи со временем исчезают. Описанные механизмы должны привести к ситуации, когда остается только одна империя в мире, а все остальные страны являются ее колониями (это является условием окончания процесса). Положение империалистического государства принимается за приближенное решение задачи. Общая схема работы метода представлена на рис. 1.29.

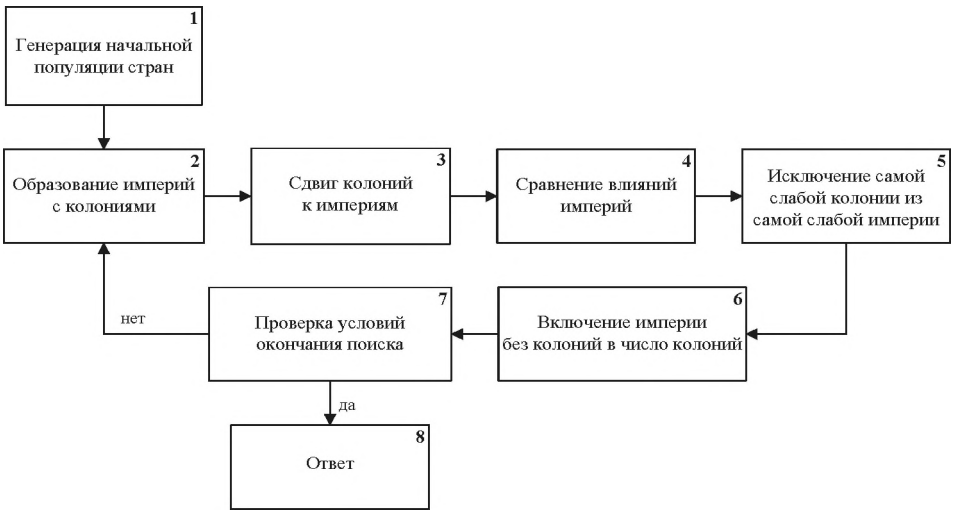


Рис. 1.29. Общая схема работы метода, имитирующего империалистическую конкуренцию

Метод, имитирующий поведение стаи криля (Krill Herd – KH), относится к биоинспирированным, поскольку основан на результатах анализа поведения стай криля – рачков, внешне напоминающих креветок [64, 89]. Их позиции меняются итерационно под действием трех факторов: присутствия других членов популяции, необходимости поиска пищи, случайных блужданий. Обычно движение популяции криля определяется двумя целями: увеличением плотности криля и достижением пищи.

В начале процесса генерируется популяция из NP особей на множестве D с помощью равномерного закона распределения. Предполагается, что движение j-го члена популяции происходит согласно уравнению $\frac{dx^j}{dt} = V^j$, где x^j – положение, а V^j – скорость, которая складывается из трех составляющих. Первая составляющая определяется влиянием соседей (членов популяции, входящих в некоторую окрестность j-го элемента определенного радиуса), наилучшего элемента во всей популяции и информации о своей старой скорости. Вторая составляющая определяется движением в сторону источника пищи (за него принимается «центр масс» популяции), информацией о старой скорости в поисках пищи, памятью своего наилучшего результата за все предыдущие итерации. Третья составляющая имитирует случайные блуждания особи, уменьшающиеся с ростом числа итераций. Нахождению результирующей скорости из трех составляющих соответствует завершение *фазы питания* стаи криля. Далее реализуется *фаза плавания*, в результате которого находятся новые положения членов стаи. Для оживления процесса поиска применяются *операции скрещивания и мутации*, используемые в других эволюционных методах, в том числе в методе дифференциальной эволюции. Процедура поиска завершается при достижении заданного числа итераций.

Общая схема работы метода, имитирующего поведение стаи криля, представлена на рис. 1.30.

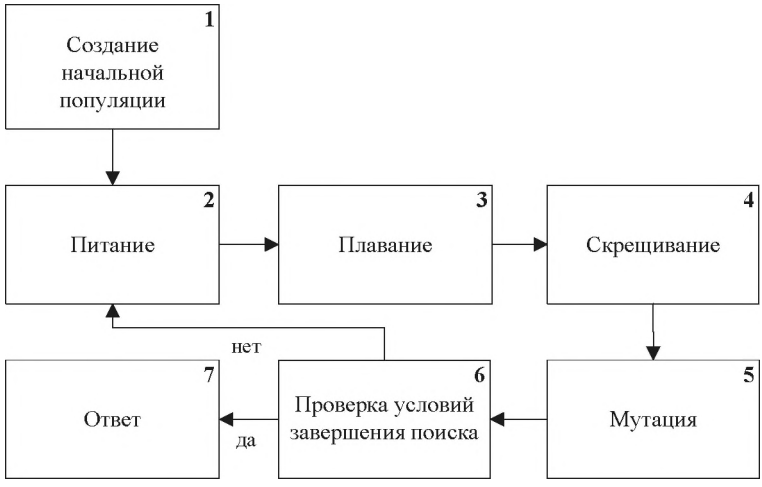


Рис. 1.30. Общая схема работы метода, имитирующего поведение стаи криля

В алгоритме, имитирующем поведение стаи рыб (Fish School Search Algorithm, FSS) [66–68], используются результаты изучения особенностей поведения некоторых сортов рыб, которые могут существовать только в пределах стаи, что уменьшает индивидуальную свободу их передвижений, но увеличивает интенсивность соревнования за пищу. Такое объединение рыб, как показывают наблюдения за ними в океанах и реках, подтверждает, что преимущества объединения существенно превышают недостатки.

В алгоритме используются следующие основные черты поведения стай рыб:

1) *питание* (производится имитация естественного инстинкта рыб, заключающегося в поиске пищи). Оно необходимо, так как рыбы должны питаться, для того чтобы вырасти сильными и способными к размножению. Получая пищу, рыбы набирают вес, а плавая – теряют;

2) *плавание* (эта функция реализуется коллективно всеми рыбами стаи с целью поиска пищи);

3) *размножение* (производится имитация естественного механизма селекции, порождая новые объекты для поддержания процесса поиска).

Каждая рыба из стаи имеет внутреннюю «память» о случившемся успехе в поиске пищи (приближении к точке экстремума), заключенную в весе рыбы. Стая эволюционирует путем обмена информацией между родителями в результате размножения, а также вследствие коллективного движения. Аквариум представляет собой множество допустимых решений D. Наличие пищи (привлекательные значения целевой функции) показывает рыбам области аквариума, которые определяют хорошие регионы для поиска решения. В процессе плавания реализуется идея глобального перенаправления всех рыб в ту часть аквариума, которая рассматривается всеми рыбами стаи как наиболее предпочтительная с точки зрения поиска пищи. К процессу размножения допускаются рыбы только с наибольшим весом. Размножение рыб, в свою очередь, позволяет перейти от сравнительного исследования областей аквариума к процессу, уточняющему решение в рамках найденной области.

В процессе поиска решений сначала генерируется начальная популяция из NP рыб – стая на множестве D с помощью равномерного закона распределения.

Далее рыбы при поиске пищи (определении точки экстремума) используют следующие операции.

Операция питания. Рыбы ищут пищу, которая разбросана в аквариуме в различных концентрациях. Для поиска пищи рыбы выполняют индивидуальные движения. В итоге рыба может прибавить в весе или наоборот потерять в зависимости от результата поиска. Предполагается, что приращение веса рыбы пропорционально нормализованному приращению значения целевой функции

$$W^{j,k+1} = W^{j,k} + \frac{f(x^{j,k+1}) - f(x^{j,k})}{\max\limits_{m=1,\ldots,NP} \left\{ \left| f(x^{m,k+1}) - f(x^{m,k}) \right| \right\}}, \qquad (1.19)$$

где $W^{j,k+1}, W^{j,k}$ – вес рыбы с номером j на $(k+1)$-й и k-й итерациях; $f(x^{j,k+1}), f(x^{j,k})$ – значения целевой функции, соответствующие новому $x^{j,k+1}$ и текущему $x^{j,k}$ положениям рыбы. Вес рыбы, как правило, изменяется от 1 до величины W_{scale}. При $k = 0$ вес всех рыб полагают одинаковым и равным $\frac{W_{scale}}{2}$.

Операция плавания:

а) каждая отдельно взятая рыба совершает *индивидуальное движение* в случайном направлении так, чтобы значение целевой функции возрастало. Если новое положение рыбы не входит в множество D (аквариум), то движения не происходит.

Новое положение каждой рыбы ($j = 1, \ldots, NP$) находится по формуле

$$x^{j,k+1} = x^{j,k} + step^{k}_{ind} \cdot \delta, \tag{1.20}$$

где $\delta \sim R[-0,5;0,5]$ – вектор, координаты которого распределены по равномерному закону на отрезке $[-0,5;0,5]$, $step^{k}_{ind}$ – индивидуальный шаг, величина которого линейно уменьшается с ростом числа итераций. Новое положение принимается, если значение целевой функции увеличивается (при выполнении ограничения на принадлежность множеству D), иначе рыба не двигается;

б) после того, как каждая рыба передвинулась, начинается *фаза инстинктивно-коллективного плавания*. Для ее реализации находятся векторы сдвигов каждой рыбы:

$$\Delta x^{j}_{ind} = x^{j,k+1} - x^{j,k}, \tag{1.21}$$

где j – номер рыбы.

Каждая рыба стаи изменяет свое положение по формуле

$$x^{j,k+1} = x^{j,k} + \frac{\sum_{m=1}^{NP} \Delta x^{m}_{ind} \cdot \left[f(x^{m,k+1}) - f(x^{m,k}) \right]}{\sum_{m=1}^{NP} \left[f(x^{m,k+1}) - f(x^{m,k}) \right]}, j = 1, \ldots, NP. \tag{1.22}$$

Это означает, что рыбы, сделавшие успешное движение, определяют результирующее направление движения более чем остальные. После выполнения движения рыба питается;

в) далее реализуется третья фаза плавания – *коллективно-волевое движение*, основанное на учете движения всей стаи. Если стая набрала вес в случае успешного перемещения, то радиус стаи увеличивается, а если нет, то уменьшается. Положение каждой рыбы изменяется по отношению к центру тяжести стаи (барицентру), определяемому на каждой итерации:

$$Bari(k) = \frac{\sum_{j=1}^{NP} x^{j,k} \cdot W^{j,k}}{\sum_{j=1}^{NP} W^{j,k}}. \tag{1.23}$$

При этом используется величина шага $step^{k}_{vol}$, линейно убывающая по величине с ростом числа итераций.

Если суммарный вес стаи увеличился, то рыбы двигаются по направлению к барицентру:

$$x^{j,k+1} = x^{j,k} - step^{k}_{vol} \cdot rand \cdot \frac{\left[x^{j,k} - Bari(k) \right]}{\left\| x^{j,k} - Bari(k) \right\|}, \tag{1.24}$$

где $rand \sim R[0;1]$.

Если суммарный вес стаи уменьшился, то рыбы двигаются по направлению от барицентра:

$$x^{j,k+1} = x^{j,k} + step_{vol}^{k} \cdot rand \cdot \frac{\left[x^{j,k} - Bari(k)\right]}{\left\|x^{j,k} - Bari(k)\right\|}. \qquad (1.25)$$

Операция размножения. Реализуется механизм селекции, и отбираются наилучшие (с точки зрения веса) представители популяции – рыбы, достигнувшие порогового веса thr ($W^{i,k} > thr$). Для каждой выбранной рыбы ищется пара, т.е. рыба, которой соответствует максимальное отношение ее веса к расстоянию до претендента. Каждая пара рыб с номерами i и j порождает потомка с весом и положением:

$$W^{child,k} = \frac{W^{i,k} + W^{j,k}}{2}, \quad x^{child,k} = \frac{x^{i,k} + x^{j,k}}{2}. \qquad (1.26)$$

Далее родители и потомки ранжируются по величине веса. Для сохранения размера популяции удаляются все рыбы, имеющие наименьший вес (остается NP рыб).

Процесс заканчивается по достижении заданного числа итераций $ITER$. В качестве решения задачи (1.2) выбирается положение рыбы с наибольшим весом.

Общая схема работы метода, имитирующего поведение стаи рыб, представлена на рис. 1.31.

Рис. 1.31. Общая схема работы метода, имитирующего поведение стаи рыб

1.6.2. Алгоритмы решения задачи

АЛГОРИТМ ПОСЛЕДОВАТЕЛЬНО-ПАРАЛЛЕЛЬНОГО ГИБРИДНОГО МУЛЬТИАГЕНТНОГО МЕТОДА

Шаг 1. Задать параметры:

- размер популяции NP;
- максимальное число проходов $P_{\max}$;
- пороговое значение $thr \in (0;1)$ для выбора метода оптимизации;
- число последовательных итераций контроля эффективности метода I_{contr};
- малое положительное число ε_{eff} для досрочного завершения прохода;
- число наилучших решений в популяции $N_{best} \in (0, NP)$.

Инициализировать счетчик числа проходов.

Шаг 2. Задать параметры метода, имитирующего империалистическую конкуренцию и реализовать поиск до выполнения условия окончания. При инициализации начальной популяции стран положить размер популяции равным $N_{pop} = NP$.

Шаг 3. *Реализация последовательного этапа.*

Шаг 3.1. Генерировать параметр ζ с помощью равномерного закона распределения на отрезке [0;1]. Если $\zeta \geq thr$, то выбрать в качестве следующего используемого метода алгоритм, имитирующий поведение стаи криля, а иначе алгоритм, имитирующий поведение стаи рыб.

Шаг 3.2. Реализовать поиск экстремума выбранным алгоритмом в течение одного прохода, задаваемого максимальным числом итераций, определяемым величиной $I_{\max}$ (метод стаи криля) или $ITER$ (метод стаи рыб). Проход может быть завершен досрочно при выполнении условия текущей неэффективности метода, когда в течение заданного числа итераций I_{contr} изменение относительной величины приращения наилучшего значения целевой функции в популяции обновляется менее, чем на ε_{eff}.

Шаг 4. *Реализация параллельного этапа.*

Шаг 4.1. Если в результате двух последовательно реализованных проходов не было зафиксировано досрочного окончания процесса поиска, то следует перейти к шагу 4.4.

Иначе необходимо из текущей популяции решений выбрать N_{best} наилучших решений. Остальные $NP - N_{best}$ решений генерировать заново с помощью равномерного закона распределения на множестве допустимых решений.

Шаг 4.2. Полученную обновленную популяцию решений использовать в качестве начальной популяции для параллельного применения методов имитации поведения стаи криля и стаи рыб. По окончании прохода полученные обоими методами популяции объединить в одну. В ней решения ранжировать по величине целевой функции и отобрать NP наилучших решений для продолжения процесса поиска. В процессе отбора допускается организовать последовательное попарное сравнение решений для исключения расположенных слишком близко друг к другу.

Шаг 4.3. Выбрать равновероятно один из способов продолжения поиска. Если выбрано параллельное применение, то перейти к шагу 4.2. Иначе перейти к шагу 4.4.

Шаг 4.4. Генерировать параметр ζ с помощью равномерного закона распределения на отрезке [0;1]. Если $\zeta \geq thr$, то выбрать в качестве следующего используе-

мого метода алгоритм, имитирующий поведение стаи криля, а иначе алгоритм, имитирующий поведение стаи рыб. Если принимается решение продолжении поиска тем же самым алгоритмом оптимизации, то из популяции следует исключить фиксированное число наихудших решений, а на их место генерировать новые способом, применяемым при формировании начальной популяции. Иначе (в случае замены алгоритма) в качестве начальной популяции решений использовать текущую популяцию.

Шаг 5. *Проверка условия завершения числа проходов.* Если число $P_{\max}$ достигнуто, в качестве решения задачи выбрать наилучшее решение из последней популяции. Иначе перейти к шагу 4.

АЛГОРИТМ, ИМИТИРУЮЩИЙ ИМПЕРИАЛИСТИЧЕСКУЮ КОНКУРЕНЦИЮ

Шаг 1. Задать параметры:

- размер популяции (число стран) N_{pop};
- число империалистических стран N_{imp};
- параметры сдвига колоний β, γ;
- параметр учета влияния колоний ξ.

Шаг 2. *Формирование начальной популяции.* Генерировать N_{pop} стран (решений из множества D) с использованием равномерного закона распределения. Подсчитать значения целевой функции и упорядочить решения (страны) по возрастанию целевой функции: $x^1, x^2, ..., x^{N_{pop}}$ (решению x^1 соответствует наименьшее значение целевой функции).

Шаг 3. *Выбор империалистических государств.* Из числа первых в списке решений, соответствующих наилучшим значениям целевой функции, выбрать N_{imp} решений (стран), называемых далее империалистическими. Подсчитать число колоний $N_{col} = N_{pop} - N_{imp}$.

Шаг 4. *Образование империй.*

Шаг 4.1. Подсчитать нормализованную стоимость каждого империалистического государства: $\tilde{f}(x_{imp}^j) = f(x_{imp}^j) - f(x^{N_{pop}}), j = 1,...,N_{imp}$.

Шаг 4.2. Найти нормализованную силу каждого империалистического государства:

$$p^j = \left| \frac{\tilde{f}(x_{imp}^j)}{\sum_{j=1}^{N_{imp}} \tilde{f}(x_{imp}^j)} \right|, j = 1,...,N_{imp}.$$

Шаг 4.3. Найти число колоний каждого империалистического государства

$$N_c^j = \underset{round}{\left[p^j \cdot N_{col} \right]}, \; j = 1,...,N_{imp} - 1;$$

$$N_c^{N_{imp}} = N_{col} - \sum_{j=1}^{N_{imp}-1} N_c^j,$$

где $round[\cdot]$ – операция округления.

Шаг 4.4. Для каждого империалистического государства выбрать N_c^j стран из числа колоний случайным образом. Империалистическое государство и выбранные колонии образуют империю.

Шаг 5. *Сдвиг колоний империи к империалистическому государству* (процедура ассимиляции). Для каждой империи последовательно выполнить $(j = 1,...,N_{imp})$.

Шаг 5.1. Выбрать первую колонию в империи с номером j случайным образом.

Шаг 5.2. Найти новое положение колонии x^{new}:

$$x^{new} = x^{old} + U(0;\beta \cdot d) \cdot V_1 + d \cdot \mathrm{tg}\theta \cdot V_2,$$

где β, θ – параметры; d – расстояние от колонии до империалистического государства; $U(a,b)$ – равномерно распределенная на $[a,b]$ случайная величина; $\theta = U(-\gamma,\gamma)$; $d = \sqrt{\sum_{i=1}^{n}(x_{imp,i}^j - x_{c,i})^2}$; V_1 – единичный вектор, направленный от колонии к империалистическому государству; V_2 – случайный единичный вектор, перпендикулярный V_1: $\{V_2\} = \left\{\{rand\} \mid V_1 \cdot V_2 = 0, \|V_2\| = 1, \|V_1\| = 1\right\}$; j – номер империалистического государства; x_{imp}^j – положение империалистического государства; x_c – положение колонии.

Таким образом, векторы V_1 и V_2 можно найти следующим образом:

$$V_1 = \frac{1}{d}\left(x_{imp,1}^j - x_{c,1}; ...; x_{imp,n}^j - x_{c,n}\right),$$

$$V_1 \cdot \tilde{V}_2 = V_{1,1} \cdot \tilde{V}_{2,1} + ... + V_{1,n} \cdot \tilde{V}_{2,n} = 0 \text{ (условие ортогональности)},$$

компоненты $\tilde{V}_{2,1},...,\tilde{V}_{2,n-1}$ можно генерировать случайным образом на отрезке $[-1;1]$, а $\tilde{V}_{2,n}$ найти из условия ортогональности. Далее нормировать вектор $\tilde{V}_2 : V_2 = \frac{\tilde{V}_2}{\|\tilde{V}_2\|}$.

Шаг 5.3. Вычислить значение целевой функции $f(x^{new})$.

Если $f(x^{new}) \geq f(x_{imp}^j)$, то перейти к шагу 5.4.

Если $f(x^{new}) < f(x_{imp}^j)$, то положить $x_c = x_{imp}^j$, $x_{imp}^j = x^{new}$ (колония становится империалистическим государством, а бывшее империалистическое государство – колонией).

Перейти к шагу 5.1.

Шаг 5.4. Если число колоний, изменивших положение, достигло N_c^j, то процедуру завершить. Иначе выбрать случайным образом следующую колонию, еще не изменявшую положение, и перейти к шагу 5.2.

Шаг 6. *Конкуренция между империями.*

Шаг 6.1. Найти стоимость империи (*Total Cost, TC*)

$$TC^j = f(x_{imp}^j) + \xi \cdot \frac{\sum_{i=1}^{N_c^j} f(x_c^i)}{N_c^j}, \quad j = 1,...,N,$$

где ξ – положительное число, меньшее 1; x_c^j – положение i-й колонии j-й империи, определяемой положением x_{imp}^j империалистического государства.

Шаг 6.2. Найти нормализованную стоимость империи (*Normalized Total Cost, NTC*):

$$NTC^j = TC^j - \max_{i\in\{1,\ldots,N_{imp}\}}\{TC^i\}.$$

Шаг 6.3. Найти уровень влияния каждой империи:

$$p^j = \left| \frac{NTC^j}{\sum_{i=1}^{N_{imp}} NTC^i} \right|, \quad j = 1,\ldots,N_{imp}.$$

Шаг 6.4. Найти наиболее слабую империю, которой соответствует минимальное значение p^j, $j = 1,\ldots,N_{imp}$.

Шаг 6.5. В найденной империи найти самую слабую колонию с наибольшим значением целевой функции.

Шаг 6.6. Сформировать векторы

$$P = (p^1, p^2, \ldots, p^{N_{imp}});$$

$$R = (r^1, r^2, \ldots, r^{N_{imp}}), \text{ где } r^j = U(0;1);$$

$$D = P - R = (d^1, d^2, \ldots, d^{N_{imp}}) = (p^1 - r^1, \ldots, p^{N_{imp}} - r^{N_{imp}}).$$

Шаг 6.7. Определенную на шаге 6.5 колонию включить в империю, соответствующую наибольшему значению среди компонент вектора-строки D.

Шаг 7. *Распад наислабейших империй.*

Если в империи нет колоний, она прекращает существование (страна включается в империю, определяемую так же, как на шаге 6.7). Находится новое число N_{imp}.

Шаг 8. *Проверка условий окончания процесса поиска.* Если $N_{imp} = 1$, процесс завершить. Решением задачи считать положение империалистического государства. Иначе перейти к шагу 5.

Замечания.

1. Рекомендуемые значения параметров:

$$\beta \cong 2;\ \gamma \cong \pi/4;\ \xi = 0{,}1;\ (N_{pop}, N_{imp}) = (150;15);(80;8);(50;5).$$

2. Можно использовать модификации формулы, определяющей движение колонии. На шаге 5 алгоритма можно применять следующие варианты:

1) $x^{new} = x^{old} + \beta \cdot d \cdot Rand \otimes V_1$,

где $Rand$ – случайный вектор, каждая компонента которого является независимой случайной величиной, равномерно распределенной на $[0;1]$; $\otimes$ – покомпонентное произведение векторов (по Адамару);

2) $x^{new} = x^{old} + \beta \cdot d \cdot Rand \otimes V_1 + d \cdot \mathrm{tg}\theta \cdot V_2$;

3) в каждой империи реализовать движение всех колоний. Затем в качестве нового империалистического государства выбрать колонию с наилучшим значением целевой функции (если оно меньше значения для империалистического государства), а иначе империалистическое государство не менять.

АЛГОРИТМ, ИМИТИРУЮЩИЙ ПОВЕДЕНИЕ СТАИ КРИЛЯ

Шаг 1. Задать параметры метода:

- количество особей криля в популяции NP;
- максимальная скорость движения криля $N_{\max}$;
- малое положительное число μ;
- максимальное число итераций $I_{\max}$;
- максимальная скорость передвижения к источнику пищи V_f;
- максимальная скорость диффузии криля $D_{\max}$.

Положить $I = 1$ (счетчик числа итераций).

Шаг 2. *Создание начальной популяции.* Генерировать начальную популяцию криля на множестве D, используя равномерный закон распределения: $x^1, \ldots, x^{NP}$.

Вычислить значения целевой функции $f(x^1), \ldots, f(x^{NP})$. Найти наилучшее и наихудшее решения x^{best}, x^{worst}.

Шаг 3. *Нахождение скорости, порождаемой остальными членами популяции.* Для каждого элемента популяции ($j = 1, \ldots, NP$) выполнить следующие шаги.

Шаг 3.1. Найти радиус окрестности около решения x^j: $d_{\varepsilon_j} = \dfrac{1}{5 \cdot NP} \sum\limits_{k=1}^{NP} d_{j,k}$,

где $d_{j,k} = \sqrt{\sum\limits_{i=1}^{n} \left(x_i^j - x_i^k \right)^2}$.

Определить число соседей N_j решения x^j из условия $d_{j,k} \le d_{\varepsilon_j}$.

Шаг 3.2. Вычислить (учесть влияние всех соседей):

$$\Delta \bar{x}^{j,k} = \frac{x^k - x^j}{d_{j,k} + \mu}, \; k = 1, \ldots, N_j;$$

$$\Delta \bar{f}^{j,k} = \frac{f(x^j) - f(x^k)}{f(x^{worst}) - f(x^{best})}, \; k = 1, \ldots, N_j;$$

$$\alpha_j^{local} = \sum_{k=1}^{N_j} \Delta \bar{f}^{j,k} \cdot \Delta \bar{x}^{j,k}.$$

Шаг 3.3. Вычислить (учесть влияние наилучшего элемента):

$$c^{best} = 2 \cdot \left(rand + \frac{I}{I_{\max}} \right), \; rand \sim U[0;1],$$

$$\Delta \bar{f}^{j,best} = \frac{f(x^j) - f(x^{best})}{f(x^{worst}) - f(x^{best})},$$

$$\Delta\overline{x}^{j,best} = \frac{x^{best} - x^{j}}{d_{j,best} + \mu},$$

$$\alpha_j^{target} = c^{best} \cdot \Delta\overline{f}^{j,best} \cdot \Delta\overline{x}^{j,best}.$$

Шаг 3.4. Найти $\alpha^j = \alpha_j^{local} + \alpha_j^{target}$.

Шаг 3.5. Определить первую составляющую новой скорости с учетом старой скорости:

$$N^{j,New} = N_{\max} \cdot \alpha^j + \omega \cdot N^{j,old},$$

где $\omega \sim U[0;1]$, $N^{j,old}$ – старая скорость, порождаемая остальными членами популяции (при $I = 1$ положить равной нулевому вектору).

Шаг 4. *Нахождение скорости, порождаемой движением к источнику пищи.* Для каждого элемента популяции ($j = 1,\ldots,NP$) выполнить следующие шаги.

Шаг 4.1. Найти положение источника пищи

$$x^{food} = \frac{\sum_{j=1}^{NP} \frac{x^j}{f(x^j)}}{\sum_{j=1}^{NP} \frac{1}{f(x^j)}}.$$

Шаг 4.2. Вычислить (учесть влияние источника пищи):

$$\Delta\overline{f}^{j,food} = \frac{f(x^j) - f(x^{food})}{f(x^{worst}) - f(x^{food})},$$

$$\Delta\overline{x}^{j,food} = \frac{x^{food} - x^{j}}{d_{j,food} + \mu},$$

$$c^{food} = 2 \cdot \left(1 - \frac{I}{I_{\max}}\right),$$

$$\beta_j^{food} = c^{food} \cdot \Delta\overline{f}^{j,food} \cdot \Delta\overline{x}^{j,food}.$$

Шаг 4.3. Вычислить (учесть влияние своего наилучшего результата):

$$\Delta\overline{f}^{j,jbest} = \frac{f(x^j) - f(x^{jbest})}{f(x^{worst}) - f(x^{jbest})},$$

где x^{jbest} – наилучшее положение j-го элемента популяции;

$$\Delta\overline{x}^{j,jbest} = \frac{x^{jbest} - x^{j}}{d_{j,jbest} + \mu};$$

$$\beta_j^{best} = \Delta\overline{f}^{j,jbest} \cdot \Delta\overline{x}^{j,jbest}.$$

Шаг 4.4. Найти $\beta^j = \beta_j^{food} + \beta_j^{best}$.

Шаг 4.5. Определить вторую составляющую новой скорости с учетом старой скорости:

$$F^{j,New} = V_f \cdot \beta^j + \omega_f \cdot F^{j,old},$$

где V_f – максимальная скорость передвижения к источнику пищи, $\omega_f \sim U[0;1]$, $F^{j,old}$ – старая скорость, порождаемая движением к источнику пищи (при $I = 1$ положить равной нулевому вектору).

Шаг 5. *Нахождение скорости, порождаемой случайными блужданиями.* Определить

$$D^j = D_{\max} \cdot \left(1 - \frac{I}{I_{\max}}\right) \cdot \delta,$$

где δ $-n$ -мерный вектор с компонентами $\delta_i \sim U[-1;1]$.

Шаг 6. *Нахождение результирующей скорости* (*питание*). Для всех $j = 1,\ldots,NP$ определить

$$V^j = N^{j,New} + F^{j,New} + D^j.$$

Шаг 7. *Определение новых положений элементов популяции* (*плавание*). Для всех $j = 1,\ldots,NP$ найти

$$x^{j,New} = x^{j,old} + V^j \cdot \Delta t,$$

где $\Delta t = c_t \cdot \sum_{i=1}^{n} (b_i - a_i)$, c_t – число на промежутке $[0;2]$.

Если $x_i^{j,New} \notin [a_i, b_i]$, то положить $x_i^{j,New} = a_i + \chi \cdot [b_i - a_i]$, $\chi \sim U[0;1]$.

Шаг 8. *Скрещивание* (*кроссовер*). Для всех $j = 1,\ldots,NP$ выполнить следующие шаги.

Шаг 8.1. Найти $Cr = 0,2 \cdot \overline{\Delta f}^{j,best}$.

Шаг 8.2. Определить

$$x_i^{j,Cr} = \begin{cases} x_i^{r,New}, & rand_i < Cr, \quad r \in \{1,\ldots,j-1,j+1,\ldots,NP\} \\ x_i^{j,New}, & rand_i \geq Cr, \end{cases}, \quad i = 1,\ldots,n,$$

где r – случайное целое число из множества $\{1,\ldots,j-1,j+1,\ldots,NP\}$, $rand_i \sim U[0;1]$.

В результате находятся решения $x^{1,Cr},\ldots,x^{NP,Cr}$.

Шаг 9. *Мутация*. Для всех $j = 1,\ldots,NP$ выполнить следующие шаги.

Шаг 9.1. Найти $Mu = 0,05 \cdot \overline{\Delta f}^{j,best}$.

Шаг 9.2. Определить

$$x_i^{j,Mu} = \begin{cases} x_i^{best} + v\left(x_i^{p,Cr} - x_i^{q,Cr}\right), & rand_i < Mu, \\ x_i^{j,Cr}, & rand_i \geq Mu, \end{cases}, i = 1,\ldots,n,$$

где $v \sim U[0;1]$, p, q – случайные, но не совпадающие друг с другом числа из множества $\{1,2,\ldots,j-1,j+1,\ldots,NP\}$.

Если $x_i^{j,Mu} \notin [a_i, b_i]$, то положить $x_i^{j,Mu} = a_i + \chi \cdot [b_i - a_i]$.

Шаг 9.3. Положить $x^j = x^{j,Mu}$, $j = 1,\ldots,NP$.

Шаг 10. *Определение наилучших положений элементов популяции.* Вычислить значения целевой функции. Найти наилучшее и наихудшее решения x^{best}, x^{worst}. Для каждого решения x^j найти наилучшее положение x^{jbest} за все прошедшие итерации.

Шаг 11. *Проверка выполнения условия окончания.*

Если $I < I_{\max}$, то положить $I = I + 1$ и перейти к шагу 3.

Если $I = I_{\max}$, процесс завершить. В качестве приближенного решения задачи выбрать x^{best}, $f(x^{best})$.

Замечание.

Рекомендуемые значения параметров: $N_{\max} = 0,01$; $V_f = 0,02$; $c_i = 0,2$; $NP = 40$; $D_{\max} \in [0,002; 0,01]$. Вместо применения равномерного на отрезке $[0;1]$ закона распределения при нахождении параметров ω_f, ω в начале процесса поиска их можно положить равными $0,9$, а затем с ростом числа итераций линейно уменьшать до $0,1$.

АЛГОРИТМ, ИМИТИРУЮЩИЙ ПОВЕДЕНИЕ СТАИ РЫБ

Шаг 1. Задать параметры метода:

- число рыб в популяции NP;
- максимальный вес рыбы W_{scale};
- пороговый вес рыбы thr;
- максимальное число итераций $ITER$;
- начальное и конечное значения шага индивидуального плавания $step_{ind,initial}$, $step_{ind,final}$;
- начальное и конечное значения шага коллективного плавания $step_{vol,initial}$, $step_{vol,final}$.

Положить $k = 0$ (счетчик числа итераций), $step^0_{ind} = step_{ind,initial}$, $step^0_{vol} = step_{vol,initial}$.

Шаг 2. *Генерация начальной популяции.*

Шаг 2.1. Используя равномерный закон распределения на множестве, сгенерировать начальную популяцию рыб

$$I_k = \{x^{j,0} = (x_1^{j,0}, x_2^{j,0}, \ldots, x_n^{j,0})^T, j = 1, \ldots, NP\} \subset D.$$

Шаг 2.2. Для каждой частицы в популяции вычислить соответствующее ей значение целевой функции: $f\left(x^{1,0}\right), \ldots, f\left(x^{NP,0}\right)$.

Шаг 2.3. Положить вес всех рыб стаи одинаковым:

$$W^{j,0} = \frac{W_{scale}}{2}, j = 1, \ldots, NP.$$

Шаг 3. *Операция индивидуального плавания.*

Шаг 3.1. Найти положение рыбы, движущейся в случайном направлении:

$$\tilde{x}^{j,k+1} = x^{j,k} + step^k_{ind} \cdot \delta, \quad j = 1, \ldots, NP,$$

где $\delta \sim R[-0,5; 0,5]$, $step^k_{ind}$ – индивидуальный шаг.

Если $\tilde{x}^{j,k+1} \notin D$, то положить $\tilde{x}^{j,k+1} = x^{j,k}$ (рыба не двигается).

Шаг 4. *Операция инстинктивно-коллективного плавания.*

Шаг 4.1. Найти индивидуальный сдвиг каждой рыбы в стае

$$\Delta x_{ind}^{j} = \tilde{x}^{j,k+1} - x^{j,k}, \quad j = 1,...,NP.$$

Шаг 4.2. Изменить положение каждой рыбы в стае:

$$\hat{x}^{j,k+1} = \tilde{x}^{j,k+1} + \frac{\sum_{m=1}^{NP} \Delta x_{ind}^{m} \cdot \left[f(\tilde{x}^{m,k+1}) - f(x^{m,k}) \right]}{\sum_{m=1}^{NP} \left[f(\tilde{x}^{m,k+1}) - f(x^{m,k}) \right]}, j = 1,...,NP.$$

Если $\hat{x}^{j,k+1} \notin D$, то положить $\hat{x}^{j,k+1} = \tilde{x}^{j,k+1}$ (рыба не двигается).

Шаг 5. *Операция питания.* Найти новый вес каждой рыбы:

$$W^{j,k+1} = W^{j,k} + \frac{f(\hat{x}^{j,k+1}) - f(x^{j,k})}{\max\limits_{m=1,...,NP} \left\{ \left| f(\hat{x}^{m,k+1}) - f(x^{m,k}) \right| \right\}}, \quad j = 1,...,NP.$$

Шаг 6. *Операция коллективно-волевого плавания.*

Шаг 6.1. Найти положение барицентра стаи:

$$Bari(k+1) = \frac{\sum_{j=1}^{NP} \hat{x}^{j,k+1} \cdot W^{j,k+1}}{\sum_{j=1}^{NP} W^{j,k+1}}.$$

Шаг 6.2. Если суммарный вес стаи увеличился, т.е. $\sum_{j=1}^{NP} W^{j,k+1} > \sum_{j=1}^{NP} W^{j,k}$, то найти новое положение каждой рыбы, двигающейся по направлению к барицентру:

$$x^{j,k+1} = \hat{x}^{j,k} - step_{vol}^{k} \cdot rand \cdot \frac{\left[\hat{x}^{j,k} - Bari(k+1) \right]}{\left\| \hat{x}^{j,k} - Bari(k+1) \right\|}, \quad j = 1,...,NP,$$

где $rand \sim R[0;1]$.

Если суммарный вес стаи не увеличился, т.е. $\sum_{j=1}^{NP} W^{j,k+1} \le \sum_{j=1}^{NP} W^{j,k}$, то найти новое положение каждой рыбы, двигающейся по направлению от барицентра:

$$x^{j,k+1} = \hat{x}^{j,k} + step_{vol}^{k} \cdot rand \cdot \frac{\left[\hat{x}^{j,k} - Bari(k+1) \right]}{\left\| \hat{x}^{j,k} - Bari(k+1) \right\|}, \quad j = 1,...,NP.$$

Если $x^{j,k+1} \notin D$, то положить $x^{j,k+1} = \hat{x}^{j,k+1}$ (рыба не двигается).

Шаг 7. *Операции селекции и размножения.*

Шаг 7.1. Из стаи выбрать рыб, достигнувших порогового веса thr, т.е. удовлетворяющих условию $W^{j,k+1} \ge thr$.

Шаг 7.2. Для каждой выбранной рыбы $x^{j,k+1}$ найти пару $x^{i,k+1}$ для размножения, которой соответствует максимальное отношение ее веса к расстоянию до $x^{j,k+1}$, т.е. удовлетворяющую условию $\max\limits_{\substack{m=1,\ldots,NP \\ m\neq j}} \dfrac{W^{m,k+1}}{\left\| x^{j,k+1} - x^{m,k+1} \right\|}$.

Шаг 7.3. Для каждой выбранной пары рыб с номерами i и j создать потомка с весом и положением:

$$W^{child,k+1} = \frac{W^{i,k+1} + W^{j,k+1}}{2}, \quad x^{child,k+1} = \frac{x^{i,k+1} + x^{j,k+1}}{2}.$$

Шаг 8. *Формирование новой стаи.*

Шаг 8.1. Ранжировать членов стаи, включая рыб-потомков, по весу.

Шаг 8.2. Удалить из стаи всех рыб с наименьшим весом, оставив NP рыб, т.е. сохранив размер стаи.

Шаг 9. *Проверка условия завершения процесса поиска.*

Если $k+1 = ITER$, то процесс поиска завершить. В качестве приближенного решения задачи (1.1) выбирается положение рыбы с наибольшим весом.

Если $(k+1) < ITER$, то вычислить

$$step_{ind}^{k+1} = step_{ind}^{k} - \frac{\left(step_{ind,ini} - step_{ind,final}\right)}{ITER},$$

$$step_{vol}^{k+1} = step_{vol}^{k} - \frac{\left(step_{vol,ini} - step_{vol,final}\right)}{ITER},$$

положить $k = k+1$ и перейти к шагу 3.

1.6.3. Программное обеспечение

На основе изложенных алгоритмов разработан программный комплекс поиска глобального экстремума функций [16]. Для его создания использовалась среда разработки Microsoft Visual Studio, язык программирования C#.

Описанный в данном разделе метод, имитирующий империалистическую конкуренцию, соответствует задаче поиска минимума целевой функции. Так как заданные тестовые функции имеют глобальный максимум, при разработке программного обеспечения этого алгоритма был заменен знак перед целевыми функциями на противоположный.

Возможности программного комплекса позволяют изучить алгоритмы методов, а также влияние параметров каждого метода на результат его работы. В список выбираемых функций включены стандартные многоэкстремальные тестовые функции двух переменных, для которых известно точное решение (табл. П.1).

На рис. 1.32, 1.34, 1.36 представлены главные окна методов, имитирующих империалистическую конкуренцию, поведение стай криля и стаи рыб соответственно, где пользователь может выбрать вид целевой функции, задать множество допустимых решений и параметры метода, просматривать результаты работы.

Разработанная программа предусматривает возможность анализа работы методов по шагам. На рис. 1.33, 1.35, 1.37 представлены окна пошаговой работы методов, где изображена обшая схема метода, отображаются результаты работы после выполнения каждого шага (график изменения наилучшего и среднего значений целевой

функции, а также графическое изображение популяции) и после завершения работы метода.

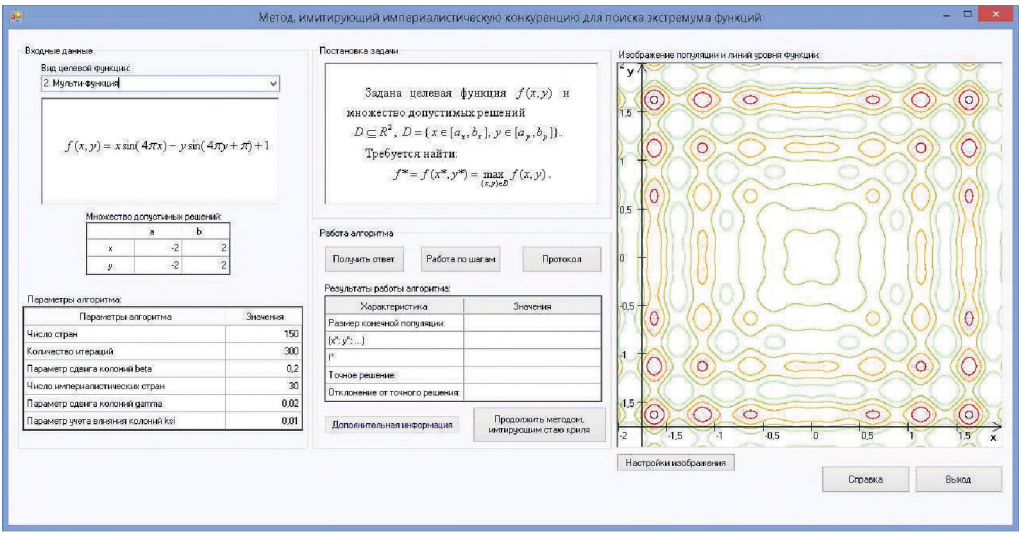


Рис. 1.32. Главное окно программы метода, имитирующего империалистическую конкуренцию

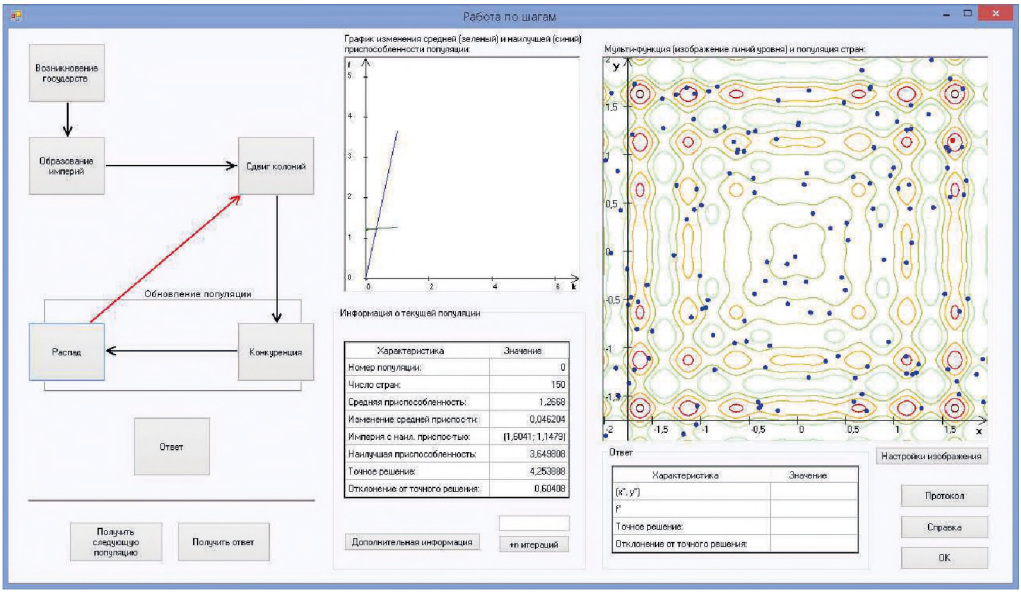


Рис. 1.33. Окно пошаговой работы метода, имитирующего империалистическую конкуренцию

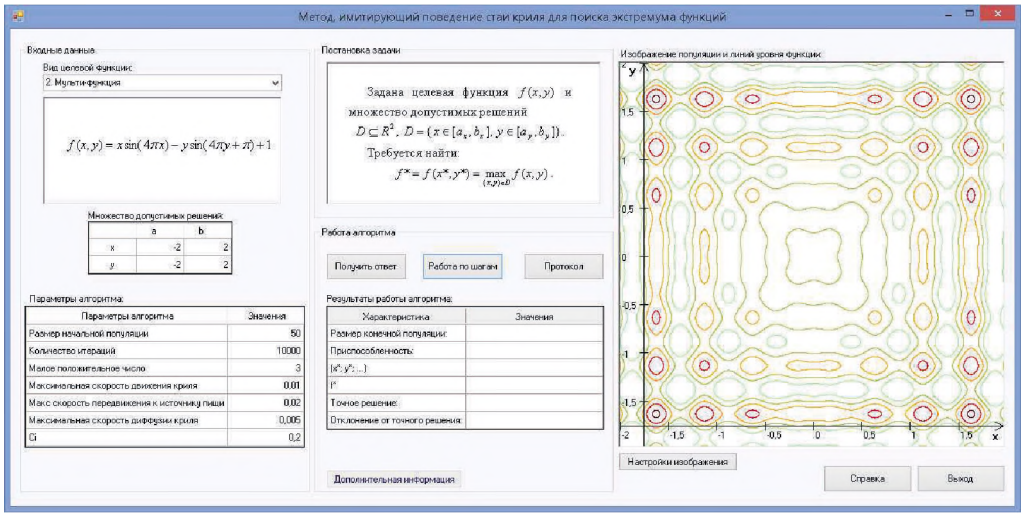


Рис. 1.34. Главное окно программы метода, имитирующего поведение стаи криля

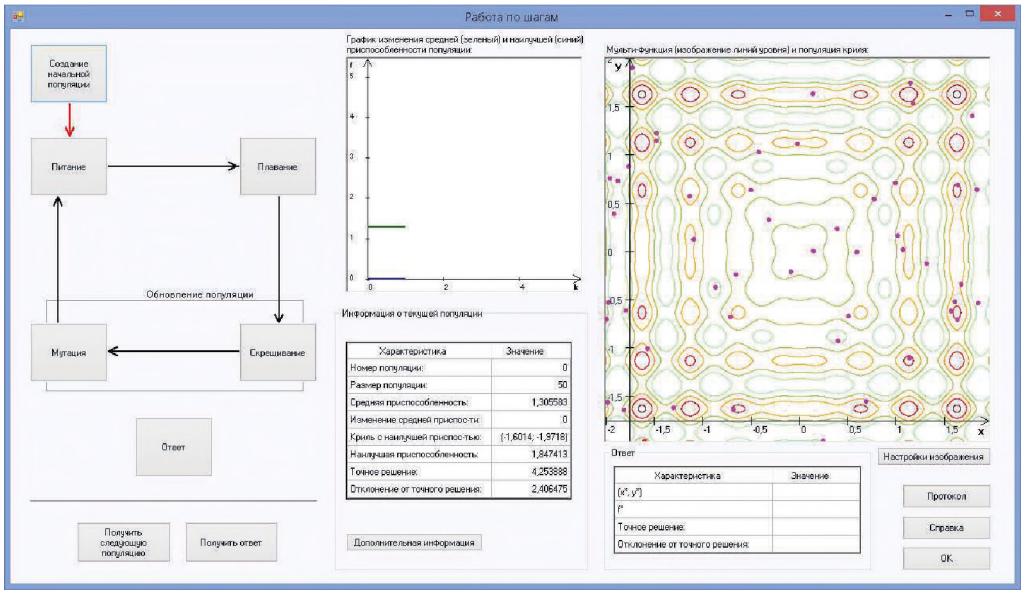


Рис. 1.35. Окно пошаговой работы метода, имитирующего поведение стаи криля

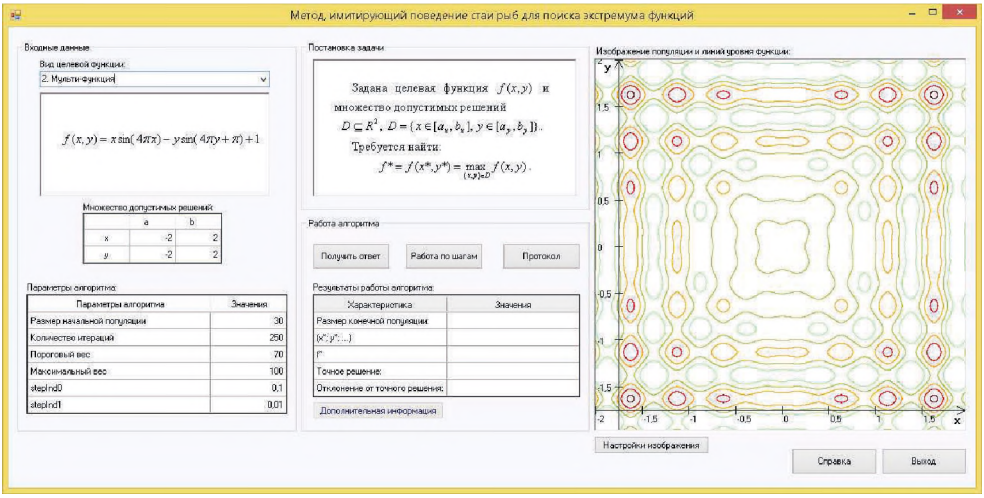


Рис. 1.36. Главное окно программы метода, имитирующего поведение стаи рыб

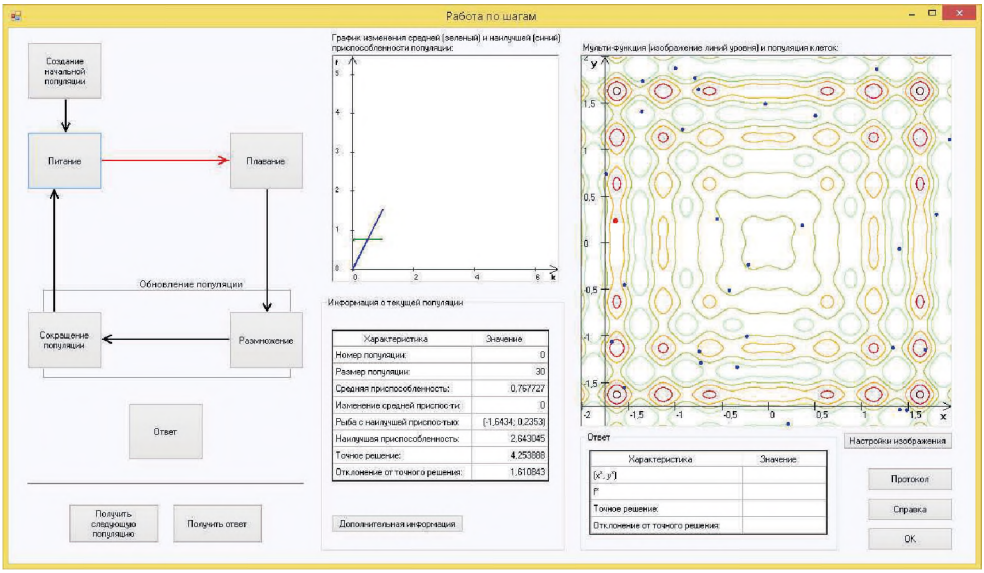


Рис. 1.37. Окно пошаговой работы метода, имитирующего поведение стаи рыб

1.6.4. Тестовые примеры

Пример 1.7. Найдем глобальный максимум корневой функции (табл. П.1) методом, имитирующим империалистическую конкуренцию. Зададим множество допустимых решений $x, y \in [-2; 2]$. Выберем следующие параметры алгоритма:

- размер популяции (число стран) $N_{pop} = 250$;
- число империалистических стран $N_{imp} = 25$;

- параметры сдвига колоний $\beta = 0,3,\ \gamma = 0,03$;
- параметр учета влияния колоний $\xi = 0,01$.

На рис. 1.38 представлена популяция на начальной ($k = 1$), промежуточных ($k = 100,\ k = 300$) и конечной ($k = 344$) итерациях.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (-0,470696; 0,837501)$;
- значение целевой функции $f(x^*, y^*) = 0,803788$;
- отклонение от точного решения $\Delta = 0,196212$.

График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции представлен на рис. 1.39.

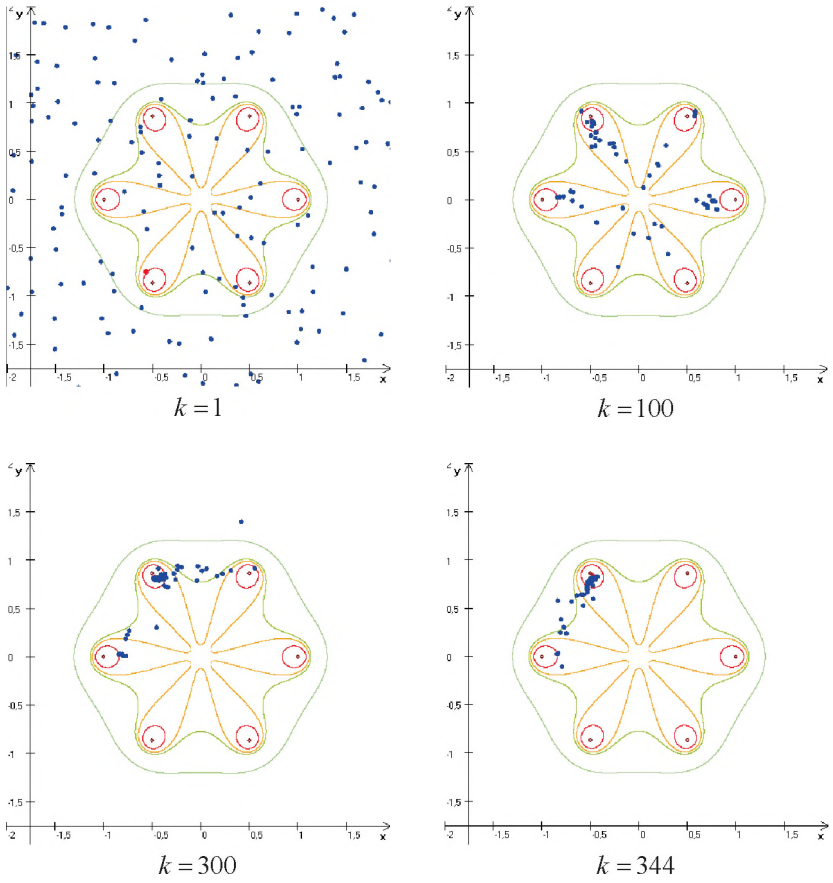


Рис. 1.38. Начальная, промежуточные и конечная итерации

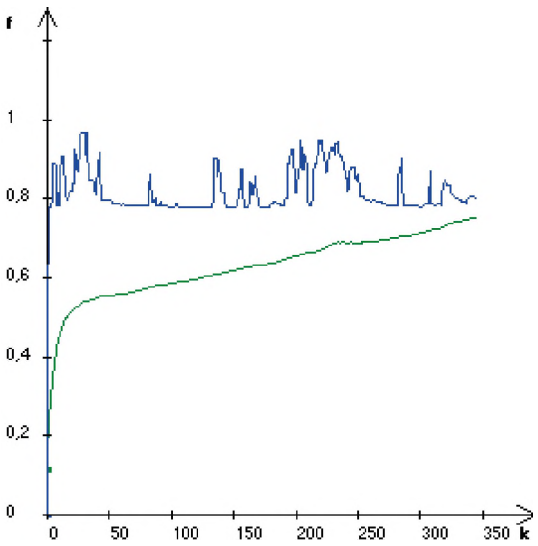


Рис. 1.39. График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции

Пример 1.8. Найдем глобальный максимум функции Экли (табл. П.1) методом, имитирующим империалистическую конкуренцию. Зададим множество допустимых решений $x, y \in [-10;10]$.

Выберем следующие параметры алгоритма:

- размер популяции (число стран) $N_{pop} = 150$;
- число империалистических стран $N_{imp} = 15$;
- параметры сдвига колоний $\beta = 0,4,\ \gamma = 0,04$;
- параметр учета влияния колоний $\xi = 0,01$.

На рис. 1.40 представлена популяция на начальной ($k = 1$), промежуточных ($k = 50$, $k = 160$) и конечной ($k = 296$) итерациях. Красным цветом обозначено наилучшее решение в популяции на данной итерации.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (0,06975; -0,013195)$;
- значение целевой функции $f(x^*, y^*) = 19,669538$;
- отклонение от точного решения $\Delta = 0,330462$.

График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции представлен на рис. 1.41.

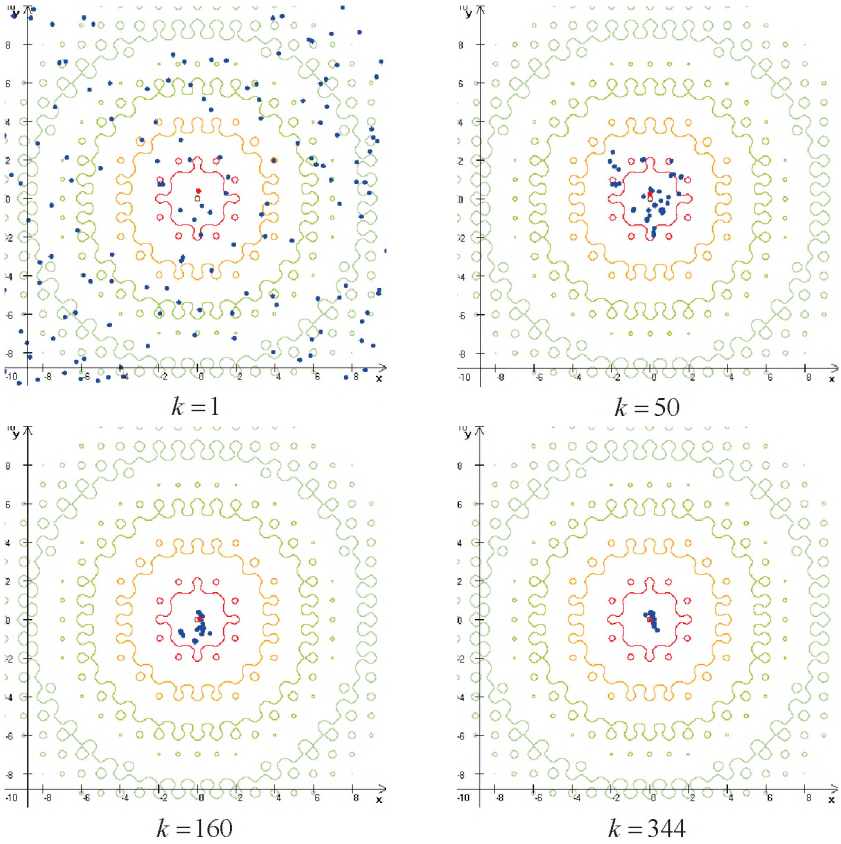


Рис. 1.40. Начальная, промежуточные и конечная популяции

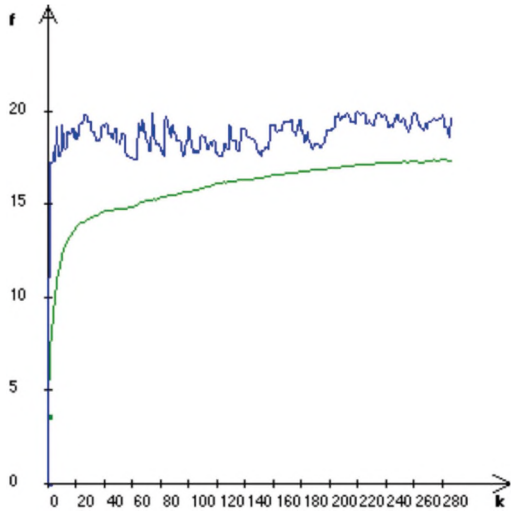


Рис. 1.41. График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции

Пример 1.9. Найдем глобальный максимум синусоидальной функции Швефеля (табл. П.1) методом, имитирующим поведение стаи криля. Зададим множество допустимых решений $x, y \in [-500; 500]$. Выберем следующие параметры алгоритма:

- число элементов в популяции $NP = 50$;
- максимальное число итераций $I_{\max} = 10000$;
- малое положительное число $\mu = 3$;
- максимальная скорость движения криля $N_{\max} = 0,01$;
- максимальная скорость передвижения к источнику пищи $V_f = 0,02$;
- максимальная скорость диффузии криля $D_{\max} = 0,005$.

На рис. 1.42 представлена популяция на начальной ($k = 1$), промежуточных ($k = 100, k = 3000$) и конечной ($k = 10000$) итерациях.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (420,969; 420,969)$;
- значение целевой функции $f(x^*, y^*) = 837,9658$;
- отклонение от точного решения $\Delta = 0$.

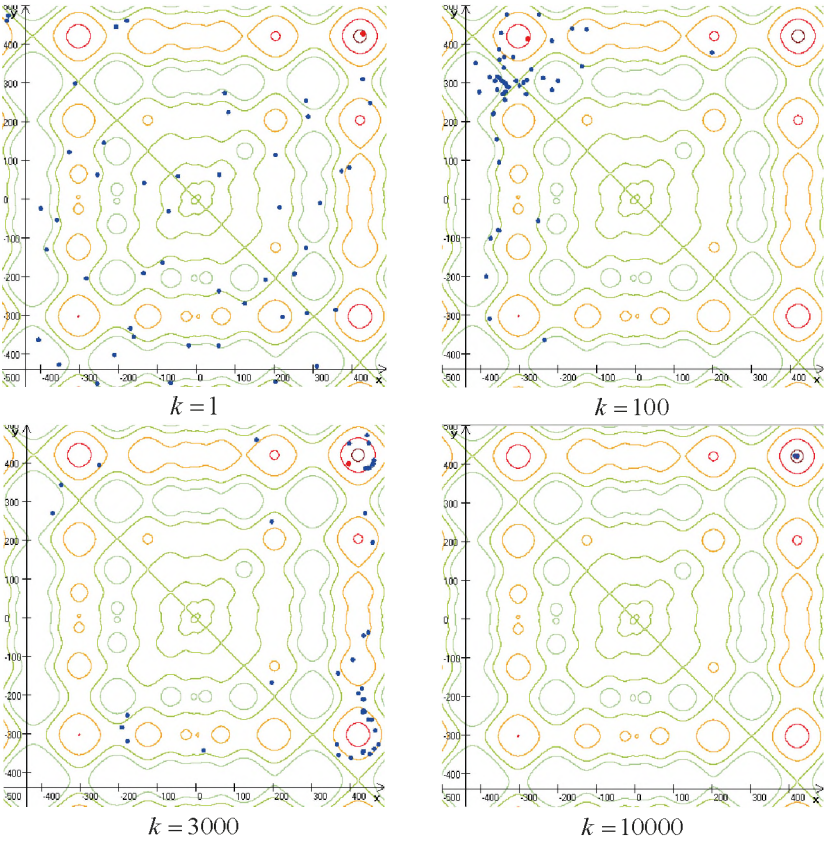


Рис. 1.42. Начальная, промежуточные и конечная итерации

График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции представлен на рис. 1.43.

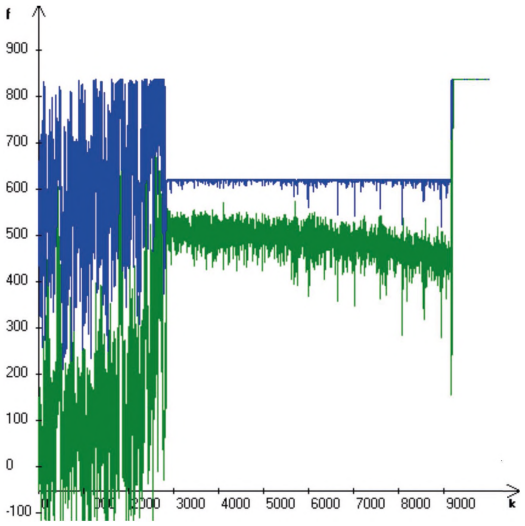


Рис. 1.43. График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции

Пример 1.10. Найдем глобальный максимум мульти-функции (табл. П.1) методом, имитирующим поведение стаи криля. Зададим множество допустимых решений $x, y \in [-2;2]$. Выберем следующие параметры алгоритма:

- число элементов в популяции $NP = 40$;
- максимальное число итераций $I_{\max} = 4000$;
- малое положительное число $\mu = 3$;
- максимальная скорость движения криля $N_{\max} = 0,01$;
- максимальная скорость передвижения к источнику пищи $V_f = 0,02$;
- максимальная скорость диффузии криля $D_{\max} = 0,005$.

На рис. 1.44 представлена популяция на начальной ($k = 1$), промежуточных ($k = 440$, $k = 522$) и конечной ($k = 4000$) итерациях.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (1,629; 1,629)$;
- значение целевой функции $f(x^*, y^*) = 4,25388$;
- отклонение от точного решения $\Delta = 0$.

График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции представлен на рис. 1.45.

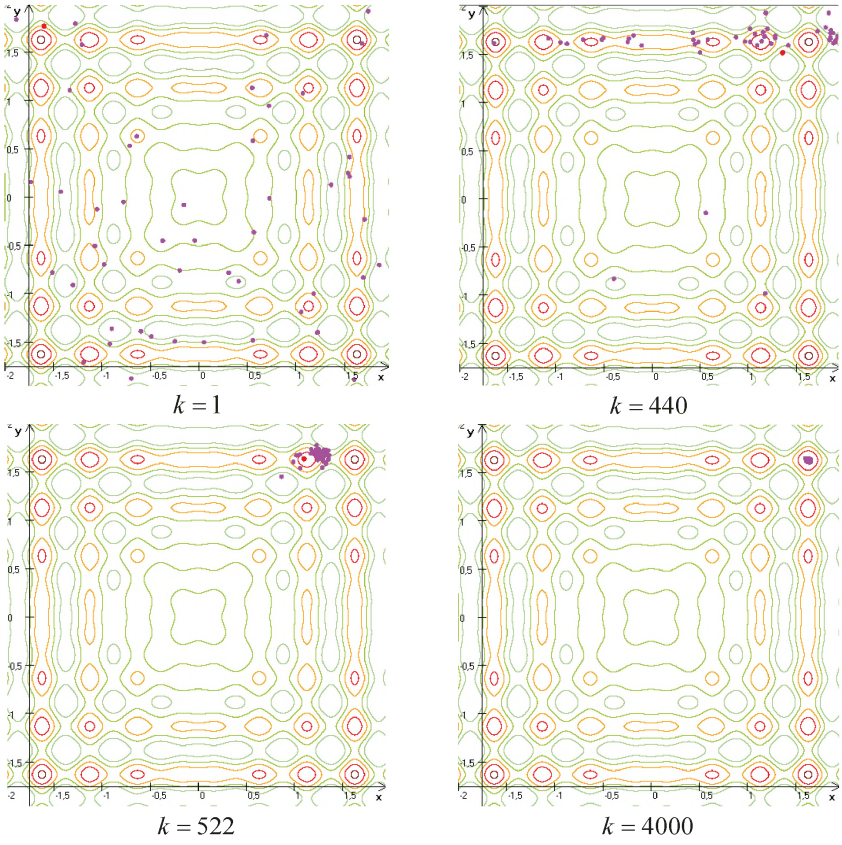


Рис. 1.44. Начальная, промежуточные и конечная популяции

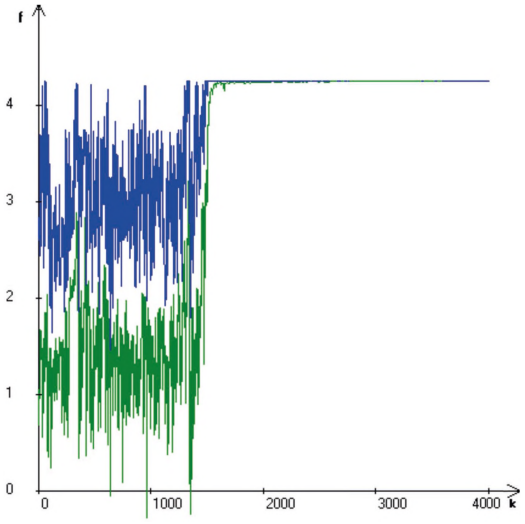


Рис. 1.45. График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции

Пример 1.11. Найдем глобальный максимум функции Шаффера (табл. П.1) методом, имитирующим поведение стаи рыб. Зададим множество допустимых решений $x, y \in [-10;10]$. Выберем следующие параметры алгоритма:

- число элементов в популяции $NP = 100$;
- максимальное число итераций $ITER = 1000$;
- пороговый вес $thr = 4500$;
- максимальный вес $W_{scale} = 5000$;
- начальное и конечное значения шага индивидуального плавания:
  $step_{ind,initial} = step_{ind0} = 0,1$, $step_{ind,final} = step_{ind1} = 0,01$;
- начальное и конечное значения шага коллективного плавания:
  $step_{vol,initial} = 0,001$, $step_{vol,final} = 0,0001$.

На рис. 1.46 представлена популяция на начальной ($k = 1$), промежуточных ($k = 50$, $k = 200$) и конечной ($k = 1000$) итерациях.

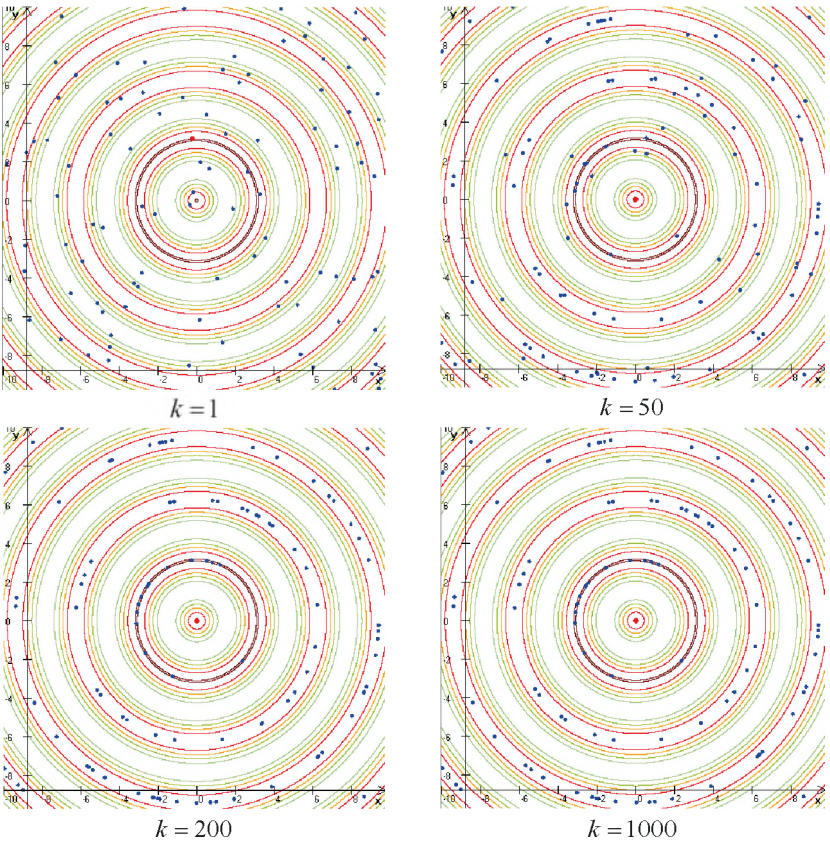


Рис. 1.46. Начальная, промежуточные и конечная итерации

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (0;0)$;

- значение целевой функции $f(x^*, y^*) = 1$;
- отклонение от точного решения $\Delta = 0$.

График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции представлен на рис. 1.47.

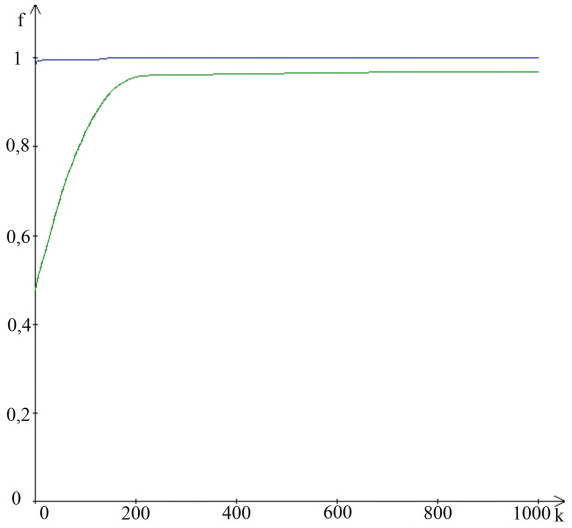


Рис. 1.47. График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции

Пример 1.12. Найдем глобальный максимум функции Швефеля (табл. П.1) методом, имитирующим поведение стаи рыб. Зададим множество допустимых решений $x, y \in [-500; 500]$. Выберем следующие параметры алгоритма:

- число элементов в популяции $NP = 30$;
- максимальное число итераций $ITER = 1000$;
- пороговый вес $thr = 4500$;
- максимальный вес $W_{scale} = 5000$;
- начальное и конечное значения шага индивидуального плавания: $step_{ind,initial} = step_{ind0} = 5,\ step_{ind,final} = step_{ind1} = 0,5$;
- начальное и конечное значения шага коллективного плавания: $step_{vol,initial} = 0,001,\ step_{vol,final} = 0,0001$.

На рис. 1.48 представлена популяция на начальной ($k = 1$), промежуточных ($k = 50$, $k = 100$) и конечной ($k = 1000$) итерациях.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (420,971; 420,97)$;
- значение целевой функции $f(x^*, y^*) = 837,965774$;
- отклонение от точного решения $\Delta = 0,000026$.

График изменения среднего (зеленый) и наилучшего значений (синий) целевой функции представлен на рис. 1.49.

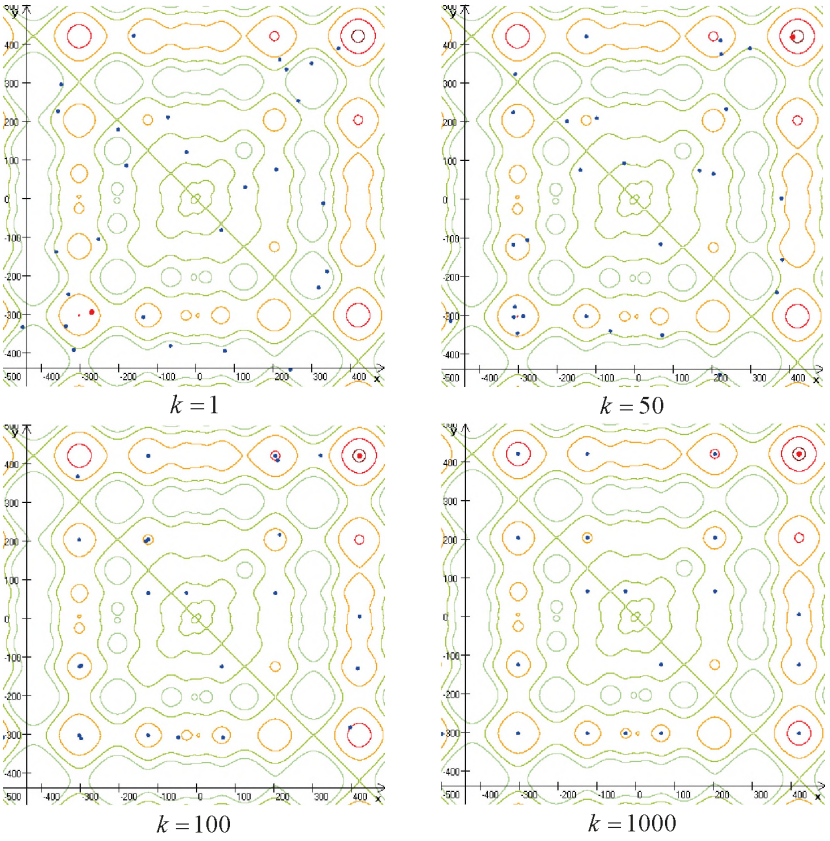


Рис. 1.48. Начальная, промежуточные и конечная популяции

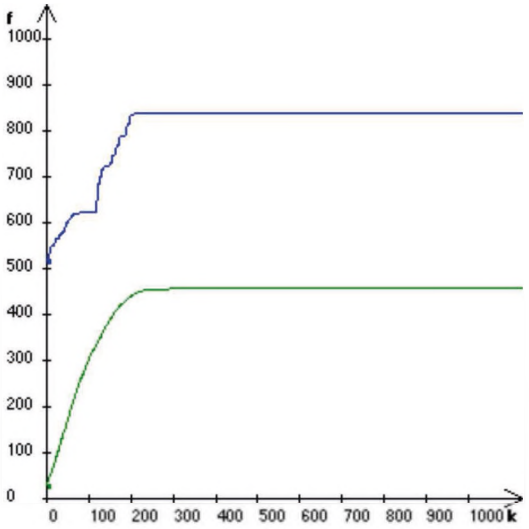


Рис. 1.49. График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции

Пример 1.13. Найдем глобальный максимум функции Розенброка (табл. П.1) последовательно-параллельным гибридным мультиагентным методом. Зададим множество допустимых решений $x, y \in [-500; 500]$. Выберем следующие параметры алгоритма:

- размер популяции $NP = 150$;
- максимальное число проходов $P_{\max} = 5$;
- пороговое значение $thr = 0,8$ для выбора метода оптимизации;
- число последовательных итераций контроля эффективности метода $I_{contr} = 66$;
- малое положительное число $\varepsilon_{eff} = 0,005$ для досрочного завершения прохода;
- число наилучших решений в популяции $N_{best} = 5$;
- число империалистических стран $N_{imp} = 15$;
- параметры сдвига колоний $\beta = 0,4,\ \gamma = 0,04$;
- параметр учета влияния колоний $\xi = 0,01$.

Параметры метода, имитирующего поведение стаи рыб: $ITER = 100,\ thr = 70,\ W_{scale} = 100,$ $step_{ind,initial} = step_{ind0} = 0,1,\ step_{ind,final} = step_{ind1} = 0,01,\ step_{vol,initial} = 0,001,\ step_{vol,final} = 0,0001.$

Параметры метода, имитирующего поведение стаи криля: $I_{\max} = 100$, $\mu = 0,3$, $N_{\max} = 0,001$, $V_f = 0,002$, $D_{\max} = 0,0005$.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (1,049; 1,096)$;
- значение целевой функции $f(x^*, y^*) = -0,003918$;
- отклонение от точного решения $\Delta = 0,003918$.

В ходе поиска решения последовательно-параллельным гибридным мультиагентным методом были получены популяции агентов, изображенные на рисунке 1.50. Слева представлена популяция, полученная методом, имитирующим империалистическую конкуренцию (см. шаг 2 алгоритма), справа – конечная популяция. График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции представлен на рис. 1.51.

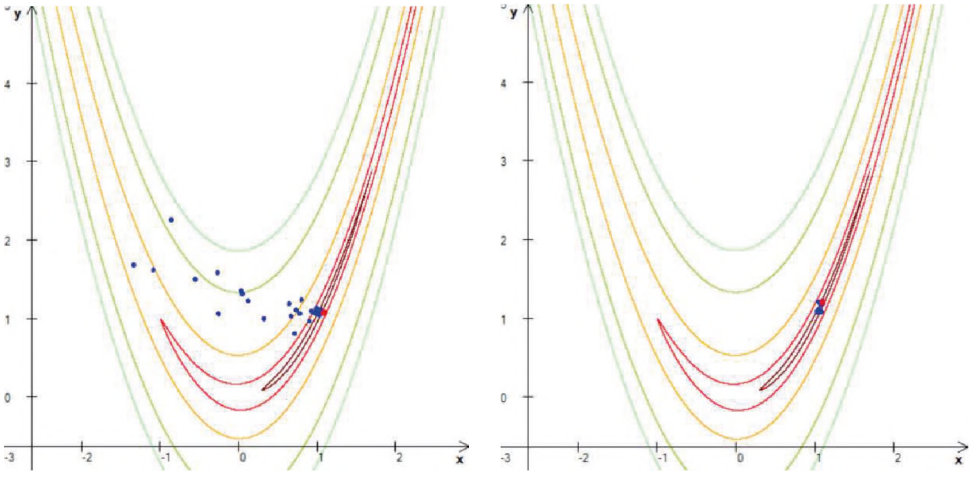


Рис. 1.50. Начальная и конечная популяции

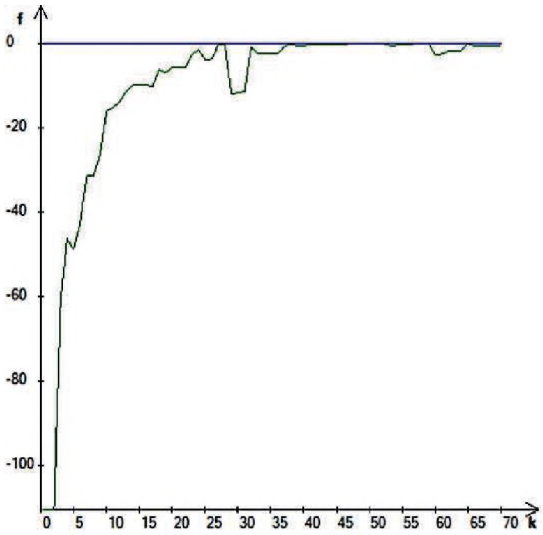


Рис. 1.51. График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции

Пример 1.14. Найдем глобальный максимум функции Растригина (табл. П.1) последовательно-параллельным гибридным мультиагентным методом. Зададим множество допустимых решений $x, y \in [-500; 500]$. Выберем следующие параметры алгоритма:

- размер популяции $NP = 150$;
- максимальное число проходов $P_{\max} = 5$;
- пороговое значение $thr = 0,8$ для выбора метода оптимизации;
- число последовательных итераций контроля эффективности метода $I_{contr} = 66$;
- малое положительное число $\varepsilon_{eff} = 0,005$ для досрочного завершения прохода;
- число наилучших решений в популяции $N_{best} = 5$;
- число империалистических стран $N_{imp} = 15$;
- параметры сдвига колоний $\beta = 0,4$, $\gamma = 0,04$;
- параметр учета влияния колоний $\xi = 0,01$.

Параметры метода, имитирующего поведение стаи рыб: $ITER = 100$, $thr = 70$, $W_{scale} = 100$, $step_{ind,initial} = step_{ind0} = 0,1$, $step_{ind,final} = step_{ind1} = 0,01$, $step_{vol,initial} = 0,001$, $step_{vol,final} = 0,0001$. Параметры метода, имитирующего поведение стаи криля: $I_{\max} = 100$, $\mu = 0,3$, $N_{\max} = 0,001$, $V_f = 0,002$, $D_{\max} = 0,0005$.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (0; 0,001)$;
- значение целевой функции $f(x^*, y^*) = 19,999985$;
- отклонение от точного решения $\Delta = 0,000015$.

В ходе поиска решения последовательно-параллельным гибридным мультиагентным методом были получены популяции агентов, изображенные на рис. 1.52. Слева представлена популяция, полученная методом, имитирующим империалистическую кон-

куренцию (см. шаг 2 алгоритма), справа – конечная популяция. График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции представлен на рис. 1.53.

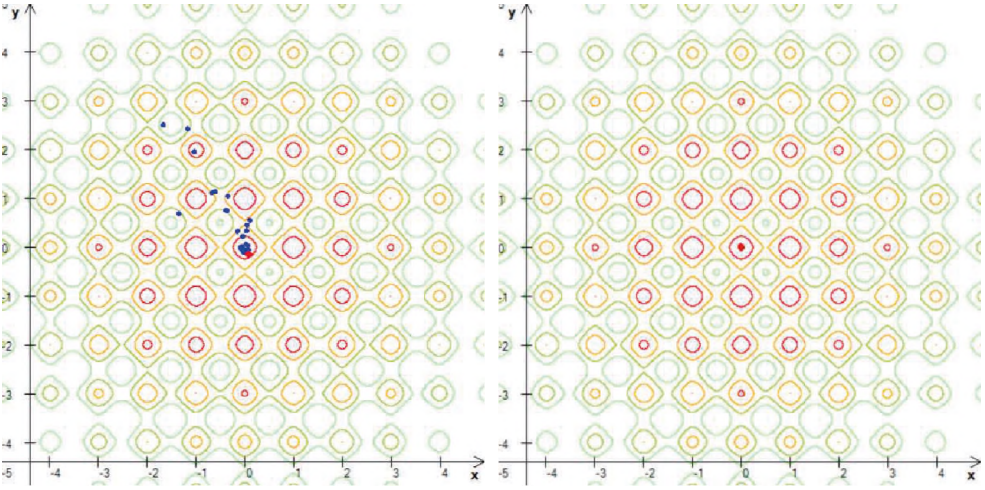


Рис. 1.52. Начальная и конечная популяции

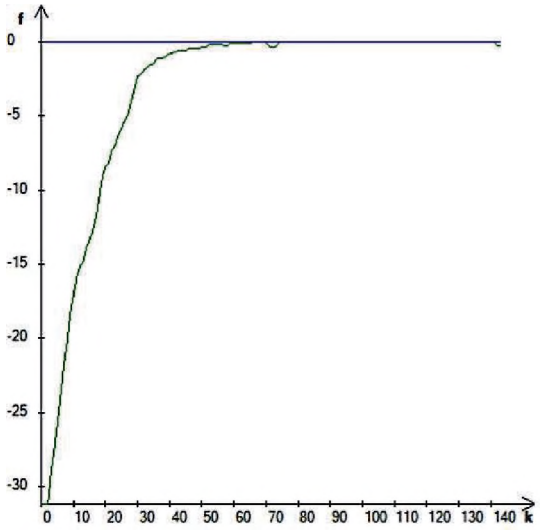


Рис. 1.53. График изменения среднего (зеленый) и наилучшего (синий) значений целевой функции

1.6.5. Анализ эффективности метода

Исследуемые методы применялись к некоторым функциям из набора тестовых функций (табл. П.1). Для каждой функции проводились серии из 100 решений одной и той же задачи с одними и теми же значениями параметров. Для полученной выборки $\{f^1, f^2, ..., f^{100}\}$ вычислялись среднее значение отклонения полученного решения

от точного $\overline{\Delta f} = \frac{1}{100}\sum_{i=1}^{100}\Delta f_i$, где $\Delta f_i = \left| f(x^*) - f^i \right|$; наименьшее значение отклонения $\Delta f_{best} = \min_i \Delta f_i$; среднеквадратическое отклонение $\overline{\sigma}_f = \sqrt{\overline{S}_{100}}$, где $\overline{S}_{100} = \frac{1}{100}\sum_{i=1}^{100}(\Delta f^i - \overline{\Delta f})^2$; количество успехов $n_{\text{усп}}$ (успехом считалось попадание лучшей точки в ε-окрестность точного решения, $\varepsilon = \frac{\max_{i=1,\ldots,n}\left|b_i - a_i\right|}{1000}$).

АНАЛИЗ РАБОТЫ МЕТОДА, ИМИТИРУЮЩЕГО ИМПЕРИАЛИСТИЧЕСКУЮ КОНКУРЕНЦИЮ, ПРИ РАЗЛИЧНЫХ ЗНАЧЕНИЯХ ПАРАМЕТРОВ

Результаты, полученные для каждой функции, представлены в табл. 1.13–1.15.

Таблица 1.13. Влияние параметров метода. Корневая функция

Параметры метода					$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{\text{усп}}$
N_{pop}	N_{imp}	β	γ	ξ				
100	15	0,2	0,02	0,01	0,294286	0,077473	0,09615	0
150	15	0,2	0,02	0,01	0,260119	0,035685	0,089547	0
100	25	0,2	0,02	0,01	0,301826	0,08401	0,091864	0
100	30	0,2	0,02	0,01	0,291763	0,054992	0,107983	0
100	30	0,09	0,02	0,01	0,272546	0,059794	0,094103	0
100	30	0,3	0,02	0,01	0,30937	0,065697	0,095413	0
100	30	0,3	0,03	0,01	0,298293	0,028702	0,101567	0
100	30	0,3	0,009	0,01	0,297539	0,098064	0,09903	0
100	30	0,09	0,02	0,02	0,284595	0,024862	0,094496	0
100	30	0,09	0,02	0,009	0,282375	0,002646	0,103811	1

Таблица 1.14. Влияние параметров метода. Мульти-функция

Параметры метода					$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{\text{усп}}$
N_{pop}	N_{imp}	β	γ	ξ				
100	15	0,2	0,02	0,01	0,662858	0,009067	0,34873	1
100	30	0,2	0,02	0,01	0,642687	0,056861	0,340352	0
100	30	0,6	0,02	0,01	0,707634	0,000541	0,356575	2
100	30	0,4	0,02	0,01	0,680159	0,021344	0,384152	0
100	30	0,6	0,009	0,01	0,700613	0,048248	0,359832	0
100	30	0,6	0,04	0,01	0,660325	0,001819	0,392237	1
100	30	0,6	0,06	0,01	0,677713	0,013398	0,341175	0
100	30	0,6	0,03	0,01	0,70207	0,006291	0,338476	0
100	30	0,6	0,04	0,008	0,695131	0,006853	0,320425	2
100	30	0,6	0,04	0,03	0,70769	0,021048	0,394252	0

Таблица 1.15. Влияние параметров метода. Функция Розенброка

Параметры метода					$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
N_{pop}	N_{imp}	β	γ	ξ				
150	15	0,4	0,04	0,01	0,963428	0,01208	1,290814	0
150	30	0,4	0,04	0,01	0,685492	0,006466	0,782858	3
150	40	0,4	0,04	0,01	0,836565	0,002387	0,906077	2
150	30	0,5	0,04	0,01	0,721781	0,011081	0,724213	0
150	30	0,7	0,04	0,01	1,045295	0,000217	1,151233	2
150	30	0,7	0,07	0,01	0,888599	0,001965	0,930612	1
150	30	0,7	0,01	0,01	0,831103	0,014354	0,934294	0
150	30	0,7	0,07	0,05	1,048038	0,009032	1,259677	1
150	30	0,7	0,07	0,009	1,375309	0,026596	1,3426	0
150	30	0,7	0,07	0,02	0,809717	0,004221	0,884309	2

Анализ работы метода, имитирующего империалистическую конкуренцию, показал, что он успешно находит область глобального экстремума, но не всегда попадает в точку глобального экстремума с достаточной точностью. Поэтому данный алгоритм рекомендуется применять для поиска хорошего начального приближения, а для уточнения решения применять другие алгоритмы, например, метод, имитирующий поведение стаи криля (см. разд. 1.6.2). Тем не менее, в серии из 100 запусков иногда удается найти глобальный минимум функций с достаточной точностью. При выборе функций с обширной областью допустимых решений близкий к точному решению результат получается только при достаточно большом размере популяции N_{pop}. При работе с функшией Розенброка метод находит область, в которой находится глобальный минимум, но не всегда успевает найти решение с высокой точностью. При выборе корневой функции или мульти-функции популяция стран находит область с одним из глобальных экстремумов и точку минимума, но с более низкой точностью.

АНАЛИЗ РАБОТЫ МЕТОДА, ИМИТИРУЮЩЕГО ПОВЕДЕНИЕ СТАИ КРИЛЯ, ПРИ РАЗЛИЧНЫХ ЗНАЧЕНИЯХ ПАРАМЕТРОВ

Результаты, полученные для каждой функции, представлены в табл. 1.16–1.18.

Таблица 1.16. Влияние параметров метода. Корневая функция

Параметры метода							$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	$I_{\max}$	$N_{\max}$	V_f	$D_{\max}$	μ	c_t				
40	10000	0,01	0,02	0,005	3	0,2	0,000149	0,00002	0,000097	100
40	2000	0,01	0,02	0,005	1	0,2	0,000416	0,000028	0,000196	100
40	2000	0,03	0,02	0,005	1	0,2	0,000524	0,000088	0,000253	100
40	2000	0,01	0,04	0,005	1	0,2	0,000741	0,000068	0,000399	100
40	2000	0,008	0,02	0,005	1	0,2	0,000804	0,000081	0,000492	100
40	2000	0,01	0,02	0,01	3	0,2	0,001618	0,000014	0,001098	100
40	2000	0,03	0,02	0,001	3	0,2	0,000154	0,000007	0,000077	100
40	2000	0,01	0,02	0,001	3	0,4	0,000233	0,000028	0,000115	100
40	10000	0,01	0,02	0,005	3	0,009	0,000011	0,000001	0,000008	100
20	5000	0,01	0,02	0,005	3	0,009	0,000023	0,000004	0,000016	100

Таблица 1.17. Влияние параметров метода. Мульти-функция

Параметры метода							$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	I_{max}	N_{max}	V_f	D_{max}	μ	c_t				
40	10000	0,01	0,02	0,005	0,5	0,2	0,021226	0	0,130833	97
40	10000	0,01	0,01	0,002	0,5	0,1	0,483446	0	0,556978	45
40	1000	0,01	0,01	0,002	0,5	0,1	0,42701	0	0,487665	47
40	1000	0,01	0,01	0,002	0,5	0,4	0,773316	0	0,870252	32
40	1000	0,03	0,02	0,005	0,5	0,2	0,826618	0	0,851022	29
40	10000	0,01	0,02	0,005	1	0,2	0,089205	0	0,290853	88
40	1000	0,01	0,02	0,008	0,5	0,2	0,683476	0	0,949103	39
40	1000	0,01	0,04	0,005	0,5	0,2	1,069537	0	1,166232	13
40	1000	0,01	0,01	0,005	0,5	0,2	0,511493	0	0,569006	42
50	10000	0,01	0,02	0,005	0,5	0,2	0,243973	0	0,457709	67

Таблица 1.18. Влияние параметров метода. Функция Экли

Параметры метода							$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	I_{max}	N_{max}	V_f	D_{max}	μ	c_t				
40	1000	0,01	0,02	0,005	0,5	0,2	0,00018	0,000029	0,000083	100
40	1000	0,01	0,02	0,005	1	0,2	0,000397	0,000044	0,000174	100
40	1000	0,01	0,02	0,005	0,1	0,2	0,00317	0,000042	0,023655	98
40	1000	0,02	0,02	0,005	0,5	0,2	0,000125	0,000013	0,00006	100
40	1000	0,01	0,03	0,005	0,5	0,2	0,000169	0,000029	0,000077	100
40	1000	0,01	0,01	0,005	0,5	0,2	0,000212	0,000011	0,000097	100
40	1000	0,01	0,01	0,007	0,5	0,2	0,000269	0,000061	0,000129	100
40	1000	0,01	0,01	0,003	0,5	0,2	0,000115	0,000019	0,000051	100
40	1000	0,01	0,01	0,003	0,5	0,4	0,000151	0,000002	0,000074	100
40	1000	0,01	0,01	0,003	0,5	0,1	0,000193	0,000023	0,000083	100

Анализ работы метода, имитирующего поведение стаи криля, показал, что с большинством тестовых функций метод справляется весьма успешно. В серии из 100 запусков удается найти глобальный максимум функций с большой точностью или вообще точное решение.

При выборе корневой функции и функции Экли метод, имитирующий поведение стаи криля, находит решение при различных сочетаниях значений параметров.

Метод не всегда может найти глобальный максимум мульти-функции, сходясь к одному из ее локальных максимумов.

При выборе функции Швефеля метод, имитирующий поведение стаи криля, успешно находит точное решение за достаточно большое число итераций. При ограничении сверху вычислительных затрат метод сходится к локальному максимуму и не успевает достигнуть глобального экстремума.

АНАЛИЗ РАБОТЫ МЕТОДА, ИМИТИРУЮЩЕГО ПОВЕДЕНИЕ СТАИ РЫБ, ПРИ РАЗЛИЧНЫХ ЗНАЧЕНИЯХ ПАРАМЕТРОВ

Результаты, полученные при следующих начальных и конечных значениях шагов коллективного и индивидуального плавания: $step_{vol\_initial} = 0,001$, $step_{vol\_final} = 0,0001$,

$step_{ind,initial} = step_{ind0}$, $step_{ind,final} = step_{ind1}$ для каждой функции, представлены в табл. 1.19–1.21.

Таблица 1.19. Влияние параметров метода. Корневая функция

Параметры метода						$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	*ITER*	*W*	W_{scale}	$step_{ind0}$	$step_{ind1}$				
30	10000	4500	5000	0,1	0,01	0,001215	0,000016	0,00069	100
20	10000	4500	5000	0,1	0,01	0,001055	0,000012	0,000572	100
10	5000	4500	5000	0,1	0,01	0,012728	0,000056	0,094832	99
20	5000	4000	5000	0,1	0,01	0,004717	0,000008	0,023464	99
20	5000	4000	4500	0,1	0,01	0,002062	0,000009	0,001096	100
20	5000	200	250	0,1	0,01	0,001699	0,000078	0,000932	100
20	5000	4500	5000	0,08	0,01	0,001419	0,000096	0,000703	100
20	10000	4500	5000	0,3	0,01	0,002643	0,000089	0,001545	100
20	10000	4500	5000	0,1	0,008	0,001632	0,000117	0,00082	100
20	10000	4500	5000	0,1	0,03	0,002507	0,000009	0,001283	100

Таблица 1.20. Влияние параметров метода. Мульти-функция

Параметры метода						$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	*ITER*	*W*	W_{scale}	$step_{ind0}$	$step_{ind1}$				
30	10000	4500	5000	0,1	0,01	0,789304	0	0,744677	12
50	10000	4500	5000	0,1	0,01	0,750076	0	0,798666	32
50	10000	4000	5000	0,1	0,01	0,940749	0	0,800207	22
50	10000	4700	5000	0,1	0,01	0,912874	0	0,650055	14
50	10000	4500	4800	0,1	0,01	0,797011	0	0,720034	26
50	10000	4500	5200	0,1	0,01	0,965895	0	0,739039	19
50	10000	4500	5000	0,08	0,01	0,539011	0	0,640241	34
50	10000	4500	5000	0,2	0,01	0,836956	0	0,656606	19
50	10000	4500	5000	0,1	0,009	0,882147	0	0,763311	19
50	10000	4500	5000	0,1	0,02	0,875453	0,000001	0,661915	19

Таблица 1.21. Влияние параметров метода. Функция Розенброка

Параметры метода						$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	*ITER*	*W*	W_{scale}	$step_{ind0}$	$step_{ind1}$				
30	10000	4500	5000	0,1	0,01	0	0	0	100
40	10000	4500	5000	0,1	0,01	0	0	0	100
40	10000	4000	5000	0,1	0,01	0	0	0	100
40	10000	4800	5000	0,1	0,01	0	0	0	100
40	10000	4500	4800	0,1	0,01	0	0	0	100
40	10000	4500	5200	0,1	0,01	0,0069	0	0,068654	99
40	10000	4500	5000	0,2	0,01	0	0	0	100
40	10000	4500	5000	0,09	0,01	18,010518	0	179,202387	99
40	10000	4500	5000	0,1	0,02	0,087777	0	0,785772	98
40	10000	4500	5000	0,1	0,009	0	0	0	100

Анализ работы метода, имитирующего поведение стаи рыб, показал, что с большинством тестовых функций метод справляется успешно. В серии из 100 запусков удается найти глобальный максимум функций с достаточно высокой точностью или вообще точное решение.

При работе с функцией Розенброка метод почти всегда успевает сойтись к точке глобального максимума и дает точный результат при использовании достаточно большого числа итераций *ITER*. Метод, имитирующий поведение стаи рыб, не всегда находит глобальный максимум мульти-функции и часто «застревает» в локальном максимуме, не успевая сойтись к точке глобального максимума. При максимизации корневой функции метод показывает хорошие результаты и успевает сойтись к одному из глобальных максимумов при различных комбинациях значений параметров метода.

АНАЛИЗ РАБОТЫ ПОСЛЕДОВАТЕЛЬНО-ПАРАЛЛЕЛЬНОГО ГИБРИДНОГО МУЛЬТИАГЕНТНОГО МЕТОДА ПРИ РАЗЛИЧНЫХ ЗНАЧЕНИЯХ ПАРАМЕТРОВ

Результаты для корневой функции представлены в табл. 1.22.

Таблица 1.22. Влияние параметров метода. Корневая функция

Параметры метода						$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NP	P_{max}	thr	I_{contr}	ε_{eff}	N_{best}				
150	5	0,6	66	0,005	1	0,271148	0,060906	0,090891	85
150	2	0,6	66	0,005	1	0,266346	0,045699	0,089232	91
150	10	0,6	66	0,005	1	0,244174	0,029258	0,091768	91
150	7	0,3	66	0,005	1	0,261976	0,044424	0,097871	85
150	5	0,8	66	0,005	1	0,279353	0,012295	0,092262	93
150	5	0,8	33	0,005	1	0,248689	0,064805	0,089628	87
150	5	0,8	99	0,005	1	0,256101	0,037962	0,092517	90
150	5	0,8	66	0,05	1	0,262049	0,042106	0,098903	84
150	5	0,8	66	0,0005	1	0,286397	0,014715	0,089066	86
150	5	0,8	66	0,005	10	0,259744	0,020160	0,096712	84
150	5	0,8	66	0,005	5	0,253996	0,056461	0,101731	87
150	5	0,8	66	0,005	100	0,259030	0,074446	0,086479	85

Анализ работы последовательно-параллельного гибридного мультиагентного метода показал, что со всеми тестовыми функциями (табл. П.1) метод справляется успешно. В серии из 100 запусков метод находит глобальный максимум функций с высокой точностью, что объясняется применением сразу трех мультиагентных алгоритмов оптимизации. Но в тоже время с вычислительной точки зрения такой гибридный подход не тратит слишком много ресурсов и позволяет находить решение за приемлемое время.

1.6.6. Рекомендации по выбору параметров

РЕКОМЕНДАЦИИ ПО ВЫБОРУ ПАРАМЕТРОВ ПОСЛЕДОВАТЕЛЬНО-ПАРАЛЛЕЛЬНОГО ГИБРИДНОГО МУЛЬТИАГЕНТНОГО МЕТОДА

Размер популяции NP определяет количество вычислений целевой функции на каждой итерации. Для задачи с большой областью допустимых решений рекомендуется брать большее значение параметра NP. Рекомендуемое значение параметра $NP = 150$.

Максимальное число проходов $P_{\max}$ определяет, как долго будет продолжаться поиск новых решений. Рекомендуемые значения параметра $P_{\max} \in [2;5]$.

Пороговое значение $thr \in (0;1)$ определяет, какой метод оптимизации будет применяться чаще. Чем ближе значение параметра к единице, тем чаще будет использоваться метод, имитирующий поведении стаи рыб, чем ближе к нулю – метод, имитирующий поведение стаи криля.

Число последовательных итераций контроля эффективности метода I_{contr} характеризует досрочное завершение метода (метода стаи рыб или метода стаи криля) при выполнении условия текущей неэффективности метода, когда в течение заданного числа итераций I_{contr} изменение относительной величины приращения наилучшего значения целевой функции в популяции обновляется менее, чем на ε_{eff}. Проверка осуществляется при реализации последовательного этапа

Параметр ε_{eff} **для досрочного завершения прохода** определяет величину приращения наилучшего значения целевой функции с ростом числа итераций $I_{\max}$ или $ITER$.

Число наилучших решений в популяции $N_{best} \in (0, NP)$ определяет количество лучших агентов в популяции, которые остаются после преобразования популяции, остальные удаляются и вместо них генерируются новые агенты.

РЕКОМЕНДАЦИИ ПО ВЫБОРУ ПАРАМЕТРОВ МЕТОДА, ИМИТИРУЮЩЕГО ИМПЕРИАЛИСТИЧЕСКУЮ КОНКУРЕНЦИЮ

Число стран N_{pop} определяет количество вычислений целевой функции на каждой итерации. Для задачи с большой областью допустимых решений рекомендуется брать большее значение параметра N_{pop}. Рекомендуемые значения параметра $N_{pop} \in [50;300]$.

Число империалистических стран N_{imp} определяет, как долго будет продолжаться поиск новых решений. В конце поиска остается только одна империя. Для рассмотренного набора стандартных функций рекомендуемые значения $N_{imp} \in [5;30]$.

Параметры сдвига колоний β, γ определяют движение колоний к ее империи (см. шаг 5.2 алгоритма). Рекомендуемые значения параметра $\beta = 2$, $\gamma = \frac{\pi}{4}$.

Параметр учета влияния колоний ξ. Рекомендуемые значения параметра $\xi = 0,1$.

РЕКОМЕНДАЦИИ ПО ВЫБОРУ ПАРАМЕТРОВ МЕТОДА, ИМИТИРУЮЩЕГО ПОВЕДЕНИЕ СТАИ КРИЛЯ

Размер популяции NP определяет количество вычислений целевой функции на каждой итерации. Для задачи с большой областью допустимых решений рекомендуется брать большее значение параметра NP. Рекомендуемые значения параметра $NP \in [40;50]$.

Число итераций $I_{\max}$ определяет, как долго будет продолжаться поиск новых решений. Чем больше $I_{\max}$, тем более точным будет решение. Для рассмотренного набора

стандартных функций рекомендуемые значения в зависимости от сложности функции $I_{\max} \in [1000;10000]$.

Максимальная скорость движения криля $N_{\max}$, с помощью которой определяется скорость каждого члена стаи (см. шаг 3.5 указанного алгоритма). Рекомендуемое значение параметра $N_{\max} = 0,01$.

Константа μ **корректировки изменения** положения j-го члена популяции (см. шаги 3.2, 3.3, 4.3 указанного алгоритма) – малое положительное число. Рекомендуемое значение параметра $\mu = 3$.

Максимальная скорость передвижения к источнику пищи V_f, с помощью которой определяется скорость передвижения к пище каждого члена стаи (см. шаг 4.5). Рекомендуемое значение параметра $V_f = 0,02$.

Максимальная скорость диффузии криля $D_{\max}$. Рекомендуемое значение параметра $D_{\max} \in [0,002;0,01]$.

РЕКОМЕНДАЦИИ ПО ВЫБОРУ ПАРАМЕТРОВ МЕТОДА, ИМИТИРУЮЩЕГО ПОВЕДЕНИЕ СТАИ РЫБ

Размер популяции NP определяет количество вычислений целевой функции на каждой итерации. Для задачи с большой областью допустимых решений рекомендуется брать большее значение параметра NP. Рекомендуемые значения параметра $NP \in [30;40]$.

Число итераций $ITER$ определяет, как долго будет продолжаться поиск новых решений. Чем больше $ITER$, тем более точным будет решение. Для рассмотренного набора стандартных функций рекомендуемые значения в зависимости от сложности функции $ITER \in [1000;10000]$.

Максимальный вес W_{scale}, который может набрать особь. По истечению времени поиска, выбирается рыба с максимальным весом, и в качестве ответа берется ее положение. Рекомендованное значение этого параметра $W_{scale} \in [1000;5000]$.

Пороговый вес thr определяет тех рыб, которые будут допущены к размножению. Рекомендованное значение параметра $thr \in [900;4500]$.

Шаги индивидуального плавания $step_{ind}^k$ и коллективного плавания $step_{vol}^k$.
Рекомендуемые варианты начального и конечного значений шагов:

а) $step_{ind,initial} = 0,001 \cdot \max_i [b_i - a_i]$, $step_{ind,final} = 0,00001 \cdot \max_i [b_i - a_i]$,
$step_{vol,initial} = 0,01 \cdot \max_i [b_i - a_i]$, $step_{vol,final} = 0,0001 \cdot \max_i [b_i - a_i]$,

б) $step_{ind,initial} = 0,1 \cdot \max_i [b_i - a_i]$, $step_{ind,final} = 0,0001 \cdot \max_i [b_i - a_i]$,
$step_{vol,initial} = 0,1 \cdot \max_i [b_i - a_i]$, $step_{vol,final} = 0,001 \cdot \max_i [b_i - a_i]$,

в) $step_{ind,initial} = 0,01 \cdot \max_i [b_i - a_i]$, $step_{ind,final} = 0,00001 \cdot \max_i [b_i - a_i]$,
$step_{vol,initial} = 0,01 \cdot \max_i [b_i - a_i]$, $step_{vol,final} = 0,0001 \cdot \max_i [b_i - a_i]$,

г) $step_{ind,initial} = 0,1 \cdot \max_i [b_i - a_i]$, $step_{ind,final} = 0,0001 \cdot \max_i [b_i - a_i]$,
$step_{vol,initial} = 0,1 \cdot \max_i [b_i - a_i]$, $step_{vol,final} = 0,001 \cdot \max_i [b_i - a_i]$.

1.7. САМООРГАНИЗУЮЩИЕСЯ МИГРАЦИОННЫЕ АЛГОРИТМЫ

1.7.1. Стратегии поиска решения

Рассматривается задача (1.1) поиска условного глобального минимума целевой функции $f(x)$ на множестве допустимых решений D, т.е. такой точки $x^* \in D$, что

$$f(x^*) = \min_{x \in D} f(x),$$

где $x = (x_1, x_2, ..., x_n)^T$, $D = \{x \mid x_i \in [a_i, b_i], i = 1, 2, ..., n\}$.

САМООРГАНИЗУЮЩИЙСЯ МИГРАЦИОННЫЙ АЛГОРИТМ

Самоорганизующийся миграционный алгоритм (Self-Organizing Migrating Algorithm – SOMA) [77, 161, 162] может быть классифицирован, как эволюционный алгоритм оптимизации, основанный на самоорганизующемся поведении групп индивидов (агентов) в социальном окружении.

При решении задачи оптимизации используются конечные наборы $I = \left\{ x^j = \left(x_1^j, x_2^j, \ldots, x_n^j \right)^T, j = 1, 2, \ldots, NP \right\} \subset D$ возможных решений, называемые популяциями, где x^j – индивид (агент) с номером j, NP – размер популяции.

Самоорганизующийся миграционный алгоритм имитирует эволюцию начальной популяции $I_0 = \{x^j,\ j = 1, 2, \ldots, NP \mid x^j = \left(x_1^j, x_2^j, \ldots, x_n^j \right)^T \in D\}$ и представляет собой итерационный процесс, исследующий множество D.

Изначально популяция создается из индивидов со случайно сгенерированными координатами x_i из промежутка $[a_i, b_i]$ с помощью равномерного закона распределения.

Далее начинаются *миграционные циклы*. Перед каждым циклом выявляется лидер – индивид с наилучшим значением целевой функции $f(x)$. В этих циклах все индивиды последовательно двигаются к лидеру по прямой их соединяющей, в том числе продвигаясь за него на некоторое расстояние вдоль той же прямой. При этом лидер остается неподвижным. Затем выбирается та позиция индивида, которой соответствует наилучшее значение целевой функции, полученное в процессе поиска, и она записывается в следующую популяцию.

Алгоритм продолжается до того момента, пока разность значений целевой функции, соответствующих лидеру и худшему индивиду популяции, не станет меньше заранее заданного значения, или пока не реализуется максимальное число миграционных циклов.

Общая схема самоорганизующегося миграционного алгоритма SOMA представлена на рис. 1.54.

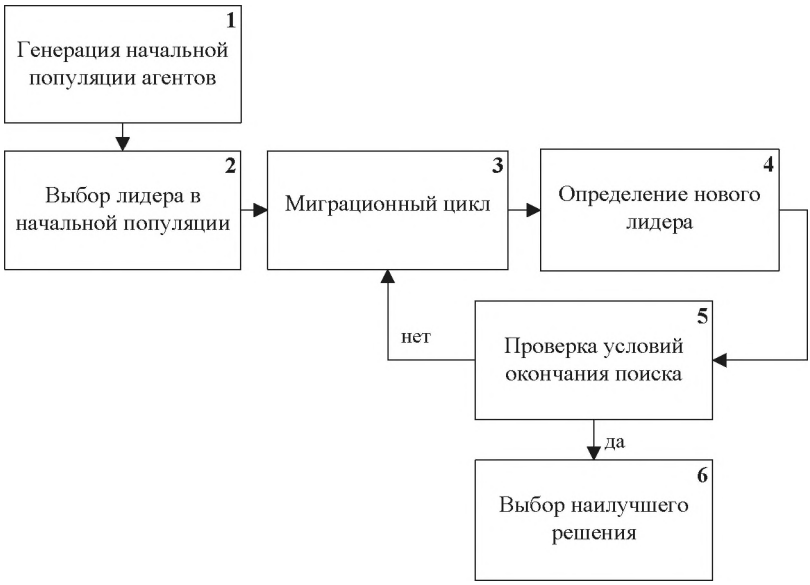


Рис. 1.54. Общая схема работы самоорганизующегося миграционного алгоритма SOMA

МОДИФИЦИРОВАННЫЙ САМООРГАНИЗУЮЩИЙСЯ МИГРАЦИОННЫЙ АЛГОРИТМ

При разработке модифицированного самоорганизующегося миграционного алгоритма оптимизации (MSOMA) [49, 142, 143] использовалась базовая версия алгоритма SOMA [77, 161, 162].

Модификации заключаются в выделении среди индивидов, образующих популяцию, трех лидеров. Для каждого из членов популяции генерируются два клона с той же позицией. Тем самым порождаются три популяции, каждая из которых реализует миграционный цикл относительно своего лидера. Для всех членов популяции находятся наилучшие положения, достигнутые в течение цикла. В процессе поиска популяция регулярно обновляется за счет новых индивидов, генерируемых на множестве допустимых решений. Они замещают выбывающих индивидов с наихудшими значениями целевой функции. После выполнения условий окончания производится уточняющий поиск (миграционный цикл), в котором участвуют три оставшихся лидера популяции. В качестве решения предъявляется наилучший результат.

При решении задачи оптимизации используются конечные наборы $I = \left\{ x^j = \left(x_1^j, x_2^j, \ldots, x_n^j \right)^T, j = 1, 2, \ldots, NP \right\} \subset D$ возможных решений, называемые популяциями, где x^j – индивид с номером j, NP – размер популяции.

Самоорганизующийся миграционный алгоритм имитирует эволюцию начальной популяции $I_0 = \{ x^j,\ j = 1, 2, \ldots, NP \mid x^j = \left(x_1^j, x_2^j, \ldots, x_n^j \right)^T \in D \}$ и представляет собой итерационный процесс, исследуюший множество D.

Начальная популяция создается из индивидов со случайно сгенерированными координатами x_i из промежутка $\left[a_i, b_i \right], i = 1, \ldots, n$.

Далее реализуются *миграционные циклы*. Перед каждым циклом индивиды упорядочиваются и выявляются три лидера – индивида с наилучшими значениями целевой функции $f(x)$. У каждого индивида создаются по два индивида-клона с такими же координатами. Таким образом, создается три популяции суммарного размера $K = 3NP$. Каждая из популяций движется относительно одного из трех лидеров. В этих циклах все индивиды последовательно с некоторой величиной шага двигаются по прямой к своему лидеру и вдоль этой же прямой за него на некоторое расстояние. При этом лидер остается неподвижным. Затем выбирается та позиция индивида, которой соответствует наилучшее значение целевой функции, полученное в процессе поиска. Эта позиция записывается в следующую популяцию размера K. После записи всех индивидов в новую популяцию из нее удаляются $2NP + \left[\frac{1}{3}NP\right]$ индивидов с худшими значениями целевой функции $f(x)$. Затем генерируются $\left[\frac{1}{3}NP\right]$ новых индивидов по правилам формирования начальной популяции для дополнения популяции до изначального размера NP с целью обновления популяции и исследования новых областей в множестве допустимых решений.

Алгоритм продолжается до тех пор, пока среднее отклонение по величине функции между тремя лидерами популяции не станет меньше заранее заданного значения или пока не реализуется максимальное число миграционных циклов. Тогда реализуется *уточняющий миграционный цикл* для трех лидеров популяции относительно абсолютного лидера. В качестве приближенного решения находятся наилучшее положение и соответствующее значение целевой функции, найденные в процессе поиска. Общая схема алгоритма MSOMA изображена на рис. 1.55.

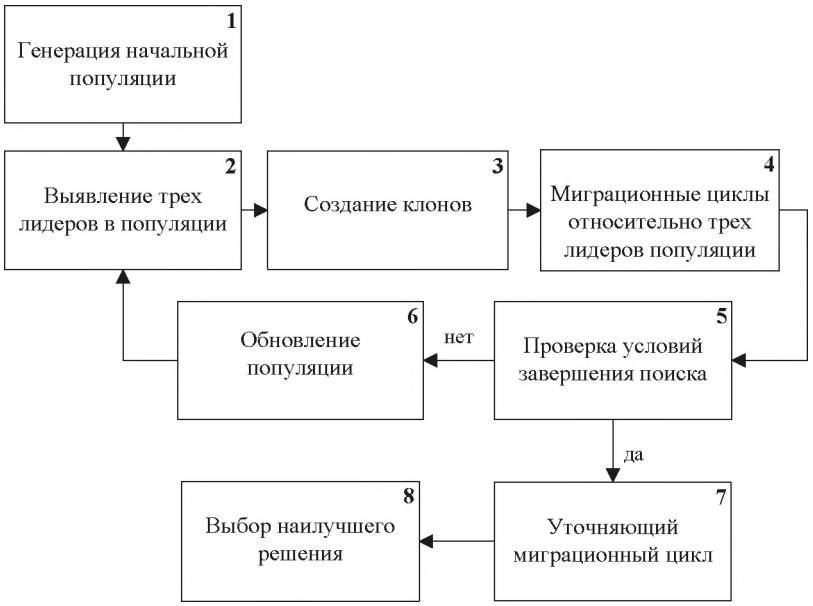


Рис. 1.55. Общая схема работы модифицированного самоорганизующегося миграционного алгоритма MSOMA

1.7.2. Алгоритмы решения задачи

САМООРГАНИЗУЮЩИЙСЯ МИГРАЦИОННЫЙ АЛГОРИТМ

Шаг 1. *Задание параметров метода*:

- контролирующий параметр *NStep*, определяющий количество шагов до окончания движения;
- контролирующий параметр *PRT*, определяющий, будет ли индивид двигаться по выбранной координате к лидеру;
- контролирующий параметр *NP*, определяющий размер популяции особей;
- останавливающий параметр *Migration*, определяющий максимальное количество итераций;
- останавливающий параметр *MinDiv*, показывающий предельную разницу величины целевой функции между лидером и худшим индивидом.

Шаг 2. *Создание начальной популяции.*

Шаг 2.1. Создать популяцию из NP индивидов со случайно сгенерированными координатами x_i из промежутка $[a_i, b_i]$ с использованием равномерного закона распределения:

$$x_i^j = a_i + rand_i[0;1] \cdot (b_i - a_i),\ i = 1,\ldots,n;\ j = 1,\ldots,NP.$$

где $rand_i[0;1]$ – равномерный закон распределения на отрезке $[0;1]$.

Шаг 2.2. Для каждого индивида вычислить значения целевой функции и среди индивидов выбрать лидера (особь с наилучшим значением):

$$x^{(L)} = \arg\min_{j=1,\ldots,Np} f\left(x^j\right).$$

Шаг 2.3. Положить $MCount = 1$ (счетчик числа миграционных циклов).

Шаг 3. *Миграционный цикл.*

Шаг 3.1. Перед началом движения каждого индивида к лидеру генерировать случайное число rnd_i на отрезке $[0;1]$ для каждой координаты индивида, а затем сравнить с заданной величиной PRT:

$$\text{если } rnd_i < PRT, \text{ то } PRTVector_i = 1, \text{иначе } 0,\ \ i = 1,\ldots,n.$$

Тем самым сформировать $PRTVector$, определяющий совокупность координатных направлений, по которым будет производиться поиск.

Шаг 3.2. Реализовать движение индивидов к лидеру. Движение происходит по шагам до тех пор, пока не будет достигнута конечная позиция на итерации под номером $NStep$, по следующей формуле

$$x^{j,m} = x^j + m\frac{\left(x^{(L)} - x^j\right)}{\left[\frac{NStep}{2}\right]} \otimes PRTVector,\quad m = 0,1,\ldots,NStep,$$

где $\otimes$ – поэлементное произведение векторов (по Адамару), $[\cdot]$ – целая часть числа.

Шаг 3.3. После всех шагов для каждого индивида найти наилучший шаг (такой шаг, на котором значение целевой функции было наименьшим), а индивиду занять

эту позицию, присваивая себе соответствующие значения координат, и перейти в следующую популяцию:

$$x^{j,New} = \underset{m=0,1,\ldots,NStep}{\operatorname{argmin}} f\left(x^{j,m}\right).$$

Шаг 3.4. Среди новых наилучших положений индивидов и положения старого лидера выбрать нового лидера $x^{(L)}$ и наихудшего индивида (worst) $x^{(W)}$.

$$x^{(L)} = \underset{j=1,\ldots,NP}{\arg\min} f(x^j), \quad x^{(W)} = \underset{j=1,\ldots,NP}{\arg\max} f(x^j).$$

Шаг 4. *Проверка условий завершения поиска.* Если разница величины целевой функции между лидером $f(x^{(L)})$ и худшим индивидом (worst) $f(x^{(W)})$ меньше *MinDiv*, т.е. $\left|f(x^{(L)}) - f(x^{(W)})\right| < MinDiv$, или достигнуто предельное число миграций (*Migration*), то положить $MCount = MCount + 1$ и перейти к шагу 5, а иначе перейти к шагу 3.

Шаг 5. *Выбор наилучшего решения.* В качестве решения задачи выбрать положение лидера популяции и соответствующее значение целевой функции.

Замечание.

1. Иллюстрация процесса миграции для двух произвольных индивидов изображена на рис. 1.56.

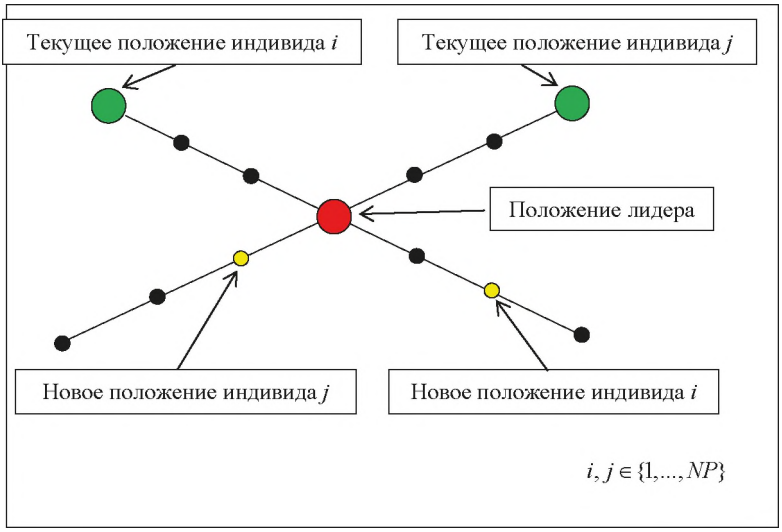


Рис. 1.56. Миграционный цикл в алгоритме SOMA для трех индивидов

2. На шаге 3.1 *PRTVector* для индивида с пятью координатами может, например, выглядеть следующим образом: $PRTVector = (0,1,1,0,1)^T$. Это означает, что поиск будет реализован только по координатам x_2, x_3, x_5.

Шаг 1. *Задать параметры метода*:

- контролирующий параметр *NStep*, определяющий количество шагов до окончания движения;
- контролирующий параметр *PRT* , определяющий, будет ли индивид двигаться по выбранной координате к лидеру;
- контролирующий параметр *NP*, определяющий размер популяции особей;
- останавливающий параметр *Migration*, задающий максимальное количество итераций;
- останавливающий параметр *MinDist*, отражающий величину среднего отклонения между тремя лидерами популяции по величине функции;
- счетчик итераций *MCount* , необходимый для остановки алгоритма при достижении им числа *Migration*.

Положить $MCount = 1$.

Шаг 2. *Создание начальной популяции.*

Создать популяцию из NP индивидов $\left\{x^1,\ldots,x^{NP}\right\}$ со случайно сгенерированными координатами x_i из промежутка $\left[a_i,b_i\right]$:

$$x_i^j = a_i + rand_i\left[0;1\right]\left(b_i - a_i\right),\ i = 1,\ldots,n;\ j = 1,\ldots,NP.$$

Шаг 2.1. Для каждого индивида вычислить значение целевой функции и выбрать лидеров (особей с наилучшими значениями). Для этого упорядочить популяцию в порядке неубывания целевой функции. Затем из нее выбрать первые три особи с наименьшими значениями целевой функции.

Шаг 3. *Миграционные циклы относительно трех лидеров.*

Шаг 3.1. Для всех индивидов создать двух «клонов» – индивидов с такими же координатами:

$$\text{для всех } j = 1,\ldots,NP:\ x_i^{Np+j} = x_i^j,\ x_i^{2Np+j} = x_i^j,\ i = 1,\ldots,n.$$

При этом индивид-родитель двигается к первому лидеру, первый клон – ко второму лидеру, второй клон – к третьему лидеру.

Шаг 3.2. Перед началом движения индивида к одному из лидеров генерировать случайное число rnd_i на отрезке $[0;1]$ для каждой координаты индивида, а затем сравнить с *PRT*:

$$\text{если } rnd_i < PRT, \text{то } PRTVector_i = 1, \text{иначе } PRTVector_i = 0,\ i = 1,\ldots,n.$$

Тем самым сформировать *PRTVector* , определяющий совокупность координатных направлений, по которым будет производиться поиск.

Шаг 3.3. Реализовать движение индивидов к соответствующему лидеру. Движение происходит по шагам до тех пор, пока не будет достигнута конечная позиция, по следующим формулам:

- для первого лидера:

$$x^{j,m} = x^j + m_1\frac{\left(x^1 - x^j\right)}{2NStep} \otimes PRTVector,\ m_1 = 0,1,\ldots,4NStep,\ j = 1,\ldots,NP;$$

- для второго лидера:

$$x^{j,m} = x^j + m_2 \frac{\left(x^2 - x^j\right)}{Nstep} \otimes PRTVector,\ \ m_2 = 0,1,\ldots,2NStep,\ \ j = \left(NP+1\right),\ldots,2NP;$$

- для третьего лидера:

$$x^{j,m} = x^j + m_3 \frac{\left(x^3 - x^j\right)}{\left[\frac{NStep}{2}\right]} \otimes PRTVector,\ \ m_3 = 0,1,\ldots,NStep,\ j = \left(2NP+1\right),\ldots,4NP,$$

где $\otimes$ – поэлементное произведение векторов (по Адамару), $[\cdot]$ – целая часть числа.

Шаг 3.4. После всех выполненных шагов для каждого индивида найти наилучшую позицию (на которой значение целевой функции было наименьшим), а индивиду занять эту позицию, присваивая себе соответствующие значения координат, после чего перейти в следующую популяцию:

- для первого лидера:

$$x^{j,New} = \underset{m_1=0,1,\ldots,4NStep}{\operatorname{argmin}} f\left(x^{j,m_1}\right),\ \ j = 1,\ldots,NP;$$

- для второго лидера:

$$x^{j,New} = \underset{m_2=0,1,\ldots,2NStep}{\operatorname{argmin}} f\left(x^{j,m_2}\right),\ \ j = \left(Np+1\right),\ldots,2NP;$$

- для третьего лидера:

$$x^{j,New} = \underset{m_3=0,1,\ldots,NStep}{\operatorname{argmin}} f\left(x^{j,m_3}\right),\ \ j = \left(2NP+1\right),\ldots,3NP.$$

Шаг 3.5. Отсортировать всех индивидов в порядке неубывания целевой функции.

Шаг 4. *Проверка условий завершения поиска.*

Если $\sqrt{\frac{1}{2}\sum_{j=2}^{3}\left[f\left(x^j\right) - f\left(x^1\right)\right]^2} < MinDist$ или достигнуто предельное число миграций (*Migration*), то необходимо перейти к шагу 6, а иначе перейти к шагу 5.

Шаг 5. *Обновление популяции.*

Шаг 5.1. Отсортировать всех индивидов по неубыванию целевой функции:

$$\{x^1,\ldots,x^{3NP} \mid f(x^k) \le f\left(x^{k+1}\right)\}, k = 1,\ldots,3NP-1.$$

Шаг 5.2. Удалить последних $2NP + \left[\frac{1}{3}NP\right]$ особей.

Шаг 5.3. Создать $\left[\frac{1}{3}NP\right]$ новых особей:

$$x_i^j = a_i + rand_i\left[0;1\right]\cdot\left(b_i - a_i\right),\ \ i = 1,\ldots,n;\ j = \left[\frac{2}{3}NP\right] + 2,\ldots,NP.$$

Шаг 5.4. Увеличить счетчик числа итераций: $MCount = MCount + 1$ и перейти к шагу 3.

Шаг 6. *Уточняющий миграционный цикл.*

Шаг 6.1. Увеличить параметр $NStep$ ($NStep = 10\,NStep$) и реализовать миграционный цикл для второго и третьего лидеров относительно первого лидера. Перед началом движения индивида к лидеру генерировать случайное число на отрезке $[0;1]$ для каждой координаты индивида, а затем сравнить с PRT:

$$\text{если } rnd_i < PRT, \text{то } PRTVector_i = 1, \text{иначе } PRTVector_i = 0,\ i = 1,\ldots,n.$$

Тем самым сформировать $PRTVector$, определяющий совокупность координатных направлений, по которым будет производиться поиск.

Движение второго и третьего лидеров реализуется по формуле

$$x^{j,m} = x^j + m\frac{\left(x^1 - x^j\right)}{\left[\frac{NStep}{2}\right]} \otimes PRTVector, \quad m = 0,1,\ldots,NStep;\ j = 1,2,3.$$

Шаг 6.2. Найти новые позиции лидеров: $x^{j,New} = \underset{m=0,1,\ldots,NStep}{\operatorname{argmin}} f\left(x^{j,m}\right), j = 1,2,3.$

Шаг 6.3. Возвращение наилучшего решения из найденных в процессе поиска:

$$x^{ans} = \underset{j=1,2,3}{\operatorname{argmin}} f\left(x^j\right).$$

З а м е ч а н и е. Иллюстрация миграционного цикла приведена на рис. 1.57.

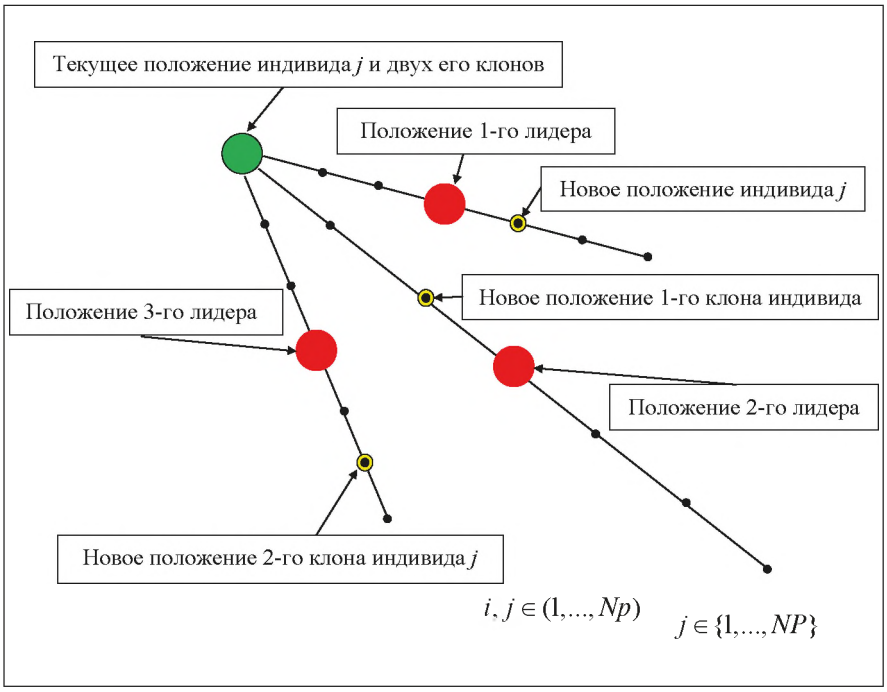


Рис. 1.57. Миграционный цикл в MSOMA одного индивида относительно трех лидеров

1.7.3. Программное обеспечение

На основе изложенных алгоритмов разработана программа поиска глобального экстремума функций [49, 142, 143]. Для ее создания использовалась среда разработки Microsoft Visual Studio, язык программирования C#.

Возможности программы позволяют изучить алгоритм метода, а также влияние параметров метода на результат его работы. В список выбираемых функций включены стандартные многоэкстремальные тестовые функции двух переменных, для которых известно точное решение (табл. П.1).

На рис. 1.58 представлено главное окно метода SOMA, где пользователь может выбрать вид целевой функции, задать множество допустимых решений и параметры метода, просматривать результаты работы. На рис. 1.59 представлено главное окно метода MSOMA.

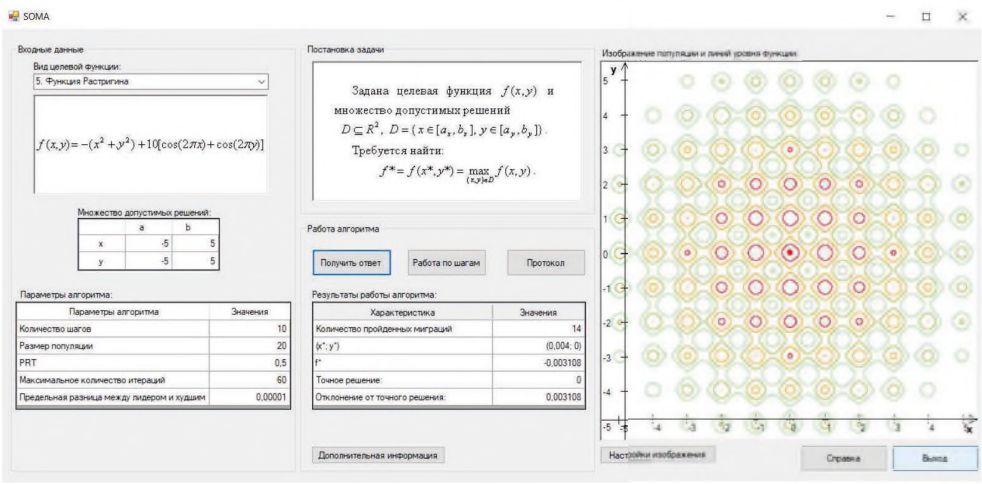


Рис. 1.58. Главное окно «Самоорганизующийся миграционный алгоритм SOMA»

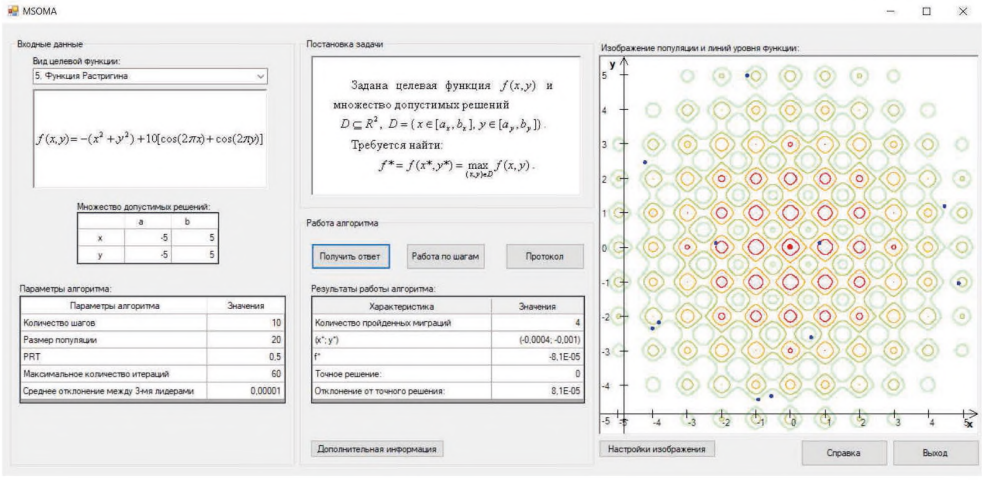


Рис. 1.59. Главное окно «Модифицированный самоорганизующийся миграционный алгоритм MSOMA»

Разработанная программа предусматривает возможность анализа работы метода по шагам. На рис. 1.60 представлено окно пошаговой работы метода SOMA, где изображена общая схема метода, отображаются результаты работы после выполнения каждого шага (график изменения наилучшего значения целевой функции, а также графическое изображение популяции) и после завершения работы метода. На рис. 1.61 представлено аналогичное окно пошаговой работы метода MSOMA.

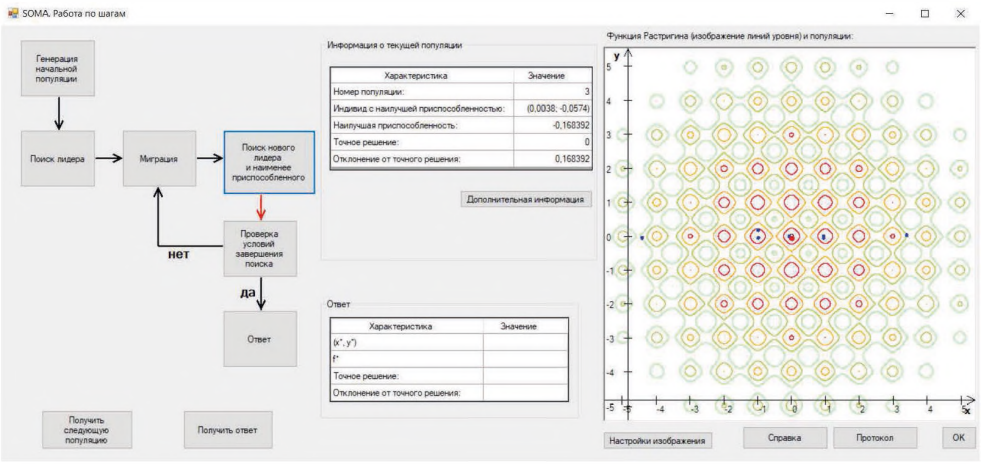


Рис. 1.60. Окно пошагового режима работы SOMA

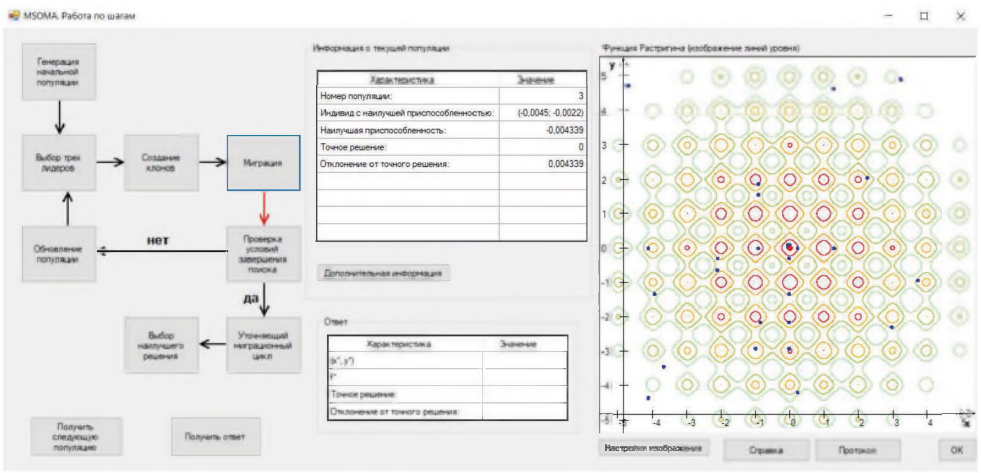


Рис. 1.61. Окно пошагового режима работы MSOMA

1.7.4. Тестовые примеры

Пример 1.15. Найдем глобальный максимум корневой функции (табл. П.1). Зададим множество допустимых решений $x, y \in [-2; 2]$.

Выберем следующие параметры алгоритма SOMA:

- количество шагов $NStep = 50$;

- размер популяции $NP = 20$;
- параметр $PRT = 0,6$;
- количество миграций $Migration = 60$;
- минимальная разница $MinDiv = 0,000001$.

На рис. 1.62 представлена популяция на начальной ($k = 1$), промежуточных ($k = 5$, $k = 10$) и конечной ($k = 25$) итерациях.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (0,5; -0,866)$;
- значение целевой функции $f(x^*, y^*) = 1$;
- отклонение от точного решения $\Delta = 0$.

График изменения наилучшего значения целевой функции представлен на рис. 1.63.

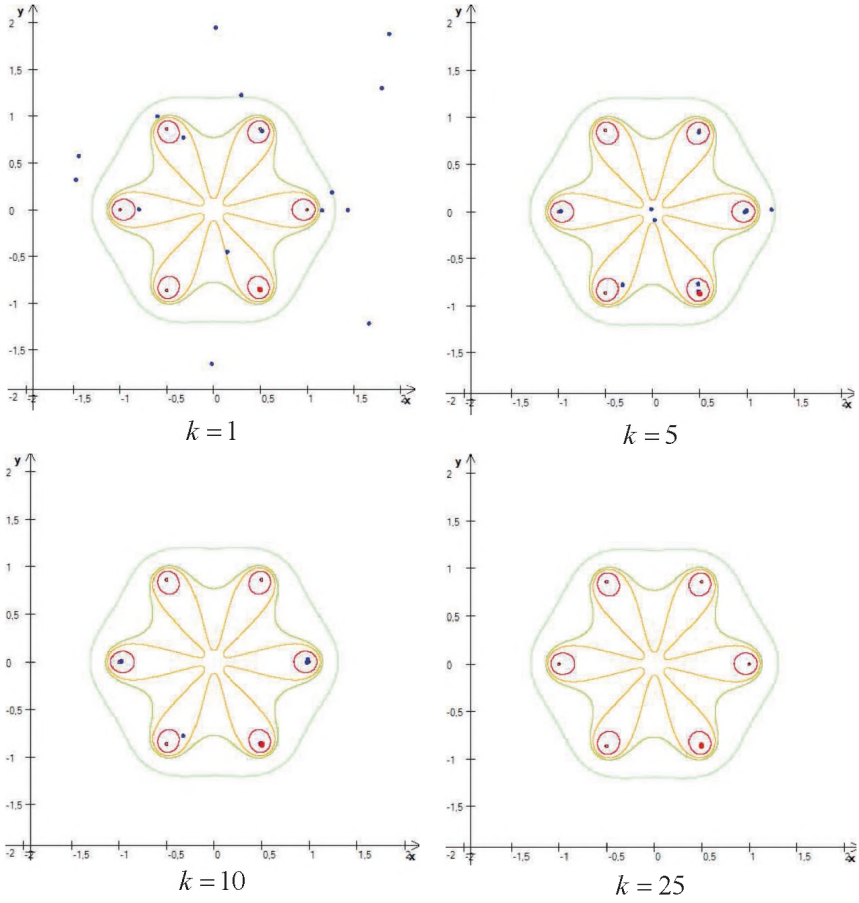


Рис. 1.62. Начальная, промежуточные и конечная итерации

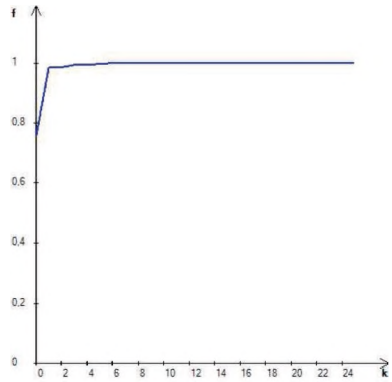


Рис. 1.63. График изменения наилучшего значения целевой функции

Выберем следующие параметры алгоритма MSOMA:

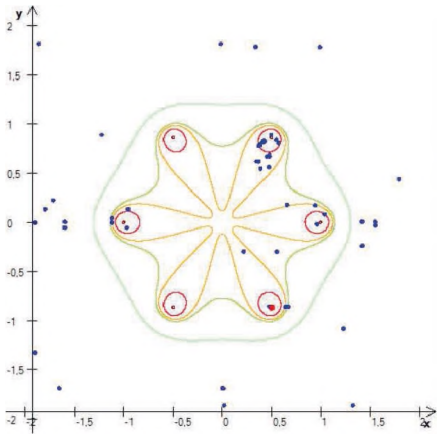
- количество шагов $NStep = 50$;
- размер популяции $NP = 20$;
- параметр $PRT = 0,6$;
- количество миграций $Migration = 60$;
- минимальная разница $MinDist = 0,000001$.

На рис. 1.64 представлена популяция на начальной ($k = 1$), промежуточных ($k = 3$, $k = 5$) и конечной ($k = 6$) итерациях.

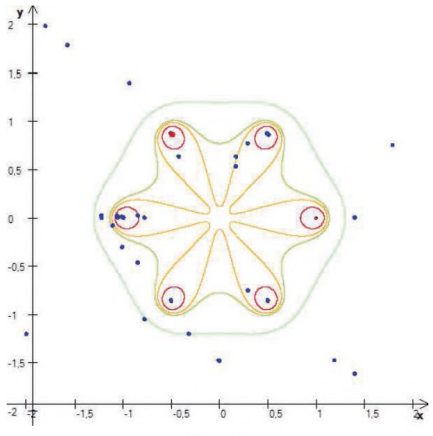
Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (-0,5; 0,866)$;
- значение целевой функции $f(x^*, y^*) = 1$;
- отклонение от точного решения $\Delta = 0$.

График изменения наилучшего значения целевой функции представлен на рис. 1.65.



$k = 1$



$k = 3$

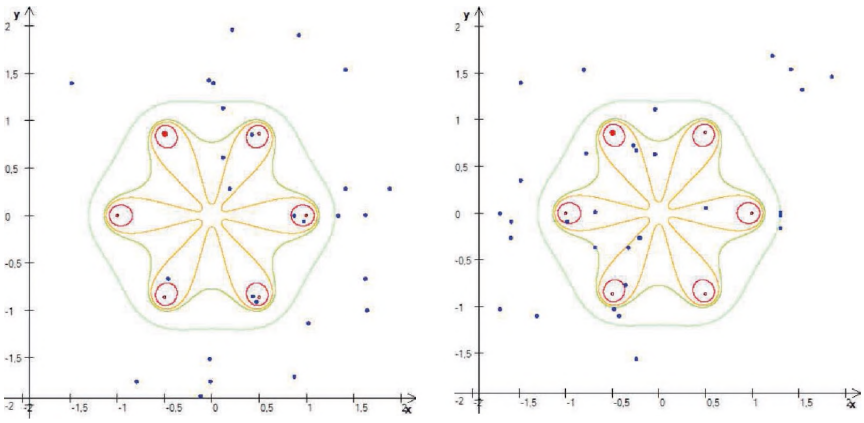


Рис. 1.64. Начальная, промежуточные и конечная итерации

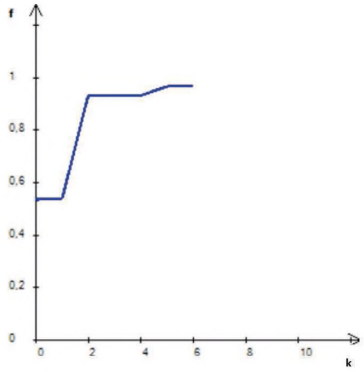


Рис. 1.65. График изменения наилучшего значения целевой функции

Пример 1.16. Найдем глобальный максимум функции Розенброка (табл. П.1). Зададим множество допустимых решений $x \in [-3;3]$, $y \in [-1;5]$.

Выберем следующие параметры алгоритма SOMA:

- количество шагов $NStep = 50$;
- размер популяции $NP = 20$;
- параметр $PRT = 0,4$;
- количество миграций $Migration = 60$;
- минимальная разница $MinDiv = 0,00001$.

На рис. 1.66 представлена популяция на начальной ($k = 1$), промежуточных ($k = 5, k = 10$) и конечной ($k = 12$) итерациях.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (1;1)$;
- значение целевой функции $f(x^*, y^*) = 0$;
- отклонение от точного решения $\Delta = 0$.

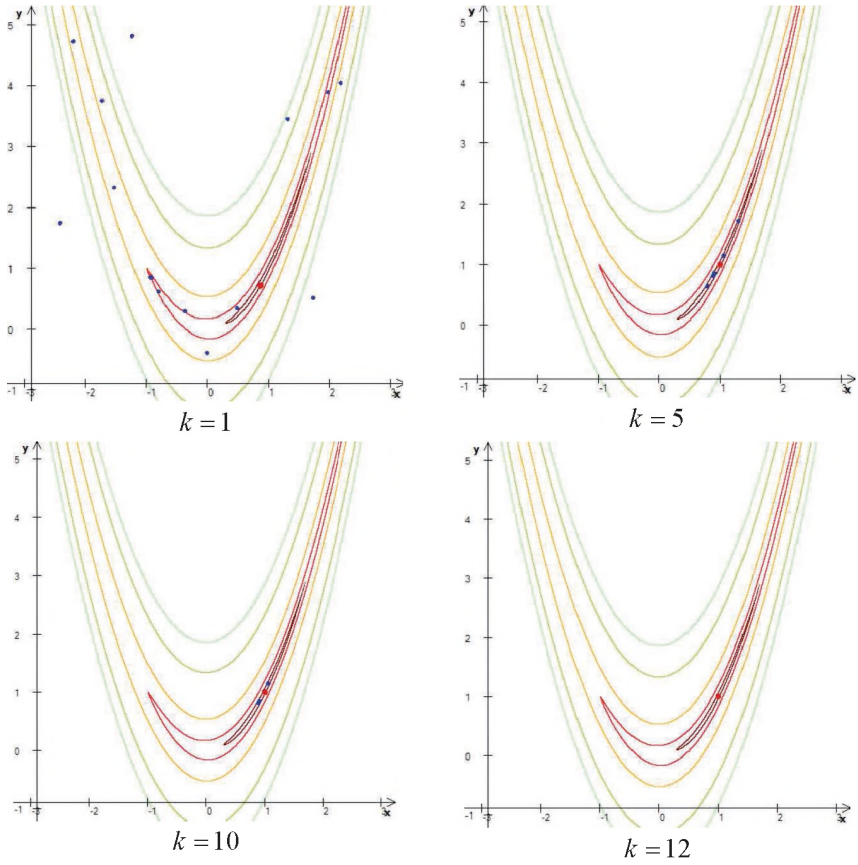


Рис. 1.66. Начальная, промежуточные и конечная популяции

График изменения наилучшего значения целевой функции представлен на рис. 1.67.

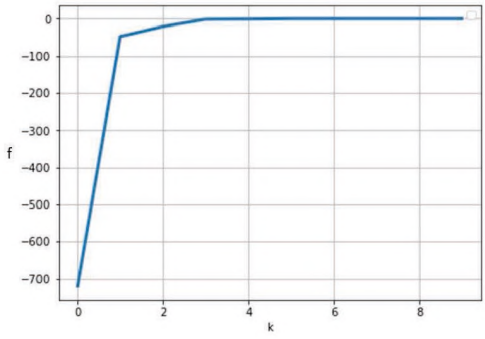


Рис. 1.67. График изменения наилучшего значения целевой функции

Выберем следующие параметры алгоритма MSOMA:

- количество шагов $NStep = 50$;
- размер популяции $NP = 20$;
- параметр $PRT = 0,4$;
- количество миграций $Migration = 60$;
- минимальная разница $MinDist = 0,00001$.

На рис. 1.68 представлена популяция на начальной ($k = 1$), промежуточных ($k = 3, k = 5$) и конечной ($k = 10$) итерациях.

Результаты работы алгоритма:

- наилучшее решение $(x^*; y^*) = (1;1)$;
- значение целевой функции $f(x^*, y^*) = 0$;
- отклонение от точного решения $\Delta = 0$.

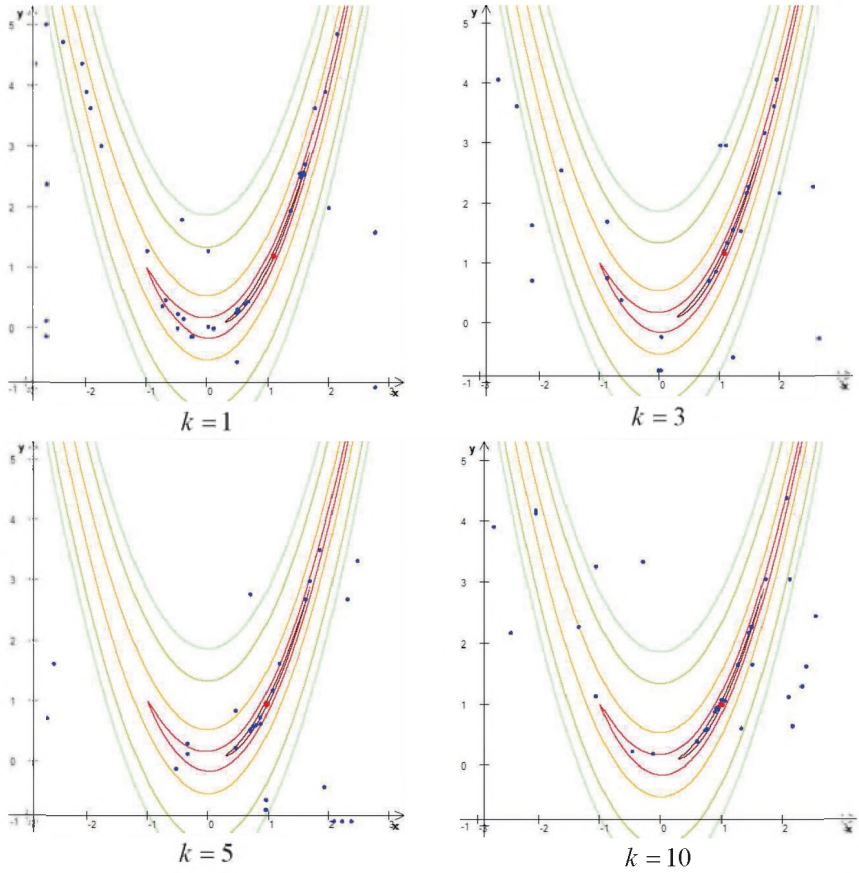


Рис. 1.68. Начальная, промежуточные и конечная популяции

График изменения наилучшего значения целевой функции представлен на рис. 1.69.

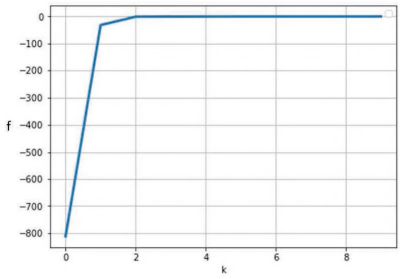


Рис. 1.69. График изменения наилучшего значения целевой функции

1.7.5. Анализ эффективности методов

АНАЛИЗ РАБОТЫ МЕТОДОВ ПРИ РАЗЛИЧНЫХ ЗНАЧЕНИЯХ ПАРАМЕТРОВ

В данном разделе приводится статистический анализ и сравнение работы методов SOMA и MSOMA при различных значениях их параметров. Исследуемый метод применялся к некоторым функциям из набора тестовых функций (табл. П.1). Для каждой функции проводились серии из 100 решений одной и той же задачи с одними и теми же значениями параметров. Для полученной выборки $\left\{f^1, f^2, ..., f^{100}\right\}$ вычислялись среднее значение отклонения полученного решения от точного $\overline{\Delta f} = \frac{1}{100}\sum_{i=1}^{100}\Delta f_i$, где $\Delta f_i = \left|f(x^*) - f^i\right|$; наименьшее значение отклонения $\Delta f_{best} = \min_i \Delta f_i$; среднеквадратическое отклонение $\overline{\sigma}_f = \sqrt{\overline{S}_{100}}$, где $\overline{S}_{100} = \frac{1}{100}\sum_{i=1}^{100}(\Delta f^i - \overline{\Delta f})^2$; количество успехов $n_{\text{усп}}$ (успехом считалось попадание лучшей точки в ε–окрестность точного решения, $\varepsilon = \frac{\max\limits_{i=1,\ldots,n}\left|b_i - a_i\right|}{1000}$). Результаты, полученные для каждой функции, представлены в табл. 1.23–1.28.

Предложенный алгоритм MSOMA реализован на языке python (ссылка на реализацию: https://github.com/Vradislav/MSOMA). Для тестирования использован стандартный набор тестовых функций со сложной структурой линий уровня.

Функция «Птица».

Целевая функция $f(x, y) = -\sin x \exp((1-\cos y)^2) - \cos y \exp((1-\sin x)^2) - (x-y)^2$, множество допустимых решений $x, y \in [-2\pi; 2\pi]$, глобальный максимум: $(x^*; y^*)^T = (-1,5708; -3,1416)^T$ или $(x^*; y^*)^T = (4,7124; 3,1416)^T$, $f(x^*, y^*) = 106,7645$.

Лучшее решение: наибольшее значение целевой функции: 106,76453675;

$(x^*; y^*)^T = (4,70104503;\ 3,15293953)^T$, $(x^*; y^*)^T = (-1,57810686;\ -3,13024376)^T$.

Таблица 1.23. Влияние параметров метода MSOMA. Функция «Птица»

Параметры					$\overline{f}$	Наибольшее значение f	$\overline{\sigma_f}$	$n_{усп}$
NStep	*PRT*	*NP*	*Migration*	*MinDist*				
3	0,4	6	30	10^{-6}	95,161	106,764485	19,227863	64
5	0,4	10	30	10^{-6}	103,38	106,764536	5,762529	73
9	0,4	15	30	10^{-6}	106,41	106,764536	3,996871	79
9	0,5	15	30	10^{-6}	106,64	106,764536	0,221813	84
12	0,5	15	30	10^{-6}	106,68	106,764536	0,141877	87
12	0,7	20	30	10^{-10}	106,74	106,764536	0,073805	86
17	0,6	26	30	10^{-10}	106,76	106,764536	0,003814	92
20	0,7	30	40	10^{-12}	106,76	106,764536	$1,293234 \cdot 10^{-6}$	99

Таблица 1.24. Влияние параметров метода SOMA. Функция «Птица»

Параметры					$\overline{f}$	Наибольшее значение f	$\overline{\sigma_f}$	$n_{усп}$
NStep	*PRT*	*NP*	*Migration*	*MinDiv*				
40	0,6	500	40	10^{-6}	105,72	106,764834	4,809178	80
20	0,1	1000	150	0,0001	105,669	106,764536	1,100343	72
20	0,1	500	150	0,0001	105,669	106,76453	18,899009	49
20	0,1	1000	100	0,0001	106,764	106,76453	11,0034365	53
10	0,1	1000	150	0,0001	94,721	106,76453	28,4178546	44
20	0,1	1000	200	0,0001	106,764	106,76453	$1,76552 \cdot 10^{-12}$	89
20	0,4	1000	200	0,0001	75,725	106,76453	30,8575606	35

Среднее время одного запуска MSOMA с параметрами для лучшего решения в таблице 1.23 равно 0,385252666 с. Среднее время одного запуска SOMA с параметрами для лучшего решения в таблице 1.24 равно 20,35154864717 с.

Трехгорбая функция.

Целевая функция $f(x,y) = -2x^2 + 1,05x^4 - (1/6)x^6 - xy - y^2$, множество допустимых решений $x, y \in [-5;5]$, глобальный максимум: $(x^*;y^*)^T = (0;0)^T$, $f(x^*,y^*) = 0$. Лучшее приближение: $(x^*;y^*)^T = (0;0)^T$. Наибольшее значение целевой функции: 0.

Среднее время одного запуска MSOMA с параметрами для лучшего решения в таблице 1.25 равно 0,2378336048126 с. Среднее время одного запуска SOMA с параметрами для лучшего решения в таблице 1.26 равно 0,00767917633056 с.

Таблица 1.25. Влияние параметров метода MSOMA. Трехгорбая функция

Параметры					$\overline{f}$	Наибольшее значение f	$\overline{\sigma_f}$	$n_{усп}$
NStep	*PRT*	*NP*	*Migration*	*MinDist*				
3	0,4	3	10	10^{-6}	–0,01383	$-1,5769\cdot10^{-5}$	0,035846	84
5	0,6	10	10	10^{-6}	–0,00214	$-4,4002\cdot10^{-17}$	0,009538	93
10	0,6	20	10	10^{-6}	$-2,61\cdot10^{-5}$	$-3,3561\cdot10^{-19}$	$2,6127\cdot10^{-5}$	99
10	0,4	20	15	10^{-10}	$-5,51\cdot10^{-5}$	$-1,814\cdot10^{-18}$	0,000163	97
30	0,6	25	20	10^{-15}	0,0	0,0	0,0	100

Таблица 1.26. Влияние параметров метода SOMA. Трехгорбая функция

Параметры					$\overline{f}$	Наибольшее значение f	$\overline{\sigma_f}$	$n_{усп}$
NStep	*PRT*	*NP*	*Migration*	*MinDiv*				
10	0,7	15	20	0,01	0,0	0,0	0,0	100
3	0,1	2	4	0,0001	–0,07043	0,0	0,07079321	96
5	0,1	10	4	0,0001	–0,07793	0,0	0,07829649	95
10	0,4	20	5	0,0001	0,0	0,0	0,00993959	98
10	0,4	20	15	0,0001	0,0	0,0	0,0	100
10	0,4	20	15	0,01	0,0	0,0	0,0	100

Функция Экли.

Целевая функция

$$f(x,y)=-e+20\exp\left(-\sqrt{\frac{x^2+y^2}{50}}\right)+\exp\left(\frac{1}{2}(\cos(2\pi x)+\cos(2\pi y))\right),$$ множество допустимых решений $x,y\in[-10;10]$, глобальный максимум: $(x^*;y^*)^T=(0;0)^T$, $f(x^*,y^*)=20$. Лучшее приближение: $(x^*;y^*)^T=(0;0)^T$. Наибольшее значение целевой функции: 20.

Таблица 1.27. Влияние параметров метода MSOMA. Функция Экли

Параметры					$\overline{f}$	Наибольшее значение f	$\overline{\sigma_f}$	$n_{усп}$
NStep	*PRT*	*NP*	*Migration*	*MinDist*				
5	0,4	5	20	0,00001	18,57383	19,999589	2,200696	76
10	0,4	10	20	0,00001	19,86717	19,999999	0,456749	81
10	0,5	15	20	0,0001	19,98560	19,999999	0,247724	88
15	0,5	20	100	0,0001	19,97390	19,999998	0,025973	90
20	0,6	30	100	0,0001	19,99987	19,999999	0,015440	92
20	0,6	30	100	10^{-7}	19,99996	20,0	$5,11075\cdot10^{-5}$	99

Таблица 1.28. Влияние параметров метода SOMA. Функция Экли

Параметры					$\overline{f}$	Наибольшее значение f	$\overline{\sigma_f}$	$n_{усп}$
NStep	*PRT*	*NP*	*Migration*	*MinDiv*				
10	0,7	15	20	0,01	20,0	20,0	0,0	100
5	0,1	5	50	0,0001	19,7024	20,0	1,56276235	89
5	0,1	15	100	0,0001	20,0	20,0	0,569074619	95
10	0,1	15	100	0,0001	20,0	20,0	0,0	100
10	0,5	15	100	0,0001	20,0	20,0	0,0	100
10	0,5	15	100	0,01	20,0	20,0	0,0	100

Среднее время одного запуска MSOMA с параметрами для лучшего решения в таблице 1.27 равно 0,688916547 с.

Среднее время одного запуска SOMA с параметрами для лучшего решения в таблице 1.28 равно 0,009973287582397 с.

Время работы программы зависит как от реализации алгоритма, так и от конфигурации компьютера, следовательно, является относительной величиной. Оно представлено только для сравнения скорости нахождения решения задачи оптимизации по сравнению с алгоритмом SOMA. Так как сложности алгоритмов SOMA и MSOMA почти одинаковы, точность найденных наилучших значений и среднее время работы одного запуска, соответствующие этому значению, показывают, какой из алгоритмов лучше справляется с конкретной тестовой функцией.

Исходя из представленных результатов тестирования, можно сделать вывод, что алгоритм MSOMA быстрее и точнее работает с теми функциями, которые для базовой версии алгоритма SOMA являются более сложными. Это преимущество возникает из-за разного способа движения индивидов популяции (движение в MSOMA покрывает большую область из-за движения вокруг трех разных лидеров и добавления новых особей), а также разных условий остановки. Но одновременно это различие в способах движения особей заставляет алгоритм MSOMA дольше работать с теми функциями, с которыми базовая версия алгоритма SOMA справляется достаточно быстро и точно. Таким образом, алгоритм MSOMA лучше показывает себя на тех функциях, на которых для получения достаточно точного результата требуется большее покрытие множества допустимых решений для нахождения экстремума.

В главе 4 приведены результаты применения обоих миграционных алгоритмов оптимизации в задачах поиска оптимального программного управления детерминированными дискретными динамическими системами, поиска оптимального в среднем программного управления пучками траекторий дискретных динамических систем, поиска оптимального в среднем программного управления дискретными стохастическими динамическими системами, что свидетельствует об их потенциальных возможностях. В целом миграционные алгоритмы глобальной оптимизации могут быть рекомендованы для решения широкого спектра сложных технических задач.

Размер популяции *NP*. Чем больше *NP*, тем лучше получаемое решение задачи, но замедляется работа алгоритма. Рекомендуемое значение $NP \geq 20$.

Контролирующий параметр *PRT*. При уменьшении числа членов популяции необходимо увеличивать значение *PRT*. Рекомендуемое значение $PRT \in [0,1; 0,5]$ при $NP \geq 20$.

Количество шагов *NStep*. Чем больше значение *NStep*, тем возможно получение более точного решения задачи, но замедляется работа алгоритма. Рекомендуемое значение $NStep \geq 20$.

Предельное количество миграций *Migration*. Для корректной работы алгоритма значение *Migration* должно быть больше некоторого числа, определяемого сложностью решаемой задачи (рекомендуемое значение $Migration \geq 50$).

Останавливающий параметр в алгоритме SOMA *MinDiv*. При уменьшении значения *MinDiv* увеличивается время работы алгоритма. Рекомендуемое значение $MinDiv \in [0,0001; 0,01]$.

Останавливающий параметр в алгоритме MSOMA *MinDist*. При уменьшении величины *MinDist* увеличивается время работы алгоритма, но одновременно увеличивается и точность. Рекомендуемое значение $MinDist \in [10^{-10}; 10^{-5}]$.

1.8. МУЛЬТИАГЕНТНЫЙ МИГРАЦИОННЫЙ АЛГОРИТМ С ПРОГНОЗИРУЮЩИМИ ДИНАМИЧЕСКИМИ МОДЕЛЯМИ ДВИЖЕНИЯ АГЕНТОВ

1.8.1. Стратегия поиска решения

Рассматривается задача (1.1) поиска условного глобального минимума целевой функции $f(x)$ на множестве допустимых решений D, т.е. такой точки $x^* \in D$, что

$$f(x^*) = \min_{x \in D} f(x),$$

где $x = (x_1, x_2, ..., x_n)^T$, $D = \{x \mid x_i \in [a_i, b_i], i = 1, 2, ..., n\}$.

В процессе поиска экстремума имитируется поведение стаи мигрирующих птиц: соек, снегирей, синиц, поползней, скворцов, осуществляющих исследование территории своего обитания в поисках пищи. Они играют роль движущихся агентов. В начале процесса стая рассредоточивается на множестве допустимых решений D. Среди всех членов стаи (агентов) находится лидер, которому соответствует наилучшее значение целевой функции (считается, что он нашел привлекательный источник пищи, жертву для атаки). Лидер сообщает всем членам стаи свою позицию.

Остальные члены стаи (агенты) реализуют полет в режиме последовательной миграции, применяя алгоритм управления с прогнозирующей моделью с фиксированными горизонтами прогнозирования и управления, двигаясь по направления к лидеру, но продолжая поиск других источников пищи во время движения. Этот процесс сопровождается наличием «живого облака», называемым «мурмурацией» птиц. Через заданный промежуток дискретного времени, определяемый горизонтом прогнозирования, в стае находится новый лидер, а остальные члены стаи далее реализуют приближение к нему. В процессе поиска происходит заданное число смен лидера. В результате находится положение лидера, которое принимается за перспективную цель для дальнейшей разработки (жертву).

Из ближайших к лидеру особей по достигнутой величине целевой функции, но отстоящих от него по расстоянию не менее чем на пороговую величину, выбираются еще два члена стаи. Они атакуют жертву (присоединяются к лидеру), используя в качестве управления алгоритм с прогнозирующей моделью, при реализации которого на основании информации о положении лидера совершают только одну итерацию из возможных на горизонте прогнозирования, после которой обмениваются информацией и оперативно пересчитывают законы управления. В процессе их движения к жертве запоминаются наилучшие положения. В результате считается, что окрестность цели достаточно хорошо исследуется. Найденное наилучшее решение (положение лидера) помещается в множество *Pool*.

Затем продолжается процесс миграции на множестве допустимых решений всей стаей с целью поиска нового источника пищи. Для этого последовательно применяется алгоритм с прогнозирующей моделью, при реализации которого каждый агент (член стаи) совершает фиксированное случайное количество итераций, ограниченное горизонтом прогнозирования. По окончании очередного процесса миграции все агенты обмениваются информацией и оперативно пересчитывают законы управления.

После осуществления заданного числа итераций агенты ранжируются по величине целевой функции. Среди них сразу удаляются σ% самых плохих по величине

функции (улетают, отделяются от стаи). Далее среди двух членов популяции, отстоящих друг от друга менее, чем на заданную величину $dist$, остается лучший по величине целевой функции. Вместо выбывших агентов генерируются новые с помощью равномерного закона распределения на множестве допустимых решений.

Процесс поиска продолжается с повторения режима миграции с заданными горизонтами прогнозирования и управления. Поиск завершается при достижении заданного общего числа итераций.

МОДЕЛИ ДВИЖЕНИЯ АГЕНТОВ

Первая модель. Модель движения агента в непрерывном времени t :

$$\ddot{x} = u \Leftrightarrow \begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = u \end{cases} \Leftrightarrow \dot{x} = A_c x(t) + B_c u(t),$$

где x_1 – положение, x_2 – скорость, $x = (x_1, x_2)^T$ – вектор состояния, u – управление полетом, $A_c = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, B_c = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

Модель движения агента в дискретном времени ($t = kh$):

$$\begin{cases} x_1(k+1) = x_1(k) + hx_2(k) + \dfrac{h^2}{2}u(k), \\ \qquad x_2(k+1) = x_2(k) + hu(k), \end{cases} \Leftrightarrow \quad x(k+1) = Ax(k) + Bu(k), \quad k = 0, 1, \ldots$$

где $A = \begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix}, B = \begin{pmatrix} \dfrac{h^2}{2} \\ h \end{pmatrix}$. Соответствующую модель движения в дискретном времени можно получить, интегрируя уравнения непрерывной модели последовательно на одном шаге величиной h, начиная со второго уравнения.

В n-мерном случае, рассматриваемом в поставленной задаче оптимизации, непрерывная и дискретная модели движения имеют вид

$$\dot{x} = A_c x(t) + B_c u(t), \; A_c = \begin{pmatrix} O_n & E_n \\ O_n & O_n \end{pmatrix}, B_c = \begin{pmatrix} O_n \\ E_n \end{pmatrix},$$

$$x(k+1) = Ax(k) + Bu(k), \; A = \begin{pmatrix} E_n & hE_n \\ O_n & E_n \end{pmatrix}, B = \begin{pmatrix} \dfrac{h^2}{2}E_n \\ hE_n \end{pmatrix}, \tag{1.27}$$

где $x \in R^{2n}$ – вектор состояния, определяемый положением (первые n координат) и скоростью (вторые n координат), $u \in R^n$ – вектор управления, на который ограничения не накладываются:

$$x = (x_1, \ldots, x_n, x_{n+1}, \ldots, x_{2n})^T, u = (u_1, \ldots, u_n)^T,$$

$x_i, i = 1, \ldots, n$ – координаты положения, $x_{i+n}, i = 1, \ldots, n$ – скорости изменения координаты x_i, $u_i, i = 1, \ldots, n$ – управление i-й координатой.

Вторая модель. Модель движения агента в непрерывном времени t :

$$\ddot{x} = u \Leftrightarrow \begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = u \end{cases} \Leftrightarrow \dot{x} = A_c x(t) + B_c u(t),$$

где x_1 – положение, x_2 – скорость, $x = (x_1, x_2)^T$ – вектор состояния, u – управление полетом, $A_c = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, B_c = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

Модель движения агента в дискретном времени ($t = kh$):

$$\begin{cases} x_1(k+1) = x_1(k) + hx_2(k), \\ x_2(k+1) = x_2(k) + hu(k), \end{cases} \Leftrightarrow \quad x(k+1) = Ax(k) + Bu(k), \quad k = 0,1,...$$

где $A = \begin{pmatrix} 1 & h \\ 0 & 1 \end{pmatrix}, B = \begin{pmatrix} 0 \\ h \end{pmatrix}$. Соответствующую модель движения в дискретном времени можно получить, используя в левой части уравнения непрерывной модели конечно–разностные формулы для первых производных.

В n-мерном случае, рассматриваемом в поставленной задаче оптимизации, непрерывная и дискретная модели движения имеют вид

$$\dot{x} = A_c x(t) + B_c u(t), \; A_c = \begin{pmatrix} O_n & E_n \\ O_n & O_n \end{pmatrix}, B_c = \begin{pmatrix} O_n \\ E_n \end{pmatrix},$$

$$x(k+1) = Ax(k) + Bu(k), \; A = \begin{pmatrix} E_n & hE_n \\ O_n & E_n \end{pmatrix}, B = \begin{pmatrix} O_n \\ hE_n \end{pmatrix}, \tag{1.28}$$

где $x \in R^{2n}$ – вектор состояния, $u \in R^n$ – вектор управления.

Третья модель. Модель движения агента в непрерывном времени:

$$\ddot{x} = u,$$

где x – положение, u – управление полетом.

Соответствующую модель движения агента в дискретном времени ($t = kh$) получим, используя конечно-разностную аппроксимацию второй производной:

$$\frac{x(k+2) - 2x(k+1) + x(k)}{h^2} = u(k) \to x(k+2) - 2x(k+1) + x(k) = h^2 u(k).$$

Вводя обозначения $x(k) = x_1(k), \; x(k+1) = x_2(k)$, получаем

$$\begin{cases} x_1(k+1) = x_2(k), \\ x_2(k+1) = -x_1(k) + 2x_2(k) + h^2 u(k), \end{cases} \Leftrightarrow \quad x(k+1) = Ax(k) + Bu(k), \quad k = 0,1,...$$

где $A = \begin{pmatrix} 0 & 1 \\ -1 & 2 \end{pmatrix}, B = \begin{pmatrix} 0 \\ h^2 \end{pmatrix}$.

В n-мерном случае дискретная модель движения имеет вид

$$x(k+1) = Ax(k) + Bu(k), \; A = \begin{pmatrix} O_n & E_n \\ -E_n & 2E_n \end{pmatrix}, B = \begin{pmatrix} O_n \\ h^2 E_n \end{pmatrix}, \tag{1.29}$$

где $x \in R^{2n}$ – вектор состояния, определяемый положением (первые n координат) и скоростью (вторые n координат), $u \in R^n$ – вектор управления.

Стратегия поиска реализуется с помощью нескольких этапов.

Первый этап. Генерация начального положения агентов на множестве допустимых решений. Популяция состоит из *NP* агентов. Начальное распределение на множестве допустимых решений – равномерное.

В качестве начальных условий движения агента можно взять случайное начальное положение на множестве допустимых решений и нулевую скорость.

Второй этап. Последовательная миграция в направлении абсолютного лидера – вожака стаи.

Среди агентов текущей популяции находится лидер с положением x^{best} (предположительно, нашел источник пищи). Остальные агенты последовательно сдвигаются к лидеру, запоминая свою наилучшую позицию за время движения. Предлагается для всех агентов использовать управление с прогнозирующей моделью (Model Predictive Control, MPC).

При этом задается N_p – горизонт прогнозирования, N_c – горизонт управления, $N_c \le N_p$ – количество дискретных моментов времени, на которых реализуется прогнозирование движения модели объекта и управление.

Пусть фиксирован текущий момент дискретного времени k_i, тогда согласно дискретной модели движения агента получаем

$$\begin{aligned} &x(k_i+1) = Ax(k_i) + Bu(k_i), \\ &x(k_i+2) = Ax(k_i+1) + Bu(k_i+1) = A^2x(k_i) + ABu(k_i) + Bu(k_i+1), \\ &\vdots \\ &x(k_i+N_p) = A^{N_p}x(k_i) + A^{N_p-1}Bu(k_i) + \ldots + A^{N_p-N_c}Bu(k_i+N_c-1). \end{aligned} \tag{1.30}$$

Здесь имеется в виду, что в дискретные моменты времени, расположенные за горизонтом управления, заданном параметром N_c, управление движением отсутствует (равно нулю):

$$x(k_i) = \begin{pmatrix} x_1(k_i) \\ x_2(k_i) \\ \vdots \\ x_n(k_i) \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Введем обозначения:

$U = \begin{pmatrix} u(k_i) \\ u(k_i+1) \\ \vdots \\ u(k_i+N_c-1) \end{pmatrix}$ – вектор управлений на горизонте управления размеров $(n \cdot N_c \times 1)$;

$X = \begin{pmatrix} x(k_i+1) \\ x(k_i+2) \\ \vdots \\ x(k_i+N_p) \end{pmatrix}$ – расширенный вектор, состоящий из векторов состояния на горизонте прогнозирования, размеров $(2n \cdot N_p) \times 1$;

$R = \begin{pmatrix} r(k_i) \\ \vdots \\ r(k_i) \end{pmatrix} \Big\} N_p$ – вектор требуемых значений координат вектора состояния (в данном случае постоянных, заданных в момент k_i и определяемых положением лидера и его скоростью) размеров $(2n \cdot N_p) \times 1$,

$X(k_i) = \begin{pmatrix} x(k_i) \\ x(k_i) \\ \vdots \\ x(k_i) \end{pmatrix} \Big\} N_p$ – вектор, определяемый начальным состоянием согласно (1.30) размеров $(2n \cdot N_p) \times 1$.

Определим функционал качества управления движением каждого агента:

$$I = (R - X)^T (R - X) + U^T Q U \to \min_U,$$

где $Q = qE_{n \cdot N_c}$ – диагональная матрица с коэффициентом $q \geq 0$. Он отражает степень близости члена стаи к лидеру на всем горизонте управления, а также интенсивность прикладываемого управления.

Требуется найти такое управление агентом, которое минимизирует величину критерия. Оно используется в (1.27)–(1.29) для получения траекторий движения.

С учетом введенных обозначений перепишем модель движения

$$X = FX(k_i) + \Phi U,$$

где $F = \begin{pmatrix} A & 0 & \cdots & 0 \\ 0 & A^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A^{N_p} \end{pmatrix}$ – матрица размеров $(2n \cdot N_p) \times (2n \cdot N_p)$,

$\Phi = \begin{pmatrix} B & 0 & 0 & \cdots & 0 \\ AB & B & 0 & \cdots & 0 \\ A^2B & AB & B & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ A^{N_p-1}B & A^{N_p-2}B & A^{N_p-3}B & \cdots & A^{N_p-N_c}B \end{pmatrix}$ – матрица размеров $(2n \cdot N_p) \times (n \cdot N_c)$,

и функционал

$$I = (R - FX(k_i) - \Phi U)^T (R - FX(k_i) - \Phi U) + U^T Q U =$$
$$= (R - FX(k_i))^T (R - FX(k_i)) + U^T \Phi^T \Phi U + U^T Q U - 2(R - FX(k_i))^T \Phi U.$$

Учитывая, что $\frac{\partial Au}{\partial u} = A^T, \frac{\partial (uAu)}{\partial u} = A + A^T$, применяем необходимые условия экстремума:

$$\frac{\partial I}{\partial U} = 0 \Leftrightarrow 2(\Phi^T \Phi + Q)U - 2\Phi^T (R - FX(k_i)) = 0.$$

Отсюда находим

$$U^* = (\Phi^T \Phi + Q)^{-1} \Phi^T (R - FX(k_i)). \quad (1.31)$$

Найденное управление применяется в течение горизонта управления N_c ко всем агентам, в качестве $X(k_i)$ для каждого из них берется его начальное положение и скорость.

После выполнения N_p итераций требуется найти нового лидера и взять его положение x^{best} и скорость (можно положить нулевой) в качестве компонент $r(k_i)$.

Требуется реализовать заданное число проходов по N_p итераций в каждом. В результате находится наилучшее решение – лидер и наилучшие положения агентов на протяжении выполненных проходов. Положение лидера принимается за новый источник пищи (новую жертву), полагая, что лидер нашел его, и некоторый промежуток времени будет оставаться неподвижным, используя источник.

Третий этап. Интенсивная миграция – из ближайших к лидеру по достигнутой величине целевой функции, но отстоящих от него по расстоянию не менее, чем на пороговую величину $dist$, выбираются еще два агента. Они атакуют жертву (присоединяются к лидеру), используя в качестве управления алгоритм с прогнозирующей моделью, при реализации которого на основании информации о положении лидера совершают только одну итерацию на горизонте управления, после которой обмениваются информацией и оперативно пересчитывают законы управления. В процессе их движения к цели запоминаются наилучшие положения. В результате считается, что окрестность цели достаточно хорошо исследуется. Найденное наилучшее решение (положение лидера) помещается в множество *Pool*. Два агента делают по одной итерации на основе текущей информации о лидере и своем положении. При этом используется управление $u(k_i) = \underbrace{(100...0)}_{N_c} U$, т.е. из найденной последовательности управлений с заданными горизонтами прогнозирования и управления берется только первое (применяется принцип отступающего горизонта на один такт дискретного времени). Затем находятся новые положения агентов, среди них новый лидер и соответствующие управления каждым агентом. Процедура выполняется заданное число раз. В итоге положение лидера запоминается в множестве *Pool*.

Четвертый этап. Импульсная миграция с оперативным обменом информацией (поиск новой жертвы, источника пищи).

Продолжается процесс миграции на множестве допустимых решений всей популяцией агентов с целью поиска нового источника пищи. Для этого последовательно применяется алгоритм с прогнозирующей моделью, при реализации которого каждый

агент совершает одно и то же фиксированное случайное количество итераций, ограниченное горизонтом прогнозирования. По окончании очередного импульса миграции все агенты обмениваются информацией и оперативно пересчитывают законы управления. Выполняется заданное число импульсов.

Пятый этап. Обновление популяции: удаляются $\sigma\%$ агентов, наихудших по величине целевой функции, а среди каждых двух членов популяции из числа оставшихся после удаления, отстоящих друг от друга менее чем на заданную величину, сохраняется лучший по величине целевой функции. Вместо выбывших агентов генерируются столько же новых на множестве допустимых решений. Далее происходит новая реализация последовательной миграции (второго этапа).

После реализации максимального общего числа итераций в качестве решения поставленной задачи из множества *Pool* выбирается решение, соответствующее абсолютному лидеру.

1.8.2. Алгоритм решения задачи

Шаг 1. *Создание начальной популяции.*

Шаг 1.1. Задать параметры метода: NP – размер популяции, $ITER_{\max}$ – максимальное число итераций, h – величина шага интегрирования, $N_{\max}$ – максимальное число моментов времени, $dist$ – пороговая величина расстояния между агентами, N_p – горизонт прогнозирования, N_c – горизонт управления ($N_c \le N_p$), $\sigma\%$ – процент удаляемых из популяции агентов, q – коэффициент штрафа за использование управления.

Шаг 1.2. Сгенерировать начальную популяцию агентов I_0, для которой хранится информация о положениях агентов и их скоростях. Для формирования начального положения с помощью равномерного распределения на единичном отрезке $[0;1]$ NP раз генерировать последовательность из n случайных точек $\{P_i^{0,j}\}_{i=1}^{n}$, $i=1,...,n$, $j=1,2,...,NP$. Используя линейное преобразование, каждую точку отобразить на соответствующий ей промежуток $[a_i,b_i]$: $P_i^j=(b_i-a_i)P_i^{0,j}+a_i$. Составляя векторы из точек последовательности $\{P_i^j\}_{i=1}^{n}$ при фиксированных j, получить NP начальных векторов $x^j=(x_1^j,x_2^j,...,x_n^j)^T$, $x_i^j=P_i^j$, $i=1,2,...,n$, координаты которых x_i имеют равномерное распределение на отрезках $[a_i,b_i]$, $i=1,...,n$. Вектор скоростей для каждого агента на начальной итерации считается нулевым. Таким образом, сформирована начальная популяция вида

$$I_0=\{x^{j,0}, j=1,2,...,NP \mid x^{j,0}=(x_1^j,x_2^j,...,x_{2n}^j)^T=(x_1^j,x_2^j,...,x_n^j,0,0,...,0)^T\}.$$

Для всех NP агентов подсчитать значение целевой функции f, отсортировать полученные значения в порядке возрастания. Лидером стаи считается агент с наименьшим значением целевой функции. Положение лидера $x^{best,0}$ и соответствующее значение целевой функции занести в множество $Pool$.

Шаг 1.3. Определить матрицы, входящие в уравнения движения, в зависимости от используемой модели движения агентов:

первая модель: $A=\begin{pmatrix} E_n & hE_n \\ O_n & E_n \end{pmatrix}$, $B=\begin{pmatrix} \frac{h^2}{2}E_n \\ hE_n \end{pmatrix}$;

вторая модель: $A=\begin{pmatrix} E_n & hE_n \\ O_n & E_n \end{pmatrix}, B=\begin{pmatrix} O_n \\ hE_n \end{pmatrix};$

третья модель: $A=\begin{pmatrix} O_n & E_n \\ -E_n & 2E_n \end{pmatrix}, B=\begin{pmatrix} O_n \\ h^2E_n \end{pmatrix},$

а также матрицы

$$Q=qE_{n\cdot N_c},\ F=\begin{pmatrix} A & 0 & \cdots & 0 \\ 0 & A^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A^{N_p} \end{pmatrix},\ \Phi=\begin{pmatrix} B & 0 & 0 & \cdots & 0 \\ AB & B & 0 & \cdots & 0 \\ A^2B & AB & B & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A^{N_p-1}B & A^{N_p-2}B & A^{N_p-3}B & \cdots & A^{N_p-N_c}B \end{pmatrix}.$$

Положить $k_i=0$, $i=0$.

Шаг 2. *Последовательная миграция на множестве допустимых решений.*

Шаг 2.1. Выполнить для каждого из NP агентов за исключением агента- лидера ($j\in\{1,...,NP\}$) следующие действия.

Составить вектор $R=(\underbrace{r(k_i)\quad r(k_i)\quad \dots\quad r(k_i)}_{N_p})^T$, где $r(k_i)=(x_1^{best,k},...,x_{2n}^{best,k})^T$ – вектор, содержащий информацию о положении и скорости агента-лидера в момент времени k_i.

Для выбранного агента составить вектор $X(k_i)=(\underbrace{x(k_i)\quad x(k_i)\quad \dots\quad x(k_i)}_{N_p})^T$, где вектор $x(k_i)=(x_1^{j,k}(k_i),...,x_{2n}^{j,k}(k_i))^T$ содержит информацию о его начальном положении и скорости в момент времени k_i.

Вычислить вектор управления $U=(u^{(j)}(k_i),u^{(j)}(k_i+1),...,u^{(j)}(k_i+N_c-1))^T$ на горизонте управления по формуле (1.31):

$$U=(\Phi^T\Phi+Q)^{-1}\Phi^T(R-FX(k_i)).$$

Вычислить расширенный вектор $X=(x^{(j)}(k_i+1),x^{(j)}(k_i+2),...,x^{(j)}(k_i+N_p))^T$, состоящий из векторов состояния на горизонте прогнозирования по формулам

$$x^{(j)}(k_i+1)=Ax^{(j)}(k_i)+Bu^{(j)}(k_i),$$
$$x^{(j)}(k_i+2)=A^2x^{(j)}(k_i)+ABu^{(j)}(k_i)+Bu^{(j)}(k_i+1),$$
$$\vdots$$
$$x^{(j)}(k_i+N_p)=A^{N_p}x^{(j)}(k_i)+A^{N_p-1}Bu^{(j)}(k_i)+...+A^{N_p-N_c}Bu^{(j)}(k_i+N_c-1).$$

Для всех моментов времени $i=k_i+1,...,k_i+N_p$ вычислить значение целевой функции и выбрать среди них наилучшее.

Шаг 2.2. Среди лучших положений, достигнутых агентами в процессе миграции, и положения текущего агента-лидера найти нового лидера $x^{best,k}$.

Если $k<N_{\max}$, положить $k_i=k_i+N_p$, $k=k+1$ и перейти к шагу 2.1. Иначе перейти к шагу 3.

Шаг 3. *Интенсивная миграция.*

Шаг 3.1. Упорядочить популяцию агентов по возрастанию значений целевой функции. Среди выявленных лидеров популяции выбрать трех агентов, наиболее близких по величине целевой функции к лидеру (цели), но отстоящих от лидера не менее, чем на величину $dist$. Будем считать положение агента-лидера в течение этого шага неподвижным: $x^{best} = (x_1^{best},...,x_n^{best},0,...,0)^T$, где $x_1^{best} = x_1^{best,k},...,x_n^{best} = x_n^{best,k}$.

Шаг 3.2**.** Для двух агентов из числа отобранных на шаге 3.1 использовать управление $u(k_i) = \underbrace{(100...0)}_{N_c} U$, т.е. из найденной последовательности управлений с заданными горизонтами прогнозирования и управления берется только первое. Затем найти новые положения агентов.

Процедуры, описанные на шагах 3.1–3.2, выполняются заданное число раз. В итоге положение агента-лидера запоминается в множестве *Pool*.

Шаг 4. *Импульсная миграция с оперативным обменом информацией* (*поиск новой жертвы, источника пищи*).

Выбрать случайное число $N_c \in [1,...,N_p]$.

Для каждого агента сделать по N_c итераций по формулам из шага 2.1 на основании информации о лидере и своем положении.

Выполнить описанные действия заданное число раз.

Для новых положений агентов вычислять значения целевой функции, ранжировать популяцию по возрастанию значений целевой функции, найти лидера, занести в множество *Pool*.

Шаг 5. *Обновление популяции.*

Из отсортированной по возрастанию популяции удалить σ% агентов с наихудшим значением целевой функции. Среди двух членов популяции, отстоящих друг от друга менее, чем на заданную величину $dist$, остается лучший по величине целевой функции. Вместо выбывших агентов генерируются столько же новых на множестве допустимых решений. Новые члены популяции генерируются аналогично созданию начальной популяции на шаге 1.2.

Шаг 6. *Завершение процесса поиска решения.*

Если достигнуто предельное число итераций, в качестве решения поставленной задачи из множества *Pool* выбрать наилучшее решение, соответствующее абсолютному лидеру. Иначе – перейти к шагу 2.

1.8.3. Анализ эффективности метода

АНАЛИЗ РАБОТЫ МЕТОДА ПРИ РАЗЛИЧНЫХ ЗНАЧЕНИЯХ ПАРАМЕТРОВ

При исследовании работы алгоритма проводились серии из 100 решений одной задачи и подсчитывались выборочное среднее $\overline{f}$, наилучшее значение целевой функции f_{best} и среднеквадратичное отклонение $\overline{\sigma_f}$.

Пример 1.17. Найдем глобальный минимум функции Экли (табл. П.1)

$$f(x) = e - 20\exp\left(-\sqrt{\frac{x_1^2 + x_2^2}{50}}\right) - \exp\left(\frac{1}{2}\left(\cos(2\pi x_1) + \cos(2\pi x_2)\right)\right)$$

на множестве допустимых решений $x, y \in [-10;10]$. Глобальный минимум: $x^* = (0;0)^T$, $f(x^*) = -20$. Найденное наилучшее приближение совпадает с точным решением.

Таблица 1.29. Первая модель движения агентов

Параметры									$\overline{f}$	f_{best}	$\overline{\sigma}_f$
NP	$ITER_{\max}$	h	$N_{\max}$	$dist$	N_p	N_c	σ	q			
10	100	0,5	20	0,1	10	5	0,3	1	–18,8856	–20	3,780791
20	90	0,6	20	0,1	11	6	0,3	2	–19,6815	–20	0,223145
30	80	0,7	20	0,2	12	6	0,3	3	–19,3915	–20	1,508544
40	70	0,8	30	0,2	13	7	0,3	1	–17,1265	–19,9993	10,15107
50	65	0,4	30	0,3	10	4	0,3	2	–19,2415	–20	1,616589
60	60	0,5	30	0,35	11	5	0,4	3	–19,7489	–20	0,233276
70	50	0,6	25	0,4	12	5	0,4	1	–8,94186	–19,9967	136,081
80	45	0,7	25	0,1	13	6	0,4	2	–14,762	–19,9997	30,9844
90	40	0,8	25	0,1	10	10	0,4	3	–17,3167	–19,8731	8,06954
100	35	0,4	35	0,2	11	11	0,4	1	–17,4526	–20	7,730111
110	30	0,5	35	0,2	12	12	0,5	2	–17,1661	–20	8,879618
120	30	0,6	35	0,3	13	13	0,5	3	–17,5911	–20	6,845631
130	30	0,7	20	0,3	10	2	0,5	1	–0,60087	–17,7349	394,8655
140	30	0,8	20	0,1	11	3	0,5	2	0,145845	–20	425,9163
150	30	0,5	30	0,1	12	4	0,5	3	–19,7344	–20	0,697989

Таблица 1.30. Вторая модель движения агентов

Параметры									$\overline{f}$	f_{best}	$\overline{\sigma}_f$
NP	$ITER_{\max}$	h	$N_{\max}$	$dist$	N_p	N_c	σ	q			
10	100	0,5	20	0,1	10	5	0,3	1	–19,1518	–20	2,645867
20	90	0,6	20	0,1	11	6	0,3	2	–19,7214	–20	0,327668
30	80	0,7	20	0,2	12	6	0,3	3	–19,9184	–20	0,145509
40	70	0,8	30	0,2	13	7	0,3	1	–17,9881	–20	7,038006
50	65	0,4	30	0,3	10	4	0,3	2	–19,5505	–20	0,767505
60	60	0,5	30	0,35	11	5	0,4	3	–19,8618	–20	0,208367
70	50	0,6	25	0,4	12	5	0,4	1	–14,3193	–20	41,77298
80	45	0,7	25	0,1	13	6	0,4	2	–18,6814	–20	3,344124
90	40	0,8	25	0,1	10	10	0,4	3	–17,1583	–19,9917	9,001067
100	35	0,4	35	0,2	11	11	0,4	1	–17,2612	–19,7895	8,472244
110	30	0,5	35	0,2	12	12	0,5	2	–17,3114	–20	8,128291
120	30	0,6	35	0,3	13	13	0,5	3	–17,7652	–20	6,431938
130	30	0,7	20	0,3	10	2	0,5	1	–1,351	–16,5703	371,4222
140	30	0,8	20	0,1	11	3	0,5	2	1,151511	–20	460,2673
150	30	0,5	30	0,1	12	4	0,5	3	–19,9941	–20	0,00099

Таблица 1.31. Третья модель движения агентов

Параметры									$\overline{f}$	f_{best}	$\overline{\sigma}_f$
NP	$ITER_{\max}$	h	$N_{\max}$	$dist$	N_p	N_c	σ	q			
10	100	0,5	20	0,1	10	5	0,3	1	–17,4653	–19,4821	7,471045
20	90	0,6	20	0,1	11	6	0,3	2	–18,0724	–19,7618	4,522834
30	80	0,7	20	0,2	12	6	0,3	3	–17,4094	–19,8799	8,091627
40	70	0,8	30	0,2	13	7	0,3	1	–6,71915	–19,9781	266,024
50	65	0,4	30	0,3	10	4	0,3	2	–17,4636	–19,9994	10,7018
60	60	0,5	30	0,35	11	5	0,4	3	–19,4938	–20	1,075073
70	50	0,6	25	0,4	12	5	0,4	1	–2,24629	–19,7541	340,0142
80	45	0,7	25	0,1	13	6	0,4	2	–6,83847	–19,8282	242,7111
90	40	0,8	25	0,1	10	10	0,4	3	–17,2067	–19,5486	8,800435
100	35	0,4	35	0,2	11	11	0,4	1	–17,2064	–19,9882	8,780311
110	30	0,5	35	0,2	12	12	0,5	2	–17,4969	–19,8081	7,242562
120	30	0,6	35	0,3	13	13	0,5	3	–17,4818	–20	7,3182
130	30	0,7	20	0,3	10	2	0,5	1	1,175164	–16,4183	458,069
140	30	0,8	20	0,1	11	3	0,5	2	1,761964	–13,8916	477,1259
150	30	0,5	30	0,1	12	4	0,5	3	–19,214	–20	2,154719

Пример 1.18. Найдем глобальный минимум функции Изома

$$f(x) = -\cos x_1 \cos x_2 \exp\{-[(x_1-\pi)^2 + (x_2-\pi)^2]\}$$

на множестве допустимых решений $x_1, x_2 \in [-100;100]$. Глобальный минимум: $x^* = (\pi;\pi)^T$, $f(x^*) = -1$. Лучшее приближение: значение целевой функции: -1, решение $x^* = (\pi;\pi)^T$. Результаты для трех моделей движения агентов приведены в табл. 1.32–1.34.

Таблица 1.32. Первая модель движения агентов

Параметры									$\overline{f}$	f_{best}	$\overline{\sigma}_f$
NP	$ITER_{\max}$	h	$N_{\max}$	$dist$	N_p	N_c	σ	q			
10	100	0,5	20	0,1	10	5	0,3	1	–0,20776	–1	0,773907
20	90	0,6	20	0,1	11	6	0,3	2	–0,28709	–1	0,660826
30	80	0,7	20	0,2	12	6	0,3	3	–0,38526	–1	0,570336
40	70	0,8	30	0,2	13	7	0,3	1	–0,02336	–0,99992	0,972562
50	65	0,4	30	0,3	10	4	0,3	2	–0,59828	–1	0,364259
60	60	0,5	30	0,35	11	5	0,4	3	–0,41325	–1	0,509474
70	50	0,6	25	0,4	12	5	0,4	1	–0,00704	–0,70433	0,990874
80	45	0,7	25	0,1	13	6	0,4	2	–0,04253	–1	0,950074
90	40	0,8	25	0,1	10	10	0,4	3	–0,82907	–1	0,122959
100	35	0,4	35	0,2	11	11	0,4	1	–0,87235	–1	0,062183
110	30	0,5	35	0,2	12	12	0,5	2	–0,87285	–1	0,07357
120	30	0,6	35	0,3	13	13	0,5	3	–0,84612	–1	0,092484
130	30	0,7	20	0,3	10	2	0,5	1	0	0	1
140	30	0,8	20	0,1	11	3	0,5	2	–0,10511	–1	0,88399
150	30	0,5	30	0,1	12	4	0,5	3	–0,99639	–1	0,000169

Таблица 1.33. Вторая модель движения агентов

Параметры									$\overline{f}$	f_{best}	$\overline{\sigma}_f$
NP	$ITER_{\max}$	h	$N_{\max}$	$dist$	N_p	N_c	σ	q			
10	100	0,5	20	0,1	10	5	0,3	1	–0,27045	–1	0,716103
20	90	0,6	20	0,1	11	6	0,3	2	–0,36769	–1	0,602725
30	80	0,7	20	0,2	12	6	0,3	3	–0,59179	–1	0,383251
40	70	0,8	30	0,2	13	7	0,3	1	–0,14002	–0,99995	0,851418
50	65	0,4	30	0,3	10	4	0,3	2	–0,84026	–1	0,122781
60	60	0,5	30	0,35	11	5	0,4	3	–0,58172	–1	0,371125
70	50	0,6	25	0,4	12	5	0,4	1	–0,06987	–1	0,930001
80	45	0,7	25	0,1	13	6	0,4	2	–0,07557	–1	0,920086
90	40	0,8	25	0,1	10	10	0,4	3	–0,74541	–1	0,207664
100	35	0,4	35	0,2	11	11	0,4	1	–0,81867	–1	0,096666
110	30	0,5	35	0,2	12	12	0,5	2	–0,83637	–1	0,107141
120	30	0,6	35	0,3	13	13	0,5	3	–0,8841	–1	0,069201
130	30	0,7	20	0,3	10	2	0,5	1	0	0	1
140	30	0,8	20	0,1	11	3	0,5	2	$-2{,}8\cdot10^{-88}$	$-2{,}8\cdot10^{-86}$	1
150	30	0,5	30	0,1	12	4	0,5	3	–0,99316	–1	0,001093

Таблица 1.34. Третья модель движения агентов

Параметры									$\overline{f}$	f_{best}	$\overline{\sigma}_f$
NP	$ITER_{\max}$	h	$N_{\max}$	$dist$	N_p	N_c	σ	q			
10	100	0,5	20	0,1	10	5	0,3	1	–0,33101	–0,99933	0,612457
20	90	0,6	20	0,1	11	6	0,3	2	–0,22987	–0,99984	0,744108
30	80	0,7	20	0,2	12	6	0,3	3	–0,51851	–0,99928	0,357502
40	70	0,8	30	0,2	13	7	0,3	1	$-2\cdot10^{-7}$	$-1{,}2\cdot10^{-5}$	1
50	65	0,4	30	0,3	10	4	0,3	2	–0,6647	–1	0,307464
60	60	0,5	30	0,35	11	5	0,4	3	–0,77337	–1	0,203337
70	50	0,6	25	0,4	12	5	0,4	1	–0,02834	–0,98492	0,970136
80	45	0,7	25	0,1	13	6	0,4	2	–0,02948	–0,99836	0,970016
90	40	0,8	25	0,1	10	10	0,4	3	–0,74897	–1	0,208049
100	35	0,4	35	0,2	11	11	0,4	1	–0,81187	–1	0,104144
110	30	0,5	35	0,2	12	12	0,5	2	–0,88406	–1	0,07459
120	30	0,6	35	0,3	13	13	0,5	3	–0,80217	–1	0,148707
130	30	0,7	20	0,3	10	2	0,5	1	–0,02686	–0,99971	0,970767
140	30	0,8	20	0,1	11	3	0,5	2	–0,01975	–0,99548	0,980004
150	30	0,5	30	0,1	12	4	0,5	3	–0,50254	–1	0,481447

Пример 1.19. Найдем глобальный минимум функции Гольдшейна–Прайса

$$f(x) = [1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times$$

$$\times [30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$$

на множестве допустимых решений $x_1, x_2 \in [-2; 2]$. Глобальный минимум: $x^* = (0; -1)^T$, $f(x^*) = 3$. Лучшее найденное приближение: значение целевой функции: 3, $x^* = (0; -1)^T$.

Результаты, полученные для трех моделей движения агентов, представлены в табл. 1.35–1.37.

Таблица 1.35. Первая модель движения агентов

Параметры									$\overline{f}$	f_{best}	$\overline{\sigma}_f$
NP	$ITER_{\max}$	h	$N_{\max}$	$dist$	N_p	N_c	σ	q			
10	100	0,5	20	0,1	10	5	0,3	1	161,759	3,000005	93701,76
20	90	0,6	20	0,1	11	6	0,3	2	7,893574	3	959,0339
30	80	0,7	20	0,2	12	6	0,3	3	11,33477	3	443,4577
40	70	0,8	30	0,2	13	7	0,3	1	37,36356	3,000548	2428,974
50	65	0,4	30	0,3	10	4	0,3	2	13,54685	3	657,1989
60	60	0,5	30	0,35	11	5	0,4	3	8,793712	3	163,9521
70	50	0,6	25	0,4	12	5	0,4	1	$1{,}19\cdot10^{8}$	3	$2{,}58\cdot10^{17}$
80	45	0,7	25	0,1	13	6	0,4	2	1014,839	3	19228441
90	40	0,8	25	0,1	10	10	0,4	3	23,16321	3,137124	1083,795
100	35	0,4	35	0,2	11	11	0,4	1	23,71029	3,222247	868,3535
110	30	0,5	35	0,2	12	12	0,5	2	22,4876	3,120264	764,599
120	30	0,6	35	0,3	13	13	0,5	3	19,57538	3,024393	554,2703
130	30	0,7	20	0,3	10	2	0,5	1	$1{,}75\cdot10^{23}$	6,602183	$1{,}52\cdot10^{48}$
140	30	0,8	20	0,1	11	3	0,5	2	$2{,}62\cdot10^{19}$	3,04952	$1{,}28\cdot10^{39}$
150	30	0,5	30	0,1	12	4	0,5	3	4,621014	3	43,75986

Таблица 1.36. Вторая модель движения агентов

Параметры									$\overline{f}$	f_{best}	$\overline{\sigma}_f$
NP	$ITER_{\max}$	h	$N_{\max}$	$dist$	N_p	N_c	σ	q			
10	100	0,5	20	0,1	10	5	0,3	1	123,5038	3	62718,8
20	90	0,6	20	0,1	11	6	0,3	2	3,599247	3	3,774223
30	80	0,7	20	0,2	12	6	0,3	3	6,492877	3	158,9667
40	70	0,8	30	0,2	13	7	0,3	1	29,37367	3,000003	1869,328
50	65	0,4	30	0,3	10	4	0,3	2	8,587825	3	262,0077
60	60	0,5	30	0,35	11	5	0,4	3	8,230652	3	113,2233
70	50	0,6	25	0,4	12	5	0,4	1	16038,34	3	$1{,}2\cdot10^{10}$
80	45	0,7	25	0,1	13	6	0,4	2	56947,02	3	$3{,}2\cdot10^{11}$
90	40	0,8	25	0,1	10	10	0,4	3	26,22964	3,053785	961,7849
100	35	0,4	35	0,2	11	11	0,4	1	23,60595	3,281163	712,5785
110	30	0,5	35	0,2	12	12	0,5	2	21,10988	3,63461	521,1074
120	30	0,6	35	0,3	13	13	0,5	3	21,32103	3,014493	606,8074
130	30	0,7	20	0,3	10	2	0,5	1	$1{,}59\cdot10^{14}$	12,63668	$2{,}54\cdot10^{30}$
140	30	0,8	20	0,1	11	3	0,5	2	$6{,}46\cdot10^{16}$	782174,7	$8{,}14\cdot10^{33}$
150	30	0,5	30	0,1	12	4	0,5	3	4,368281	3	37,2014

Таблица 1.37. Третья модель движения агентов

Параметры									$\bar{f}$	f_{best}	$\bar{\sigma}_f$
NP	$ITER_{\max}$	h	$N_{\max}$	$dist$	N_p	N_c	σ	q			
10	100	0,5	20	0,1	10	5	0,3	1	106,8533	3,070495	28821,97
20	90	0,6	20	0,1	11	6	0,3	2	73,75066	3,031026	24712,03
30	80	0,7	20	0,2	12	6	0,3	3	$6,83\cdot10^{8}$	3,051269	$4,63\cdot10^{19}$
50	65	0,4	30	0,3	10	4	0,3	2	$3,79\cdot10^{15}$	3,000558	$1,44\cdot10^{33}$
60	60	0,5	30	0,35	11	5	0,4	3	106,0867	3,001057	38394,41
80	45	0,7	25	0,1	13	6	0,4	2	$3\cdot10^{37}$	3,024332	$9,02\cdot10^{76}$
90	40	0,8	25	0,1	10	10	0,4	3	21,41323	3,128469	642,0922
100	35	0,4	35	0,2	11	11	0,4	1	24,70943	3,07662	895,4431
110	30	0,5	35	0,2	12	12	0,5	2	18,2086	3,094909	430,8168
120	30	0,6	35	0,3	13	13	0,5	3	17,73205	3,34423	399,7322
140	30	0,8	20	0,1	11	3	0,5	2	$2,18\cdot10^{41}$	3,000746	$4,33\cdot10^{83}$
150	30	0,5	30	0,1	12	4	0,5	3	49710975	3,000133	$3,2\cdot10^{16}$
10	100	0,5	20	0,1	10	5	0,3	1	106,8533	3,070495	28821,97
20	90	0,6	20	0,1	11	6	0,3	2	73,75066	3,031026	24712,03
30	80	0,7	20	0,2	12	6	0,3	3	$6,83\cdot10^{8}$	3,051269	$4,63\cdot10^{19}$

Применение первой модели движения агентов приводит к тому, что результаты минимизации двух из трех исследованных функций содержат наилучшие или близкие к наилучшим значения целевой функции. Вторая модель движения агентов позволяет найти наилучшие или близкие к наилучшим значения целевой функции во всех трех примерах. Третья модель движения уступает первым двум. Поэтому можно сделать вывод о предпочтительности использования второй модели.

РЕКОМЕНДАЦИИ ПО ВЫБОРУ ПАРАМЕТРОВ

Размер популяции NP определяет количество вычислений целевой функции на каждой итерации. Чем больше параметр размера популяции NP, тем лучше полученный результат, но в силу возрастающих вычислительных затрат замедляется алгоритм. Рекомендуемое значение $NP \geq 20$.

Число итераций $ITER_{\max}$ определяет, как долго будет продолжаться поиск новых решений. При увеличении $ITER_{\max}$ точность решения увеличивается. Для рассмотренного набора стандартных функций рекомендуемые значения в зависимости от сложности функции $ITER_{\max} \in [20;80]$.

Величина шага интегрирования h моделей движения агентов. Рекомендуемое значение $h = 0,5$.

Максимальное число моментов времени $N_{\max} \in [20;35]$.

Пороговое расстояние между агентами $dist$. Рекомендуемое значение 0,1.

Горизонт прогнозирования N_p. Обычно принимают $N_p \in \{10,11,12\}$.

Горизонт управления N_c ($N_c \leq N_p$). Рекомендуемые значения $N_c \in \{4,5,6\}$.

Процент удаляемых из популяции агентов $\sigma\%$. Рекомендуемое значение от 30% до 50%.

Коэффициент штрафа за использование управления $q \in [2;3]$.

1.9. ПРИМЕНЕНИЕ МУЛЬТИАГЕНТНЫХ МЕТОДОВ В ЗАДАЧАХ ПАРАМЕТРИЧЕСКОЙ ОПТИМИЗАЦИИ ТЕХНИЧЕСКИХ СИСТЕМ

1.9.1. Постановка задачи и стратегия поиска решения

В разд. 1.1 сформулирована постановка задачи поиска условного экстремума при параллелепипедных ограничениях, когда предполагается, что значения каждой независимой переменной принадлежит заданному отрезку. Однако в прикладных задачах помимо параллелепипедных могут быть наложены ограничения типа равенств или неравенств. Рассмотрим часто встречающуюся задачу оптимизации.

Постановка задачи. Дана целевая функция

$$f(x) = f(x_1, \ldots, x_n),$$

определенная на множестве допустимых решений $D \subseteq R^n$ и удовлетворяющая ограничениям

$$g^1(x) \le 0, \ldots, g^m(x) \le 0,$$

где $D = \{x \mid x_i \in [a_i, b_i],\ i = 1, 2, \ldots, n\}$, число m ограничений типа неравенств задано.

Предполагается, что целевая функция и функции, задающие ограничения, непрерывны.

В рассматриваемых далее прикладных задачах параметрами системы x_i являются характерные параметры конструкций, ограничения вида $g^1(x) \le 0, \ldots, g^m(x) \le 0$ характеризуют физические свойства конструкции (например, ограничения по напряжению на отдельные детали), а целевая функция $f(x)$ характеризует стоимость или вес конструкции.

Требуется минимизировать значение целевой функции $f(x)$ с учетом выполнения всех заданных ограничений.

Стратегия поиска решения. Заключается в переходе от исходной задачи к задаче поиска глобального экстремума вспомогательной функции $F(x_1, \ldots, x_n)$, полученной с помощью применения метода внешних штрафов [3, 4, 46]. При этом функция $F(x_1, \ldots, x_n)$ принимает вид:

$$F(x, c) = f(x) + \sum_{i=1}^{m} c_i \cdot [\max\{0,\ g^i(x)\}]^2 \to \min_{x \in D},$$

где $c_1, \ldots, c_m$ – коэффициенты штрафной функции, которые подбираются так, чтобы получить решение достаточно хорошего качества. Они играют роль весовых коэффициентов, поскольку определяют важность вклада каждого из слагаемых, характеризующих величину штрафа за невыполнение соответствующего ограничения. Кроме того, решается проблема приведения всех величин, участвуюших в задаче, к одной размерности.

Полученную задачу можно решить рассмотренными в первой главе мультиагентными метаэвристическими методами нахождения глобального условного минимума с применением метода внешних штрафов.

1.9.2. Задача определения параметров сварной балки

В рассматриваемой задаче [41, 74, 108] требуется определить параметры конструкции сварной балки, исходя из имеющихся ограничений (рис. 1.70).

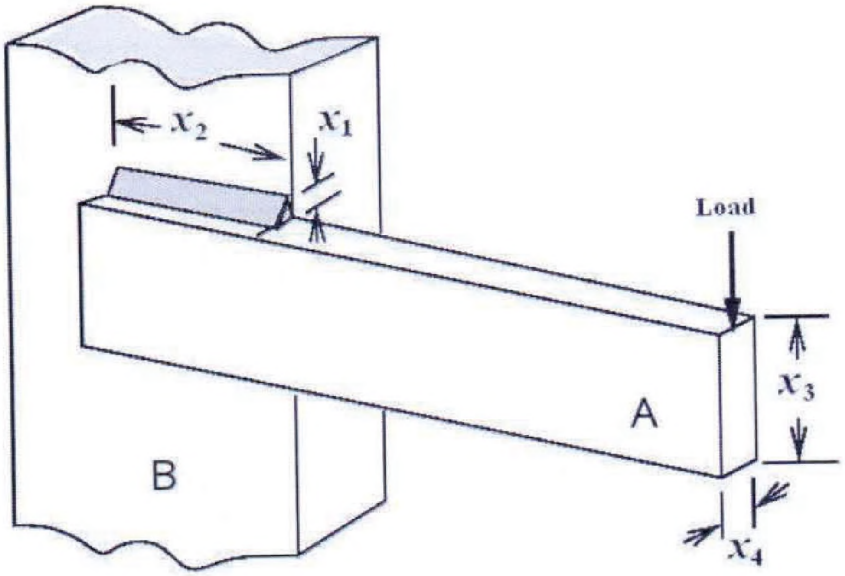


Рис. 1.70. Сварная балка

Рассматриваемая балка состоит из балки A и сварки, необходимой для ее прикрепления к балке B. Целью является определение минимальной по стоимости конструкции балки, заключающейся в определении вектора параметров $x = (x_1, x_2, x_3, x_4)^T$, где x_1 – высота сварного шва, x_2 – длина сварного шва, x_3 – высота балки, x_4 – ширина балки. Кроме того, балка должна удовлетворять ограничениям по напряжению сдвига τ, изгиба σ, продольной нагрузке P_c и отклонению края балки δ.

Задача может быть формализована следующим образом:

$$f(x) = 1{,}10471 \cdot x_1^2 \cdot x_2 + 0{,}04811 \cdot x_3 \cdot x_4 \cdot (14 + x_2),$$

$$g^1(x) = \tau(x) - 13600 \le 0, \quad g^2(x) = \sigma(x) - 30000 \le 0, \quad g^3(x) = x_1 - x_4 \le 0,$$

$$g^4(x) = 0{,}10471 \cdot x_1^2 + 0{,}04811 \cdot x_3 \cdot x_4 \cdot (14 + x_2) - 5 \le 0,$$

$$g^5(x) = 0{,}125 - x_1 \le 0, \quad g^6(x) = \delta(x) - 0{,}25 \le 0, \quad g^7(x) = 6000 - P_c(x) \le 0,$$

$$D = [0{,}1; 2{,}0] \times [0{,}1; 10{,}0] \times [0{,}1; 10{,}0] \times [0{,}1; 2{,}0],$$

где $\tau(x) = \sqrt{(\tau')^2 + (2 \cdot \tau' \cdot \tau'') \cdot \frac{x_2}{2 \cdot R} + (\tau'')^2}$, $\tau' = \frac{6000}{\sqrt{2} \cdot x_1 \cdot x_2}$, $\tau'' = \frac{M \cdot R}{J}$,

$$M = 6000 \cdot \left(14 + \frac{x_2}{2}\right), \; R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \; J = 2\sqrt{2} \cdot x_1 \cdot x_2 \cdot \left(\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right),$$

$$\sigma(x) = \frac{504000}{x_3^2 \cdot x_4}, \; \delta(x) = \frac{65.856}{30 \cdot x_3^3 \cdot x_4}, \; P_c(x) = \frac{4.013 \cdot 5 \cdot 10^6 \cdot \sqrt{x_3^2 \cdot x_4^6}}{196} \cdot \left(1 - \frac{x_3 \cdot \sqrt{\frac{5}{8}}}{28}\right).$$

В табл. 1.38 приведены результаты сравнительного анализа результатов методов, описанных в разд. 1.3 и 1.4, и решения, найденного в [74].

Параметры гибридного мультиагентного метода интерполяционного поиска: $NP = 30$, $ITER = 100$, $M_1 = 3$, $M_2 = 4$, $PRT = 0,008$, $nstep = 6$, $b_2 = 7$. Коэффициенты штрафной функции: $c_1 = 0,001$, $c_2 = 0,001$, $c_3 = 10$, $c_4 = 1$, $c_5 = 1$, $c_6 = 1$, $c_7 = 0,001$.

Параметры мультиагентного метода, использующего линейные регуляторы для управления движением агентов: $NP = 501$, $I_{\max} = 50$, $P_{\max} = 10$, $k_s = 1$, $k_l = 5$, $h = 0,0001$. Коэффициенты штрафной функции: $c_1 = 0,001$, $c_2 = 0,001$, $c_3 = 10$, $c_4 = 1$, $c_5 = 1$, $c_6 = 1$, $c_7 = 0,001$.

Таблица 1.38. Результаты работы методов

	Гибридный мультиагентный алгоритм интерполяционного поиска	Мультиагентный метод, использующий линейные регуляторы для управления движением агентов	Решение, найденное в [74]
x_1^*	0,26	0,29	0,20573
x_2^*	2,67	2,73	3,470489
x_3^*	8,88	9,76	9,036624
x_4^*	0,21	0,21	0,205729
$f(x^*)$	**1,69**	1,9	1,724852
$g^1(x^*)$	– 152,28	– 2615,5	–0,025400
$g^2(x^*)$	– 435,84	– 4805,15	0,092700
$g^3(x^*)$	0,05	0,08	0,000001
$g^4(x^*)$	– 3,49	1,67	–3,432989
$g^5(x^*)$	–1,14	– 0,016	–0,080730
$g^6(x^*)$	– 0,24	– 0,24	–0,235540
$g^7(x^*)$	– 308,06	– 703,28	0,055938

Зададим параметры методов, описанных в разд. 1.6, и значения коэффициентов штрафной функции.

Параметры метода, имитирующего поведение стаи рыб: $NP = 30$, $ITER = 500$, $W_{scale} = 100$, $step_{vol} = 0,1$, $step_{ind} = 0,01$, $W = 70$. Коэффициенты штрафной функции: $c_1 = 0,001$, $c_2 = 0,001$, $c_3 = 10$, $c_4 = 1$, $c_5 = 1$, $c_6 = 1$, $c_7 = 0,001$.

Параметры метода, имитирующего поведение стаи криля: $NP = 40$, $ITER = 1000$, $N_{\max} = 0,01$, $V_f = 0,02$, $D_{\max} = 0,005$, $\mu = 3$, $c_t = 0,02$. Коэффициенты штрафной функции: $c_1 = 0,001$, $c_2 = 0,001$, $c_3 = 10$, $c_4 = 1$, $c_5 = 1$, $c_6 = 1$, $c_7 = 0,001$.

Параметры метода, имитирующего империалистическую конкуренцию: $N_{pop} = 150$, $N_{imp} = 15$, $ITER = 500$, $\beta = 0,6$, $\gamma = 0,06$, $\xi = 0,01$. Коэффициенты штрафной функции: $c_1 = 0,001$, $c_2 = 0,001$, $c_3 = 10$, $c_4 = 1$, $c_5 = 1$, $c_6 = 1$, $c_7 = 0,001$.

В табл. 1.39 приведены полученные результаты.

Таблица 1.39. Результаты работы методов

	Метод, имитирующий поведение стаи рыб	Метод, имитирующий поведение стаи криля	Метод, имитирующий империалистическую конкуренцию
x_1^*	0,3	0,23	0,36
x_2^*	3,82	3,6	2,7
x_3^*	9,36	9,07	9,03
x_4^*	0,17	0,21	0,71
$f(x^*)$	**1,7439**	1,8232	5,5376
$g^1(x^*)$	– 5262,61	– 100,19	– 4182,615
$g^2(x^*)$	3839,97	– 246,33	– 21294,44
$g^3(x^*)$	0,1299	– 0,005	–0,35
$g^4(x^*)$	– 3,6264	– 3,413	0,1646
$g^5(x^*)$	– 0,175	– 0,076	– 0,235
$g^6(x^*)$	– 0,2343	– 0,236	– 0,2458
$g^7(x^*)$	2536,46	– 47,169	– 240505,307

1.9.3. Задача определения параметров сосуда высокого давления

В рассматриваемой задаче [41, 108, 147] требуется определить параметры баллона для хранения сжатого газа (рис. 1.71).

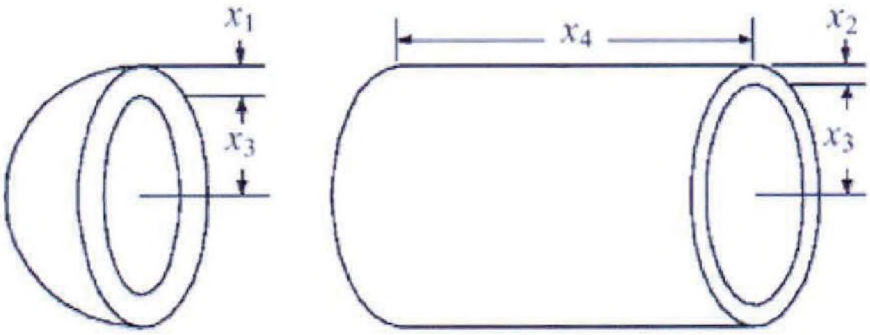


Рис. 1.71. Сосуд высокого давления

Целью является определение минимальной по стоимости конструкции сосуда, заключающейся в определении вектора параметров $x = (x_1, x_2, x_3, x_4)^T$, соответствующих толщине, толщине головки, внутреннему радиусу и длине цилиндрической части. Кроме того, величины x_1 и x_2 являются дискретными величинами (описывающими кратность параметра величине $0,0625$).

Задача может быть формализована следующим образом:

$$f(x) = 0,6224 \cdot \tilde{x}_1 \cdot x_3 \cdot x_4 + 1,7781 \cdot \tilde{x}_2 \cdot x_3^2 + 3,1661 \cdot \tilde{x}_1^2 \cdot x_4 + 19,84 \cdot \tilde{x}_1^2 \cdot x_3,$$

$$g^1(x) = -\tilde{x}_1 + 0,0193 \cdot x_3 \le 0,$$

$$g^2(x) = -\tilde{x}_2 + 0,00954 \cdot x_3 \le 0,$$

$$g^3(x) = -\pi \cdot x_3^2 \cdot x_4 - \frac{4}{3} \cdot \pi \cdot x_3^3 + 1296000 \le 0,$$

$$g^4(x) = x_4 - 240 \le 0,$$

$$D = [1;99,99] \times [1;99,99] \times [10;200] \times [10;200],$$

где $\tilde{x}_1 = 0,0625 \cdot \langle x_1 \rangle$, $\tilde{x}_2 = 0,0625 \cdot \langle x_2 \rangle$, $\langle \cdot \rangle$ – целая часть числа.

Зададим параметры методов, описанных в разд. 1.3 и 1.4, и значения коэффициентов штрафной функции.

Параметры гибридного мультиагентного метода интерполяционного поиска: $NP = 40$, $ITER = 200$, $M_1 = 3$, $M_2 = 4$, $PRT = 0,008$, $nstep = 6$, $b_2 = 7$. Коэффициенты штрафной функции: $c_1 = 40000$, $c_2 = 35000$, $c_3 = 1000$, $c_4 = 900$.

Параметры мультиагентного метода, основанного на применении линейных регуляторов управления движением агентов: $NP = 501$, $I_{\max} = 50$, $P_{\max} = 10$, $k_s = 1$,

$k_l = 5$, $h = 0,0001$. Коэффициенты штрафной функции: $c_1 = 40000$, $c_2 = 35000$, $c_3 = 1000$, $c_4 = 900$.

В табл. 1.40 приведены данные для сравнительного анализа результатов обоих методов с решением, полученным в [147].

Таблица 1.40. Результаты работы методов

	Гибридный мультиагентный алгоритм интерполяционного поиска	Мультиагентный метод, использующий линейные регуляторы для управления движением агентов	Решение, найденное в [147]
x_1^*	11,73	10,97	13
x_2^*	5,56	5,3	7
x_3^*	42,37	42,4	42,098446
x_4^*	173,87	173,68	176,636596
$f(x^*)$	5218,41	**4855,28**	6059,714335
$g^1(x^*)$	0,0846	0,1327	0,000
$g^2(x^*)$	0,0567	0,0732	–0,035881
$g^3(x^*)$	– 3212,3936	– 4205,6803	–0,028761
$g^4(x^*)$	– 66,13	– 66,32	–63,363404

Зададим параметры методов, описанных в разд. 1.6, и значения коэффициентов штрафной функции.

Параметры метода, имитирующего поведение стаи рыб: $NP = 30$, $ITER = 10000$, $W_{scale} = 5000$, $step_{vol} = 0,3$, $step_{ind} = 0,03$, $W = 4500$. Коэффициенты штрафной функции: $c_1 = 40000$, $c_2 = 35000$, $c_3 = 1000$, $c_4 = 900$.

Параметры метода, имитирующего поведение стаи криля: $NP = 60$, $ITER = 1000$, $N_{\max} = 0,01$, $V_f = 0,02$, $D_{\max} = 0,005$, $\mu = 3$, $c_t = 0,02$. Коэффициенты штрафной функции: $c_1 = 40000$, $c_2 = 35000$, $c_3 = 1000$, $c_4 = 900$.

Параметры метода, имитирующего империалистическую конкуренцию: $N_{pop} = 150$, $N_{imp} = 15$, $ITER = 500$, $\beta = 0,9$, $\gamma = 0,09$, $\xi = 0,01$. Коэффициенты штрафной функции: $c_1 = 40000$, $c_2 = 35000$, $c_3 = 1000$, $c_4 = 900$.

В табл. 1.41 приведены полученные результаты.

Таблица 1.41. Результаты работы методов

	Метод, имитирующий поведение стаи рыб	Метод, имитирующий поведение стаи криля	Метод, имитирующий империалистическую конкуренцию
x_1^*	11,74	11,71	11,07
x_2^*	5,73	5,73	6,93
x_3^*	42,3	42,1	42,51
x_4^*	174,14	176,65	179,35
$f(x^*)$	**5252,06**	5263,29	5350,41
$g^1(x^*)$	0,083	0,081	0,1285
$g^1(x^*)$	0,083	0,081	0,1285
$g^3(x^*)$	83,868	–181,88	– 43983,001
$g^4(x^*)$	– 65,86	– 63,349	– 60,65

1.9.4. Задача определения параметров редуктора

В рассматриваемой задаче [41, 108, 150] требуется определить параметры редуктора (рис. 1.72).

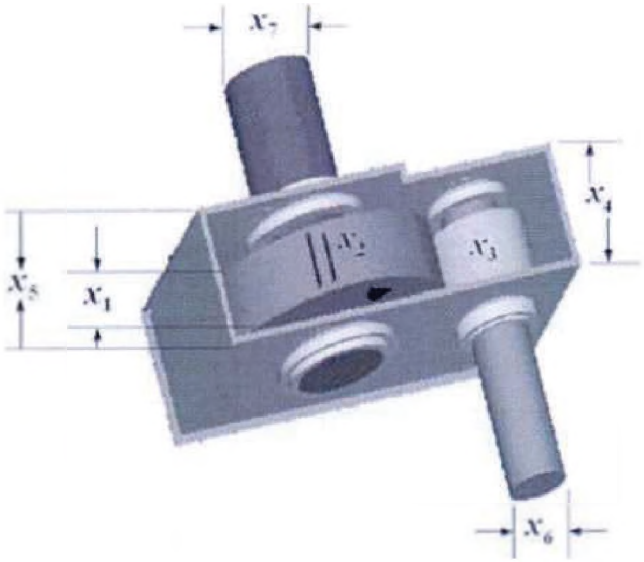


Рис. 1.72. Конструкция редуктора

Целью является определение минимальной по весу конструкции редуктора, заключающейся в определении вектора параметров $x = (x_1, x_2, x_3, x_4, x_5, x_6, x_7)^T$, соответствующих ширине лицевой стороны, длине зубцов, числу зубцов на шестерне (целочисленная величина), длине первого вала, длине второго вала, диаметру первого вала и диаметру второго вала. Конструкция редуктора должна удовлетворять ограничениям по напряжению изгиба зубцов шестерни, поверхностному напряжению, поперечным отклонениям валов и напряжению на валах.

Задача может быть формализована следующим образом:

$$f(x) = 0,7854 \cdot x_1 \cdot x_2^2 \cdot (3,3333 \cdot \langle x_3 \rangle^2 + 14,9334 \cdot \langle x_3 \rangle - 43,0934) -$$
$$-1,508 \cdot x_1 \cdot (x_6^2 + x_7^2) + 7,4777 \cdot (x_6^3 + x_7^3) + 0,7854 \cdot (x_4 \cdot x_6^2 + x_5 \cdot x_7^2),$$

$$g^1(x) = \frac{27}{x_1 \cdot x_2^2 \cdot \langle x_3 \rangle} - 1 \le 0, \quad g^2(x) = \frac{397,5}{x_1 \cdot x_2^2 \cdot \langle x_3 \rangle^2} - 1 \le 0,$$

$$g^3(x) = \frac{1,93 \cdot x_4^3}{x_2 \cdot \langle x_3 \rangle \cdot x_6^4} - 1 \le 0, \quad g^4(x) = \frac{1,93 \cdot x_5^3}{x_2 \cdot \langle x_3 \rangle \cdot x_7^4} - 1 \le 0,$$

$$g^5(x) = \frac{1}{110 \cdot x_6^3} \cdot \sqrt{\left(\frac{745 \cdot x_4}{x_2 \cdot \langle x_3 \rangle} \right)^2 + 16,9 \cdot 10^6} - 1 \le 0,$$

$$g^6(x) = \frac{1}{85 \cdot x_7^3} \cdot \sqrt{\left(\frac{745 \cdot x_5}{x_2 \cdot \langle x_3 \rangle} \right)^2 + 157,5 \cdot 10^6} - 1 \le 0,$$

$$g^7(x) = \frac{x_2 \cdot \langle x_3 \rangle}{40} - 1 \le 0, \quad g^8(x) = \frac{5 \cdot x_2}{x_1} - 1 \le 0, \quad g^9(x) = \frac{x_1}{12 \cdot x_2} - 1 \le 0,$$

$$g^{10}(x) = \frac{1,5 \cdot x_6 + 1,9}{x_5} - 1 \le 0, \quad g^{11}(x) = \frac{1,1 \cdot x_7 + 1,9}{x_5} - 1 \le 0,$$

$$D = [2,6;3,6] \times [0,7;0,8] \times [17;28,99] \times [7,3;8,3] \times [7,8;8,3] \times [2,9;3,9] \times [5,0;5,5],$$

где $\langle \cdot \rangle$ – целая часть числа.

Зададим параметры методов, описанных в разд. 1.3 и 1.4, и значения коэффициентов штрафной функции. Результаты сравнения с решением, найденным в [150], приведены в табл. 1.42.

Параметры гибридного мультиагентного метода интерполяционного поиска: $NP = 40$, $ITER = 200$, $M_1 = 3$, $M_2 = 4$, $PRT = 0,008$, $nstep = 6$, $b_2 = 7$. Коэффициенты штрафной функции: $c_1 = 1$, $c_2 = 1$, $c_3 = 1$, $c_4 = 1$, $c_5 = 1$, $c_6 = 1$, $c_7 = 1$, $c_8 = 1$, $c_9 = 1$, $c_{10} = 1$, $c_{11} = 1$.

Параметры мультиагентного метода, основанного на применении линейных регуляторов управления движением агентов: $NP = 501$, $I_{\max} = 50$, $P_{\max} = 10$, $k_s = 1$, $k_l = 5$, $h = 0,0001$. Коэффициенты штрафной функции: $c_1 = 1$, $c_2 = 1$, $c_3 = 1$, $c_4 = 1$, $c_5 = 1$, $c_6 = 1$, $c_7 = 1$, $c_8 = 1$, $c_9 = 1$, $c_{10} = 1$, $c_{11} = 1$.

Таблица 1.42. Результаты работы методов

	Гибридный мультиагентный алгоритм интерполяционного поиска	Мультиагентный метод, использующий линейные регуляторы для управления движением агентов	Решение, полученное в [150]
x_1^*	2,3	2,7	3,5
x_2^*	0,7	0,7	0,7
x_3^*	15,4	17,1	17
x_4^*	7	7,8	7,3
x_5^*	7,7	7,8	7,8
x_6^*	2,6	3,0	3,350214
x_7^*	4,9	5,0	5,286683
$f(x^*)$	**1910,7991**	2426,45	2996,348165
$g^1(x^*)$	0,5972	0,2004	$-0,073915$
$g^2(x^*)$	0,5675	0,0396	$-0,197999$
$g^3(x^*)$	0,3796	$-0,0498$	$-0,499172$
$g^4(x^*)$	$-0,8544$	$-0,08768$	$-0,901472$
$g^5(x^*)$	1,1418	0,3938	$6\cdot10^{-7}$
$g^6(x^*)$	0,2562	0,1821	$1,3\cdot10^{-7}$
$g^7(x^*)$	$-0,7375$	$-0,7025$	$-0,702500$
$g^8(x^*)$	0,5217	0,2963	0,000
$g^9(x^*)$	$-0,7262$	$-0,6785$	$-0,583333$
$g^{10}(x^*)$	$-0,2467$	$-0,1795$	$-0,112138$
$g^{11}(x^*)$	$-0,0532$	$-0,0513$	$-0,010852$

Зададим параметры методов, описанных в разд. 1.6, и значения коэффициентов штрафной функции. Результаты приведены в табл. 1.43.

Таблица 1.43. Результаты работы методов

	Метод, имитирующий поведение стаи рыб	Метод, имитирующий поведение стаи криля	Метод, имитирующий империалистическую конкуренцию
x_1^*	3	3,2	2,6
x_2^*	0,2	0,7	0,8
x_3^*	17	18	18
x_4^*	7,4	7,4	7,6
x_5^*	7,8	7,8	8,3
x_6^*	3,5	3,1	3
x_7^*	5,3	5,3	5,3
$f(x^*)$	**1605,301**	2990,011	3112,991
$g^1(x^*)$	12,235	– 0,04	– 0,0986
$g^2(x^*)$	10,462	– 0,217	– 0,2627
$g^3(x^*)$	0,533	– 0,327	– 0,2736
$g^4(x^*)$	– 0,659	– 0,91	– 0,9029
$g^5(x^*)$	– 0,063	0,262	0,3905
$g^6(x^*)$	0,00089	– 0,008	– 0,0077
$g^7(x^*)$	– 0,915	– 0,69	– 0,64
$g^8(x^*)$	– 0,666	0,09	0,5385
$g^9(x^*)$	0,2499	– 0,619	– 0,7292
$g^{10}(x^*)$	– 0,0833	– 0,16	– 0,2289
$g^{11}(x^*)$	– 0,0089	– 0,009	– 0,0687

Параметры метода, имитирующего поведение стаи рыб: $NP = 30$, $ITER = 10000$, $W_{scale} = 5000$, $step_{vol} = 0,1$, $step_{ind} = 0,01$, $W = 4500$. Коэффициенты штрафной функции: $c_1 = 0,1$, $c_2 = 0,1$, $c_3 = 0,1$, $c_4 = 0,1$, $c_5 = 0,1$, $c_6 = 0,1$, $c_7 = 0,1$, $c_8 = 0,1$, $c_9 = 0,1$, $c_{10} = 0,1$, $c_{11} = 0,1$.

Параметры метода, имитирующего поведение стаи криля: $NP = 40$, $ITER = 1000$, $N_{\max} = 0,01$, $V_f = 0,02$, $D_{\max} = 0,005$, $\mu = 3$, $c_t = 0,02$. Коэффициенты штрафной функции: $c_1 = 1$, $c_2 = 1$, $c_3 = 1$, $c_4 = 1$, $c_5 = 1$, $c_6 = 1$, $c_7 = 1$, $c_8 = 1$, $c_9 = 1$, $c_{10} = 1$, $c_{11} = 1$.

Параметры метода, имитирующего империалистическую конкуренцию: $N_{pop} = 150$, $N_{imp} = 15$, $ITER = 500$, $\beta = 0,4$, $\gamma = 0,04$, $\xi = 0,01$. Коэффициенты штрафной функции: $c_1 = 1$, $c_2 = 1$, $c_3 = 1$, $c_4 = 1$, $c_5 = 1$, $c_6 = 1$, $c_7 = 1$, $c_8 = 1$, $c_9 = 1$, $c_{10} = 1$, $c_{11} = 1$.

1.9.5. Задача определения параметров натяжной/компрессионной пружины

В рассматриваемой задаче [41, 94, 108] требуется определить параметры натяжной/ компрессионной пружины, учитывая физические ограничения (рис. 1.73).

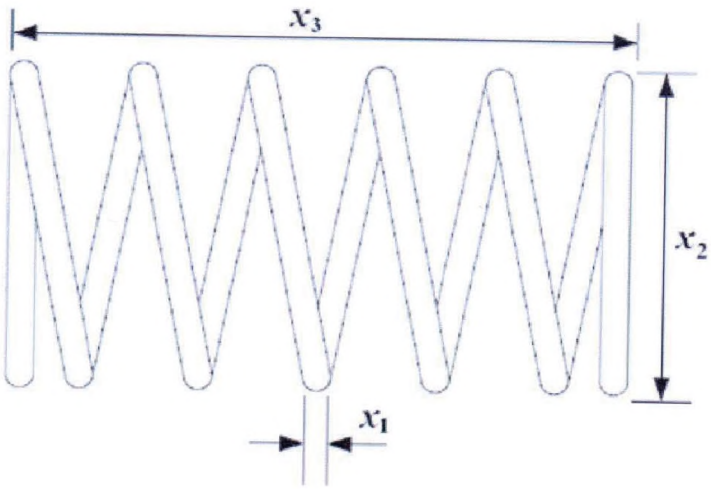


Рис. 1.73. Конструкция пружины

Целью является определение минимальной по весу конструкции пружины, заключающейся в определении вектора параметров $x = (x_1, x_2, x_3)^T$, соответствующих диаметру проволоки, среднему диаметру витка и числу активных витков. Конструкция пружины должна удовлетворять ограничениям по минимальному отклонению, напряжению сдвига, частоте колебаний и ограничениям на внешний диаметр.

Задача может быть формализована следующим образом:

$$f(x) = (x_3 + 2) \cdot x_1^2 \cdot x_2,$$

$$g^1(x) = 1 - \frac{x_2^3 \cdot x_3}{71875 \cdot x_1^4} \le 0, \quad g^2(x) = \frac{4 \cdot x_2^2 - x_1 \cdot x_2}{12566 \cdot (x_1^3 \cdot x_2 - x_1^4)} + \frac{2,46}{12566 \cdot x_1^2} - 1 \le 0,$$

$$g^3(x) = 1 - \frac{140,54 \cdot x_1}{x_2^2 \cdot x_3} \le 0, \quad g^4(x) = \frac{x_1 + x_2}{1,5} - 1 \le 0,$$

$$D = [0,05; 2,0] \times [0,25; 1,3] \times [2,0; 15,0].$$

Зададим параметры методов, описанных в разд. 1.3 и 1.4, и значения коэффициентов штрафной функции. Результаты применения обоих методов в сравнении с полученными в [94] приведены в табл. 1.44.

Параметры гибридного мультиагентного метода интерполяционного поиска: $NP = 30$, $ITER = 200$, $M_1 = 3$, $M_2 = 4$, $PRT = 0,008$, $nstep = 6$, $b_2 = 7$. Коэффициенты штрафной функции: $c_1 = 6$, $c_2 = 1$, $c_3 = 1$, $c_4 = 0,5$.

Параметры мультиагентного алгоритма оптимизации на основе применения линейных регуляторов управления движением агентов: $NP = 2001$, $P_{\max} = 50$, $N_{\max} = 10$, $k_s = 0,1$, $k_l = 5$, $h = 0,0001$. Коэффициенты штрафной функции: $c_1 = 6$, $c_2 = 1$, $c_3 = 1$, $c_4 = 0,5$.

Таблица 1.44. Результаты работы методов

	Гибридный мультиагентный алгоритм интерполяционного поиска	Мультиагентный метод, использующий линейные регуляторы для управления движением агентов	Решение, полученное в [94]
x_1^*	0,05	0,04	0,05169
x_2^*	0,42	0,37	0,35675
x_3^*	8,32	10,95	11,287126
$f(x^*)$	**0,0108**	0,0156	0,012665
$g^1(x^*)$	– 0,37	– 0,171	–9,001053
$g^2(x^*)$	0,2562	0,1418	0,000020
$g^3(x^*)$	– 3,79	– 3,725	–4,057026
$g^4(x^*)$	– 0,68	– 0,69	–0,727707

Зададим параметры методов, описанных в разд. 1.6, и значения коэффициентов штрафной функции. Результаты приведены в табл. 1.45.

Параметры метода, имитирующего поведение стаи рыб: $NP = 30$, $ITER = 10000$, $W_{scale} = 5000$, $step_{vol} = 0,1$, $step_{ind} = 0,01$, $W = 4500$. Коэффициенты штрафной функции: $c_1 = 6$, $c_2 = 1$, $c_3 = 1$, $c_4 = 0,5$.

Параметры метода, имитирующего поведение стаи криля: $NP = 40$, $ITER = 1000$, $N_{\max} = 0,01$, $V_f = 0,02$, $D_{\max} = 0,005$, $\mu = 3$, $c_t = 0,02$. Коэффициенты штрафной функции: $c_1 = 6$, $c_2 = 1$, $c_3 = 1$, $c_4 = 0,5$.

Параметры метода, имитирующего империалистическую конкуренцию: $N_{pop} = 150$, $N_{imp} = 15$, $ITER = 500$, $\beta = 0,6$, $\gamma = 0,06$, $\xi = 0,01$. Коэффициенты штрафной функции: $c_1 = 6$, $c_2 = 1$, $c_3 = 1$, $c_4 = 0,5$.

Таблица 1.45. Результаты работы методов

	Метод, имитирующий поведение стаи рыб	Метод, имитирующий поведение стаи криля	Метод, имитирующий империалистическую конкуренцию
x_1^*	0,05	0,05	0,06
x_2^*	0,36	0,37	0,49
x_3^*	10,93	10,37	10,28
$f(x^*)$	**0,0114**	**0,0114**	0,0216
$g^1(x^*)$	– 0,169	– 0,169	– 0,2983
$g^2(x^*)$	0,1309	0,1309	– 0,1479
$g^3(x^*)$	– 3,949	– 3,949	– 2,4164
$g^4(x^*)$	– 0,72	– 0,72	– 0,6333

На основании анализа решения прикладных задач, описанных в разд. 1.9.2–1.9.5, можно сделать следующие выводы. Гибридный мультиагентный алгоритм интерполяционного поиска и мультиагентный метод, использующий линейные регуляторы для управления движением агентов, продемонстрировали свою относительную эффективность. Этот же вывод относится к методам, имитирующим поведение стаи рыб и стаи криля. Полученные результаты оказались сравнимыми с известными наилучшими решениями. Метод, имитирующий империалистическую конкуренцию, может служить для получения хорошего начального приближения, требующего дальнейшего уточнения.

В [54] приведены результаты решения перечисленных задач с применением метода спиральной динамики, метода, имитирующего поиск группой людей, и метода стохастической диффузии. Результаты сравнимы с приведенными в данном разделе. Они подтверждают применимость широкого класса метаэвристических алгоритмов оптимизации к данному классу прикладных технических проблем.

ГЛАВА 2
БИОИНСПИРИРОВАННЫЕ МЕТОДЫ ОПТИМИЗАЦИИ

2.1. ПОСТАНОВКА ЗАДАЧИ ОПТИМИЗАЦИИ

Дана целевая функция $f(x) = f(x_1, x_2, ..., x_n)$, определенная на множестве допустимых решений $D \subseteq R^n$.

Задача 1. Требуется найти условный глобальный минимум функции $f(x)$ на множестве D, т.е. такую точку $x^* \in D$, что

$$f(x^*) = \min_{x \in D} f(x), \tag{2.1}$$

где $x = (x_1, x_2, ..., x_n)^T$, $D = \{x \mid x_i \in [a_i, b_i], i = 1, 2, ..., n\}$.

Задача 2. Требуется найти условный глобальный максимум функции $f(x)$ на множестве D, т.е. такую точку $x^* \in D$, что

$$f(x^*) = \max_{x \in D} f(x), \tag{2.2}$$

где $x = (x_1, x_2, ..., x_n)^T$, $D = \{x \mid x_i \in [a_i, b_i], i = 1, 2, ..., n\}$.

З а м е ч а н и я.

1. Задача поиска максимума функции $f(x)$ сводится к задаче поиска минимума путем замены знака перед функцией на противоположный: $f(x^*) = \max_{x \in D} f(x) = -\min_{x \in D} [-f(x)]$. Аналогично задача поиска минимума функции $f(x)$ сводится к задаче поиска максимума путем замены знака перед функцией на противоположный: $f(x^*) = \min_{x \in D} f(x) = -\max_{x \in D} [-f(x)]$.

2. Предполагается, что целевая функция является непрерывной.

2.2. ПРИНЦИПЫ ФОРМИРОВАНИЯ БИОИНСПИРИРОВАННЫХ МЕТОДОВ ОПТИМИЗАЦИИ

Биоинспирированные метаэвристические алгоритмы оптимизации порождаются наблюдениями за различными, физическими и химическими процессами, поведением наземных, подводных и летающих представителей животного мира, микроорганизмов, а также за поведением людей в процессе их взаимодействия.

Можно условно выделить семь категорий биоинспирированных алгоритмов.

1. Имитирующие взаимодействие наземных растений, насекомых и животных.
2. Имитирующие взаимодействие подводных растений и животных.
3. Имитирующие взаимодействие летающих насекомых и птиц.
4. Имитирующие взаимодействие микроорганизмов.
5. Имитирующие физические и химические процессы.
6. Имитирующие социальное взаимодействие людей.
7. Имитирующие спортивные, настольные, компьютерные игры.

Биоинспирированные алгоритмы оптимизации, как правило, содержат шесть основных этапов. Опишем основные используемые эвристики с указанием примеров их применения в данном классе метаэвристических алгоритмов.

Формирование начальной популяции (первый этап). Начальная популяция решений образуется допустимыми решениями (называемыми также особями, членами популяции, членами стаи в зависимости от смыслового содержания алгоритма) задачи оптимизации (2.1). При этом каждая координата члена популяции должна удовлетворять заданным в задаче интервальным ограничениям. Имеются следующие сценарии.

Сценарий 1. Для генерирования начальной популяции используется равномерный закон распределения с учетом гарантий принадлежности каждого члена популяции множеству допустимых решений.

Сценарий 2. Используется равномерный закон распределения, но с учетом непревышения порогового значения расстояния между членами популяции (например, в методе рассеивания [79, 92]).

Сценарий 3. Для генерирования начальной популяции используется равномерный закон распределения с учетом принадлежности множеству допустимых решений. Далее производится ранжирование членов популяции по величине целевой функции и задается фиксированное число элитных особей с наилучшим значением целевой функции. Для каждой элитной особи генерируются клоны с тем же положением на множестве допустимых решений. Количество клонов может быть одинаковым для каждой элитной особи или может быть пропорционально величине целевой функции, т.е. чем лучше значение целевой функции, тем больше генерируется клонов. Данный вариант используется в методах, имитирующих иммунные системы организмов и распространение сорняков. Положение элитной особи и ее клонов может изменяться в процессе поиска экстремума различными способами.

В начальной популяции, как правило, находится текущий абсолютный лидер по величине целевой функции.

Начальная популяция решений может быть поделена на группы (стаи) различными способами:

а) равномерно без учета соответствующих значений целевой функции;

б) неравномерно с учетом ранжирования по величине целевой функции. При этом выделяется группа лидеров (три волка в методе, имитирующем поведение стаи серых волков; группа империй в методе, имитирующем поведение империалистических государств);

в) равномерно с учетом значений целевой функции. При этом первоначально производится ранжирование решений по величине целевой функции. Порядок формирования M групп: наилучшее решение помещается в первую группу, следующее – во вторую, M-е – в M-ю группу, $(M+1)$-е – снова в первую группу и т.д. В каждой группе выявляется лидер группы по величине целевой функции. Алгоритмы поиска экстремума в каждой группе, как правило, отличаются друг от друга. Такой подход применяется в методе, имитирующем поведение стаи лягушек [81–83].

Исследовательский поиск (второй этап). Реализуется путем повторяющихся итераций, обеспечивающих эволюцию популяции решений в процессе поиска экстремума. В некоторых случаях задается фиксированное число итераций, образующих один проход, для того, чтобы изменить или скорректировать стратегию поиска по окончании прохода.

На текущей итерации реализуются различные типы движения членов популяции.

Индивидуальное движение.

Сценарий 1. Движение под действием генетических операторов – селекции, скрещивания, мутации (генетические алгоритмы, метод дифференциальной эволюции, метод искусственных иммунных систем, метод рассеивания) [54, 75, 79, 92, 93].

Сценарий 2. Движение к абсолютному лидеру популяции (метод искусственной пчелиной колонии; метод, имитирующий поведение стаи лягушек; метод, имитирующий поведение стаи криля) [54, 81–83, 89, 159].

Сценарий 3. Движение к лидеру группы (метод, имитирующий поведение стаи лягушек; метод, имитирующий поведение стаи окуней; метод, имитирующий поиск группой людей) [54, 67, 76, 133].

Сценарий 4. Движение к случайно выбранному соседу с лучшим значением целевой функции (метод частиц в стае; метод, имитирующий поведение стаи светлячков) [54, 74, 76, 114].

Сценарий 5. Движение путем подражания действиям случайно выбранного соседа (например, движение с той же скоростью в методе частиц в стае) [74, 76, 114].

Сценарий 6. Движение с учетом информации о своем наилучшем положении в прошлом (метод частиц в стае) [74, 76, 114]. Иногда память ограничивается последними тремя итерациями (метод, имитирующий поиск группой людей).

Сценарий 7. Движение с учетом взаимодействия с остальными членами популяции (например, в методе гравитационной кинематики с учетом сил гравитационного притяжения тел, а в методе, имитирующем империалистическую конкуренцию, влияние всех колоний на силу империалистического государства) [54, 63, 113].

Сценарий 8. Движение, при котором изменяются только те координаты решения, которые выбираются путем определенной стохастической процедуры (миграционные алгоритмы) [77, 161, 162].

Сценарий 9. Движение под действием сил, определяемых природой взаимодействия особей в популяции. Используются кинематические соотношения, связывающие положение особи со скоростью и ускорением (метод частиц в стае, метод имитации поведения бактерий, метод, имитирующий поведение светлячков; методы, имитирующие поведение стаи криля и стаи стрекоз), а также дифференциальные уравнения движения (метод, имитирующий поведение стаи синиц) [54, 64, 89, 128, 134, 159].

Сценарий 10. Движение по спирали в множестве допустимых решений (метод, имитирующий поведение стаи горбатых китов; метод спиральной динамики) [39, 54, 128],

Сценарий 11. Движение, связанное с переходом в положение с худшим значением целевой функции с некоторой вероятностью (метод имитации отжига) [54, 80]. Это позволяет алгоритму не застревать в окрестности локальных экстремумов.

Сценарий 12. Движение к конечному числу лидеров популяции (трем лидерам в методе, имитирующем поведение стаи серых волков) [38, 54, 128].

Сценарий 13. Движение методом случайных блужданий (метод, имитирующий поведение стаи рыб; метод, имитирующий поведение стаи криля; метод поиска гармонии) [54, 64, 66–68, 89].

Коллективное движение.

Сценарий 1. Движение относительно центра масс всей популяции (метод, имитирующий поведение стаи рыб) [66–68, 122].

Сценарий 2. Движение с целью обеспечения сплоченности, выравнивания, разделения (метод, имитирующий поведение стаи стрекоз) [39, 54, 128].

Сценарий 3. Движение от центра масс, описываемое нормальным законом распределения, параметры которого определяются соответствующей величиной целевой функции (метод большого взрыва-большого сжатия; метод, имитирующий распространение сорняков) [54, 124].

Перераспределительный поиск (третий этап). Используется для активизации процесса поиска и исследования еще не посещенных членами популяции подмножеств множества допустимых решений.

Сценарий 1. Движение наихудшей особи или лидера наихудшей группы с помощью полета Леви (метод, имитирующий поведение стаи кукушек) [157, 158]. При этом реализуются значительные приращения координат решения, что гарантирует радикальную смену области поиска.

Сценарий 2. Движение в перспективную область (метод роя пчел), центр которой определяется положением решения, отличающегося от лучшего по величине целевой функции, но не более, чем на заданную величину, а границы определяются расстояниями до границ множества допустимых решений [54, 80].

Сценарий 3. Генерировать новые положения членов популяции с учетом информации, хранящейся в листе посещенных областей и табу-областей (метод направленного табу-поиска) [54, 79, 80].

Формирование новой популяции (четвертый этап). Производится по окончании фиксированного числа итераций (или прохода). При этом реализуется идея эволюционного отбора наиболее лучших решений и исключения остальных.

Сценарий 1. С целью сохранения постоянного размера популяции все полученные в результате прохода решения ранжируются по величине целевой функции, а затем удаляются все лишние. При этом число сохраненных решений, как правило, меньше размера начальной популяции.

Сценарий 2. Из текущей популяции удаляется задаваемое пользователем количество наихудших решений.

Сценарий 3. Из текущей популяции помимо наихудших членов путем попарного сравнения удаляются близко расположенные решения (из двух удаляется то, которому соответствует худшее значение целевой функции).

Вместо удаленных из популяции в ходе реализации сценариев 1-3 решений генерируются новые решения способами, применяемыми при формировании начальной популяции.

Как правило, для сохранения текущих результатов поиска наилучшее решение в популяции помещается в хранилище (множество *Pool*, матрицу памяти и т.д.).

Если популяция была разделена на группы, то возможно реализовать обмен информацией между группами.

Сценарий 4. В группу с наихудшими значениями целевой функции у лидера группы добавляются члены популяции с положениями, соответствующими лидерам более успешных групп.

Сценарий 5. Производится заново деление на группы способами, используемыми при формировании начальной популяции.

После формирования новой популяции процесс поиска продолжается до завершения заданного числа проходов.

Интенсивно-уточняющий поиск (пятый этап). Применяется в случае необходимости уточнения полученного решения и может отсутствовать в алгоритме оптимизации.

Сценарий 1. Применить известные классические методы нулевого порядка для поиска в окрестности наилучшего решения в популяции. Могут быть использованы метод деформируемого многогранника, метод конфигураций, методы случайного поиска (адаптивный, с возвратом при неудачном шаге, метод наилучшей пробы), локальный линейный поиск.

Сценарий 2. Если в ходе итераций поиска из локальных наилучших решений сформировано множество *Pool*, то реализуется процедура поиска на основе этих решений. Например, используется процедура перекоммутации (path-relinking) [92] с использованием трех случайно выбранных элементов этого множества (метод, имитирующий поведение стаи окуней).

Нахождение решения задачи оптимизации (2.1) (шестой этап).

Сценарий 1. В качестве приближенного решения используется наилучшее решение, найденное в процессе интенсивно-уточняющего поиска.

Сценарий 2. В качестве приближенного решения используется наилучшее решение в последней популяции, которая определяется достижением предельного числа итераций или заданного числа проходов.

2.3. МЕТОД, ИМИТИРУЮЩИЙ ПОВЕДЕНИЕ СТАИ ОКУНЕЙ

2.3.1. Стратегия поиска решения

Метод (Perch School Search – PSS) имитирует поведение стай речного окуня [133]. С целью добывания пищи окуни средних размеров сбиваются в стаи по 5–12 особей. Совсем мелкие окуни образуют стаи, содержащие около 100 особей. Как правило, они берут рыбу, на которую они охотятся, в кольцо (окуневый котел) и из него не выпускают. Окуни атакуют жертв, находящихся ближе к границе котла, постепенно продвигаясь к его центру. Для поиска новых источников пищи окуни используют механизм миграции. Более крупные окуни обычно держатся на глубине, в ямах и омутах и охотятся поодиночке. Они реже, чем мелкие, организуют стаи, но известно, что они объединяются для борьбы с другими хищными рыбами (щуками и судаками). Речной окунь использует весьма агрессивную модель охоты, он активно преследует жертву, иногда выскакивая за ней даже на поверхность воды. Самые крупные окуни, – одиночные, самостоятельные хищники. Это связанно с тем, что крупный окунь уже не нуждается в коллективной охоте и может самостоятельно охотиться на любую, доступную по размеру рыбу. При отсутствии кормовых объектов на постоянном месте обитания окунь начинает перемещаться в поисках мест, изобилующих мелкой рыбешкой и другим кормом.

При решении задачи поиска глобального условного минимума функции используются конечные наборы $I = \{x^j = (x_1^j, x_2^j, ..., x_n^j)^T, j = 1, 2, ..., NP\} \subset D$ возможных решений, называемые популяциями, где x^j – особь (окунь) с номером j, NP – размер популяции.

В начале процесса метод порождает популяцию окуней с помощью равномерного на множестве допустимых решений распределения. При этом все решения упорядочиваются по возрастанию значения целевой функции. Затем популяция делится на стаи. Порядок формирования стай: наилучшее решение помещается в первую стаю, следующее – во вторую и т.д. M-е в M-ю стаю, $(M+1)$-е снова в первую стаю, $(M+2)$-е во вторую стаю и т.д. Описанный процесс соответствует обмену информацией между стаями с целью эффективного приближения к глобальному экстремуму.

В каждой стае определяется лидер по величине целевой функции. Каждая стая организует окуневый котел, в котором происходит охота. При этом реализуется движение всех окуней стаи к своему лидеру, исследуя границы котла. Во время движения запоминается наилучшее положение (в этом положении охота считается удачной,

происходит фаза питания окуней). В результате находятся новые лидеры каждой стаи и новые положения ее членов. Среди лидеров стай находится абсолютный лидер и наименее успешный.

В стае, соответствующей абсолютному лидеру, реализуется окуневый котел, в котором тщательно исследуется вся область, занимаемая стаей. В результате находится новый абсолютный лидер.

Затем среди стай выбирается стая с самым слабым лидером. Она перемещается в другую область множества допустимых решений (водоема). Для этого реализуется движение лидера стаи на основе распределения Леви с проверкой принадлежности множеству допустимых решений. Остальные члены стаи генерируются с помощью равномерного закона распределения на параллелепипедном множестве, размер которого по каждой координате определяется удвоенным расстоянием от лидера до ближайшей границы множества D. В полученной стае реализуется окуневый котел и, как следствие, находится новый лидер.

Остальные стаи совершают плавание по направлению к текущему абсолютному лидеру всей популяции. При этом локальный лидер стаи двигается по прямой к абсолютному лидеру, а остальные окуни этой стаи двигаются параллельно ему. В этом движении все окуни запоминают свою наилучшую позицию и в ней остаются (питаются).

По окончании миграции всех стай популяции абсолютный лидер помещается в множество *Pool*. Происходит новое деление популяции на стаи и начинается новая глобальная итерация до достижения заданного их числа.

На заключительном шаге в множестве *Pool* организуется взаимодействие лидеров, выявленных на каждой глобальной итерации алгоритма (охота крупных окуней за более серьезной добычей). Заданное число раз в множестве *Pool* выбирается тройка окуней и реализуется операция перекоммутации (path-relinking) [92], в результате которой множество пополняется еще одним решением.

После окончания процедуры перекоммутации среди элементов множества *Pool* находится наилучшее решение, которое считается приближенным решением поставленной задачи.

Предложенный метод является гибридным, так как содержит идеи, использованные в алгоритме лягушек (деление на стаи), алгоритме кукушек (перелет Леви), миграционном алгоритме (движение к лидеру), алгоритме перекоммутации (поиск в множестве *Pool*).

Иллюстрация работы метода изображена на рис. 2.1., а общая схема работы метода на рис. 2.2.

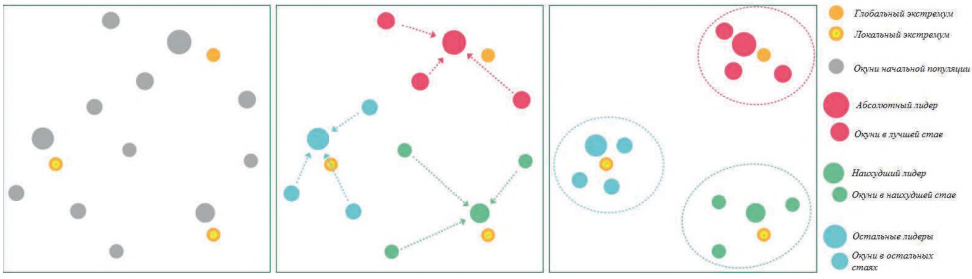


Рис. 2.1. Иллюстрация работы метода, имитирующего поведение стаи окуней

Рис. 2.2. Общая схема работы метода, имитирующего поведение стаи окуней

2.3.2. Алгоритм решения задачи

Шаг 1. *Задание параметров метода*:

- контролирующий параметр *NStep*, определяющий количество шагов до окончания движения;
- количество стай в популяции M;
- количество окуней в стае s;
- число членов популяции $NP = s \cdot M$;
- останавливающий параметр $Iter_{\max}$, определяющий максимальное количество итераций;
- параметр λ распределения Леви;
- величина шага α;
- максимальное число перекоммутаций $PR_{\max}$;
- число шагов в процедуре перекоммутации Δ_{pr}.

Шаг 2. *Создание начальной популяции окуней.*

Шаг 2.1. Создать популяцию $I = \{x^j = (x_1^j, x_2^j, ..., x_n^j)^T, j = 1, 2, ..., NP\} \subset D$ из NP решений (окуней) со случайно сгенерированными координатами x_i из промежутка $[a_i, b_i]$ с использованием равномерного закона распределения:

$$x_i^j = a_i + rand_i[0;1] \cdot (b_i - a_i),\ i = 1, \ldots, n;\ j = 1, \ldots, NP,$$

где $rand_i[0;1]$ – равномерный закон распределения на отрезке $[0;1]$.

Шаг 2.2. Для каждого решения (окуня) в популяции вычислить значения целевой функции.

Положить $iter = 1$ (счетчик числа глобальных итераций).

Шаг 3. *Деление популяции на стаи.*

Шаг 3.1. Упорядочить решения в популяции по возрастанию значений целевой функции.

Шаг 3.2. Сформировать M стай по s окуней в каждой: наилучшее (с наименьшим значением целевой функции) решение поместить в первую стаю, следующее – во вторую и т.д., M -е решение поместить в M -ю стаю, $M+1$-е снова в первую стаю и т.д. Результатом являются M стай, содержащих по m окуней каждая, так что $P = M \times m$. Первый помещенный в стаю окунь является ее лидером $x^{loc,m}, m = 1,...,M$. Лидер первой стаи одновременно является лидером всей популяции: $x^{loc,1} = x^{glob}$.

Шаг 4. *Реализация окуневого котла в каждой стае.*

Шаг 4.1. Для каждой стаи $m = 1,...,M$ выполнить следующие действия.

Передвинуть каждого окуня по направлению к лидеру стаи в окрестности ее границы:

$$x^{j,m,k} = x^{j,m} + k\frac{\left(x^{loc,m} - x^{j,m}\right)}{Nstep}, k = 0,1,...,\left[\sigma \cdot Nstep\right]; j = 1,...,s;$$

где $x^{j,m}$ – начальное положение окуня с номером j в стае с номером m; $x^{j,m,k}$ – положение окуня во время движения; $x^{loc,m}$ – положение лидера стаи с номером m; $[\cdot]$ – целая часть числа; значение параметра котла $\sigma \in [0,1;0,5]$ генерируется с помощью равномерного закона распределения на каждой итерации для каждой стаи независимо.

После всех выполненных шагов для каждого окуня найти наилучший шаг (такой шаг, на котором значение целевой функции было наименьшим), а окуню занять эту наилучшую позицию $x^{j,m,new}$:

$$x^{j,m,new} = \underset{k=0,1,...,[\sigma \cdot Nstep]}{\operatorname{argmin}} f\left(x^{j,m,k}\right), j = 1,...,s.$$

Шаг 4.2. В каждой стае определить нового лидера $x^{loc,m,new}, m = 1,...,M$.

Шаг 4.3. Упорядочить стаи по возрастанию значений целевой функции. Первой стае соответствует абсолютный лидер $x^{loc,1} = x^{glob}$, в остальных стаях находится локальный лидер $x^{loc,m}, m = 2,...,M$; стае с номером M соответствует наибольшее значение целевой функции среди лидеров.

Шаг 5. *Плавание стаи с абсолютным лидером.*

Шаг 5.1. Передвинуть каждого окуня по направлению к абсолютному лидеру стаи, двигаясь вдоль соединяющей их прямой (приближаясь, а затем удаляясь в том же направлении):

$$x^{j,1,k} = x^{j,1} + k\frac{\left(x^{glob} - x^{j,1}\right)}{Nstep}, k = 0,1,...,\left[\sigma_1 \cdot Nstep\right]; j = 1,...,s;$$

где значение параметра котла $\sigma_1 \in [1;1,5]$ генерируется с помощью равномерного закона распределения на каждой итерации.

Шаг 5.2. После всех выполненных шагов для каждого окуня стаи найти наилучший шаг (такой шаг, на котором значение целевой функции было наименьшим), а окуню занять эту наилучшую позицию $x^{j,1,new}$:

$$x^{j,1,new} = \underset{k=0,1,...,[\sigma_1 \cdot Nstep]}{\operatorname{argmin}} f\left(x^{j,1,k}\right), j = 1,...,s.$$

Шаг 5.3. В стае определить нового лидера $x^{loc,1,new} = x^{glob,new}$.

Шаг 6. *Плавание стаи с наихудшим лидером.*

Шаг 6.1. Плавание лидера. Новое положение лидера генерируется случайным образом с помощью распределения Леви:

$$x_i^{loc,M,new} = x_i^{loc,M} + \frac{\alpha}{iter} \cdot Levy_i(\lambda), \quad i = 1,\ldots,n,$$

где $x_i^{loc,M}$ – координата положения лидера стаи на текущей итерации, α – величина шага, $\lambda \in (1;3]$, а для генерации случайной величины согласно распределению Леви требуется:

- для каждой координаты x_i с помощью равномерного закона распределения на множестве $[\varepsilon; b_i - a_i]$, где $\varepsilon = 10^{-7}$ – константа различимости, сгенерировать число R_i, $i = 1,\ldots,n$:
- найти числа $\theta_i = R_i \cdot 2\pi$ и $L_i = R_i^{-\frac{1}{\lambda}}$, $i = 1,\ldots,n$, где λ – параметр распределения;
- вычислить значения координат по формулам:

$$x_i = L_i \sin\theta_i,\ i = 1,\ldots,\left[\frac{n}{2}\right];\ x_i = L_i \cos\theta_i,\ i = \left[\frac{n}{2}\right]+1,\ldots,n.$$

Если полученное значение координаты x_i не принадлежит множеству допустимых решений, т.е. $x_i \notin [a_i; b_i]$, то процесс его генерации повторяется.

Шаг 6.2. Генерировать новые позиции членов стаи $x^{j,M}$ с помощью равномерного распределения на параллелепипеде, образованном прямым произведением отрезков $[x_i^{loc,M} - \hat{x}_i, x_i^{loc,M} + \hat{x}_i]$, где $\hat{x}_i = \min\{(x_i^{loc,M} - a_i), (b_i - x_i^{loc,M})\}$.

Шаг 6.3. Реализовать окуневый котел в полученной стае.

Передвинуть каждого окуня по направлению к лидеру стаи в окрестности ее границы:

$$x^{j,M,k} = x^{j,M} + k\frac{\left(x^{loc,M,new} - x^{j,M}\right)}{Nstep}, k = 0,1,\ldots,\left[\sigma_3 \cdot Nstep\right]; j = 1,\ldots,s;$$

где значение параметра котла $\sigma_3 \in [0,1; 0,5]$ генерируется с помощью равномерного закона распределения на каждой итерации.

После всех выполненных шагов для каждого окуня найти наилучший шаг (такой шаг, на котором значение целевой функции было наименьшим), а окуню занять эту наилучшую позицию $x^{j,M,new}$:

$$x^{j,M,new} = \underset{k=0,1,\ldots,[\sigma_3 \cdot Nstep]}{\operatorname{argmin}} f\left(x^{j,M,k}\right), j = 1,\ldots,s.$$

Шаг 6.4. Определить нового лидера стаи $x^{loc,M}$.

Шаг 7. *Плавание остальных стай.*

Для всех стай с номерами $m \in \{2,\ldots,M-1\}$ выполнить следующие действия.

Шаг 7.1. Передвинуть лидера стаи по направлению к абсолютному текущему лидеру:

$$x^{loc,m,k} = x^{loc,m} + k\frac{\left(x^{glob,new} - x^{loc,m}\right)}{Nstep}, k = 0,1,...,\left[\sigma_2 \cdot Nstep\right];$$

где значение параметра $\sigma_2 \in [0,6;0,8]$ генерируется с помощью равномерного закона распределения на каждой итерации для каждой стаи независимо.

Шаг 7.2. Организовать движение остальных членов стаи параллельно лидеру:

$$x^{j,m,k} = x^{j,m} + k\frac{\left(x^{glob,new} - x^{loc,m}\right)}{Nstep}, k = 0,1,...,\left[\sigma_2 \cdot Nstep\right]; j = 1,...,s.$$

После всех выполненных шагов для каждого окуня найти наилучший шаг (такой шаг, на котором значение целевой функции было наименьшим), а окуню занять эту наилучшую позицию $x^{j,m,new}$:

$$x^{j,m,new} = \underset{k=0,1,...,\left[\sigma_2 \cdot Nstep\right]}{\operatorname{argmin}} f\left(x^{j,m,k}\right), j = 1,...,s.$$

Шаг 7.3. Каждому из членов стаи занять наилучшую позицию, достигнутую в процессе плавания. Найти нового лидера стаи $x^{loc,m}, m \in \{2,...,M-1\}$.

Шаг 8. *Нахождение нового абсолютного лидера популяции среди лидеров стай.* Среди решений, соответствующих лидерам стай, выбрать наилучшее и поместить его в множество $Pool$.

Шаг 9. *Проверка условий завершения поиска.*

Если $iter = Iter_{\max}$, то перейти к шагу 10. Иначе положить $iter = iter + 1$ и перейти к шагу 3.

Шаг 10. *Интенсивный поиск в множестве* $Pool$.

Шаг 10.1. Положить $pr = 1$.

Шаг 10.2. Выбрать три различных случайных решения $x^p_{pool}, x^q_{pool}, x^r_{pool}$ из множества $Pool$.

Шаг 10.3. Найти решение

$$x^{pq}_{pool} = \arg \min_{j=1,...,\Delta_{pr}-1} f\left(x^p_{pool} + j\,(x^q_{pool} - x^p_{pool})/\Delta_{pr}\right).$$

Шаг 10.4. Добавить решение

$$x^{new} = \arg \min_{j=1,...,\Delta_{pr}-1} f\left(x^{pq}_{pool} + j\,(x^r_{pool} - x^{pq}_{pool})/\Delta_{pr}\right)$$

в множество $Pool$. Положить $pr = pr + 1$.

Шаг 10.5. Если $pr > PR_{\max}$, то перейти к шагу 11. Иначе – к шагу 10.2.

Шаг 11. *Выбор наилучшего решения.* Среди решений в множестве $Pool$ выбрать наилучшее. Считать его приближенным решением поставленной задачи.

2.3.3. Программное обеспечение

На основе изложенного алгоритма сформирована программа поиска глобального минимума функций многих переменных [54, 128]. Среда разработки Microsoft Visual Studio 2019, язык программирования C# .

Возможности программы позволяют изучить алгоритм метода, а также влияние параметров метода на результат его работы. В список выбираемых функций включены стандартные многоэкстремальные тестовые функции двух переменных, для которых известно точное решение – положение точки (точек) максимума и соответствующее значение целевой функции (табл. П.1). Поскольку сформированная программа решает задачу поиска минимума, то перед исследуемыми тестовыми функциями поставлен знак минус.

Программа состоит из главного окна (рис. 2.3) и окна пошаговой работы метода (рис. 2.4). В главном окне метода пользователь выбирает оптимизируемую функцию, задает множество допустимых решений и параметры метода. Также в этом окне отображаются результаты работы метода. При необходимости можно создать отчет – текстовый файл с десятью сериями решений одной и той же задачи с одними и теми же значениями параметров. Внизу окна расположены кнопки «Описание алгоритма» и «Выход».

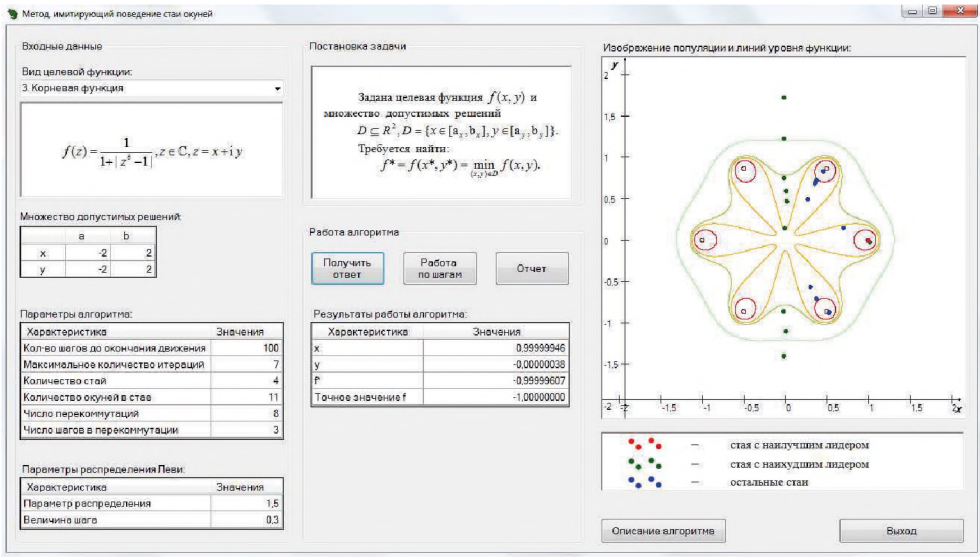


Рис. 2.3. Главное окно программы метода стаи окуней

Во время пошаговой работы отображаются схема работы метода, результаты выполнения каждого шага и окончательный результат работы метода. В центральной части окна расположен график изменения средней и наилучшей приспособленности популяций. В данном окне предусмотрена возможность выполнить N итераций (задается пользователем) по нажатию кнопки «Выполнить N итераций».

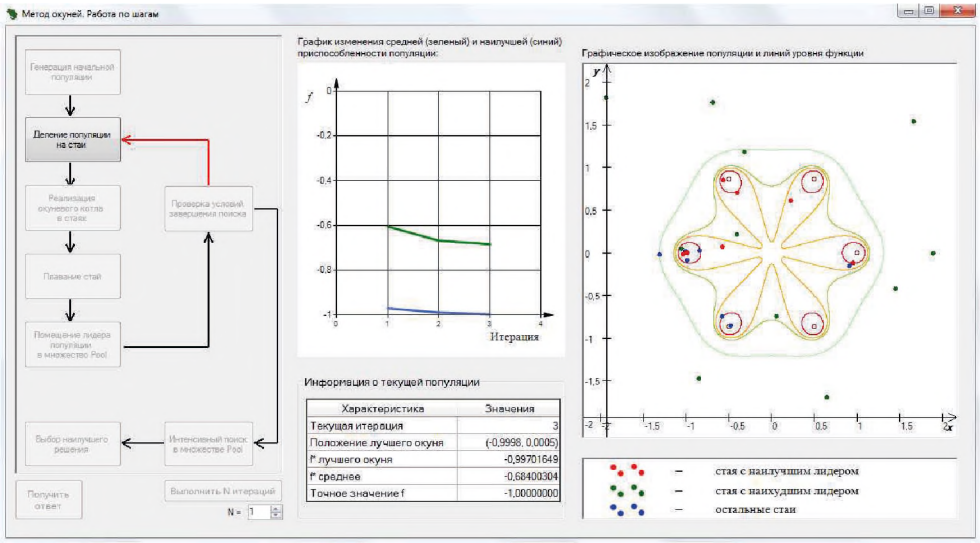


Рис. 2.4. Окно пошаговой работы метода стаи окуней

2.3.4. Тестовые примеры

Пример 2.1. Рассмотрим функцию «Кожа» с обратным знаком. Зададим множество допустимых решений $x \in [-5;5],\ y \in [-5;5]$.

Выберем следующие параметры метода:

- контролирующий параметр $NStep = 100$, определяющий количество шагов до окончания движения;
- количество стай в популяции $M = 4$;
- количество окуней в стае $s = 15$ (число членов популяции $NP = s \cdot M$);
- останавливающий параметр $Iter_{\max} = 12$, определяющий максимальное количество итераций;
- параметр $\lambda = 1,5$ распределения Леви;
- величина шага $\alpha = 0,3$;
- максимальное число перекоммутаций $PR_{\max} = 10$;
- число шагов в процедуре перекоммутации $\Delta_{pr} = 5$.

На рис. 2.5 представлена популяция на начальной ($iter = 1$), промежуточных ($iter = 5$, $iter = 9$) и конечной ($iter = 12$) итерациях.

Результаты работы метода:

- решение с наилучшим положением $(x^*; y^*) = (-3,31581608;\ -3,07450023)$;
- значение целевой функции $f(x^*; y^*) = -14,06053734$;
- отклонение от точного решения $\Delta = 0$.

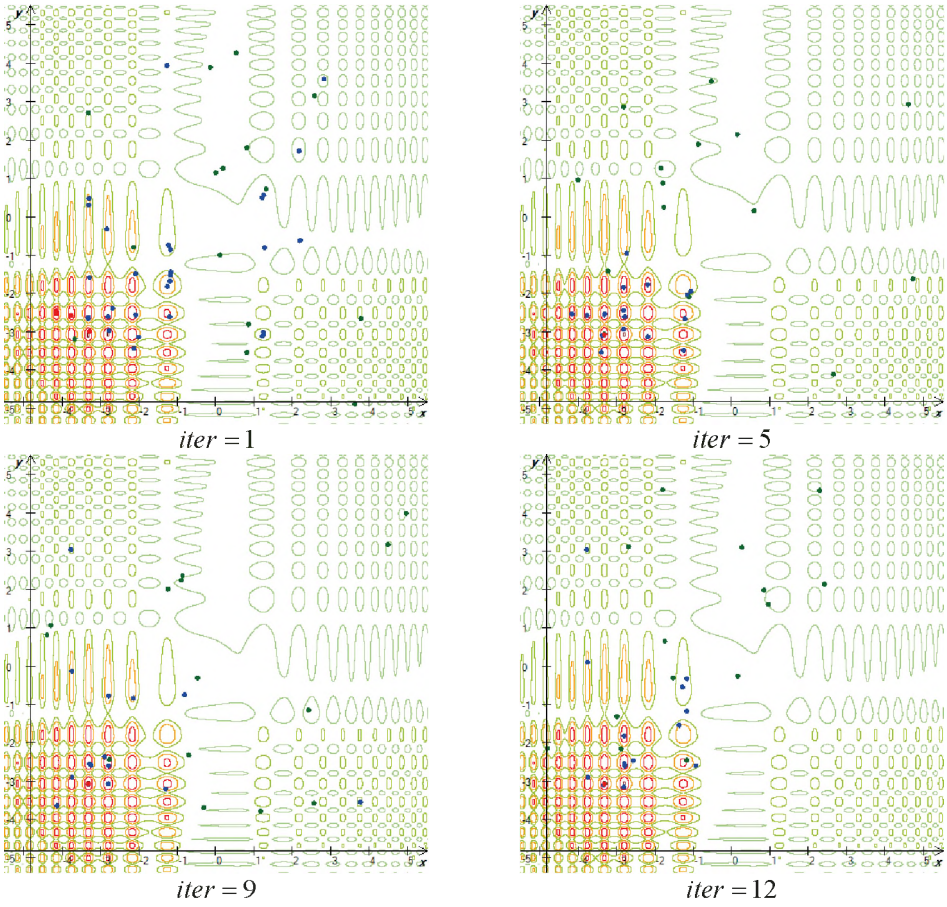


Рис. 2.5. Начальная, промежуточные и конечная популяции

Изменение наилучшего значения целевой функции при переходе от одной итерации поиска к другой представлено на рис. 2.6.

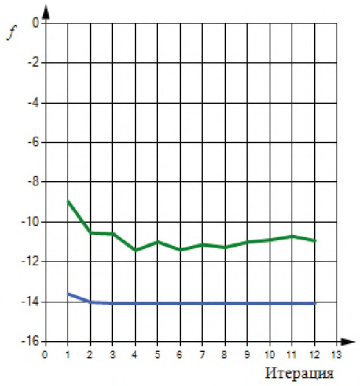


Рис. 2.6. Графики изменения среднего (зеленый) и наилучшего (синий) значений целевой функции

Пример 2.2. Рассмотрим функцию Экли с обратным знаком. Зададим множество допустимых решений $x \in [-10;10]$, $y \in [-10;10]$.

Выберем следующие параметры метода:

- контролирующий параметр $NStep = 100$, определяющий количество шагов до окончания движения;
- количество стай в популяции $M = 4$;
- количество окуней в стае $s = 11$ (число членов популяции $NP = s \cdot M$);
- останавливающий параметр $Iter_{\max} = 7$, определяющий максимальное количество итераций;
- параметр $\lambda = 1,5$ распределения Леви;
- величина шага $\alpha = 0,6$;
- максимальное число перекоммутаций $PR_{\max} = 8$;
- число шагов в процедуре перекоммутации $\Delta_{pr} = 3$.

На рис. 2.7. представлена популяция на начальной ($iter = 1$), промежуточных ($iter = 3$, $iter = 5$) и конечной ($iter = 7$) итерациях.

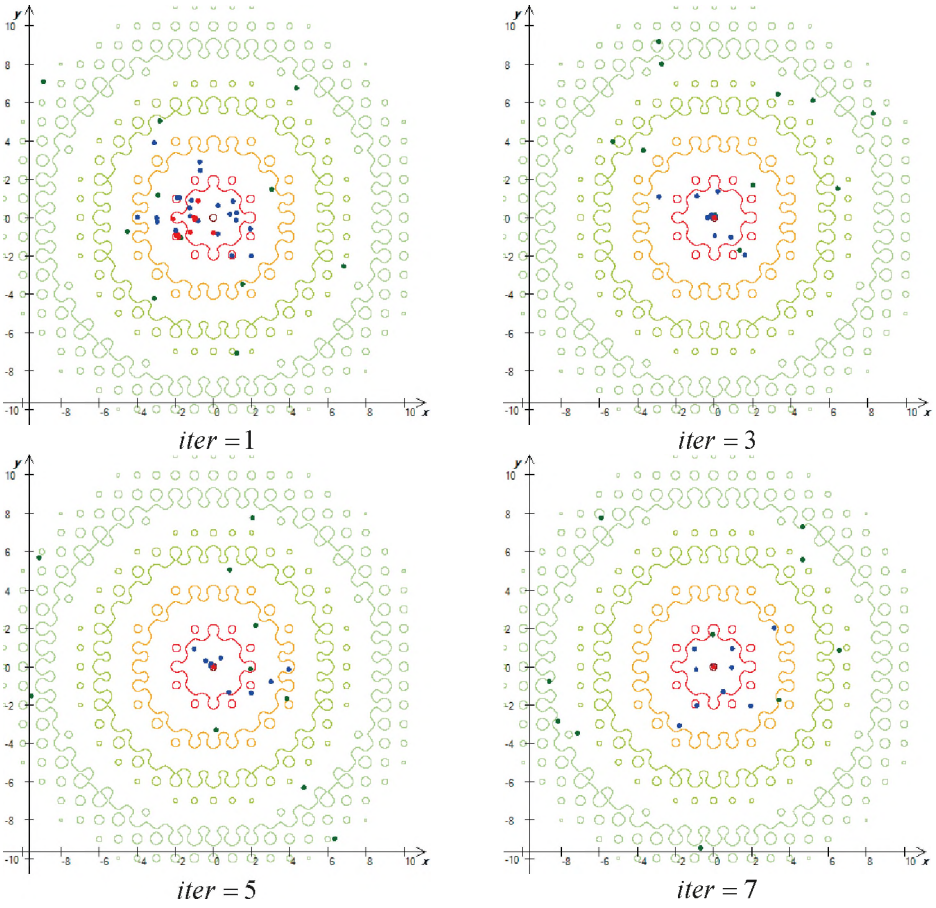


Рис. 2.7. Начальная, промежуточные и конечная популяции

Результаты работы метода:

- решение с наилучшим положением $(x^*; y^*) = (0{,}00002203;\ -0{,}00001490)$;
- значение целевой функции $f(x^*; y^*) = -19{,}99992561$;
- отклонение от точного решения $\Delta = 0$.

График изменения наилучшего значения целевой функции при переходе от одной итерации поиска к другой представлен на рис. 2.8.

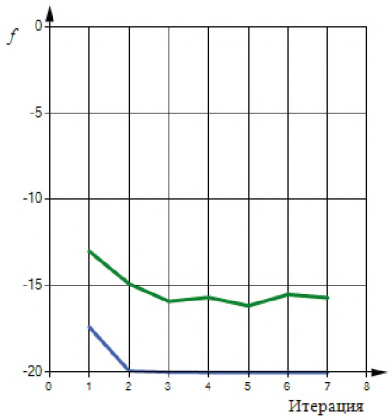


Рис. 2.8. Графики изменения среднего (зеленый) и наилучшего (синий) значений целевой функции

2.3.5. Анализ эффективности метода

АНАЛИЗ РАБОТЫ МЕТОДА ПРИ РАЗЛИЧНЫХ ЗНАЧЕНИЯХ ПАРАМЕТРОВ

В данном разделе приводится статистический анализ и сравнение результатов применения метода имитации поведения стаи окуней при различных значениях его параметров. Исследуемый метод применялся к некоторым функциям из набора тестовых функций (табл. П.1). Для каждой функции проводились серии из 100 решений одной и той же задачи с одними и теми же значениями параметров. Для полученной выборки $\left\{f^1, f^2, ..., f^{100}\right\}$ вычисляются:

- среднее значение отклонения полученного решения от точного: $\overline{\Delta f} = \frac{1}{100}\sum_{i=1}^{100} \Delta f_i$, где $\Delta f_i = |f(x^*) - f^i|$;
- наименьшее значение отклонения $\Delta f_{best} = \min_i \Delta f_i$;
- среднеквадратическое отклонение $\overline{\sigma}_f = \sqrt{\overline{S}_{100}}$, где $\overline{S}_{100} = \frac{1}{99}\sum_{i=1}^{100} (\Delta f_i - \overline{\Delta f})^2$;
- количество успехов $n_{\text{усп}}$ (попаданий лучшей точки в ε-окрестность точного решения, $\varepsilon = \frac{1}{1000}\max_{i=1,\ldots,n} |b_i - a_i|$).

Результаты, полученные для каждой функции, представлены в табл. 2.1–2.3.

Таблица 2.1. Влияние параметров метода. Параболическая функция

Параметры метода								$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NStep	$Iter_{max}$	*M*	*s*	PR_{max}	Δ_{pr}	α	λ				
100	4	4	3	8	3	0,6	1,5	0,000253	0	0,000383	72
100	6	4	3	8	3	0,6	1,5	0,000209	0	0,000520	82
100	10	4	3	8	3	0,6	1,5	0,000045	0	0,000206	96
100	4	4	7	8	3	0,6	1,5	0,000003	0	0,000007	100
100	3	4	1	8	3	0,6	1,5	0,000018	0	0,000027	100
100	4	4	11	8	3	0,6	1,5	0	0	0,000001	100

Таблица 2.2. Влияние параметров метода. Функция «Кожа»

Параметры метода								$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NStep	$Iter_{max}$	*M*	*s*	PR_{max}	Δ_{pr}	α	λ				
100	20	4	3	10	5	0,6	1,5	0,076972	0	0,125540	47
100	10	4	3	10	5	0,6	1,5	0,292875	0,000002	0,586289	30
100	12	4	5	10	5	0,3	1,5	0,033539	0	0,056396	56
100	10	4	15	10	5	0,6	1,5	0,004537	0	0,022553	92
100	10	4	15	10	5	0,3	1,5	0,028184	0,000001	0,068515	66
100	12	4	15	10	5	0,3	1,5	0,001513	0	0,007861	97
100	5	4	15	10	5	0,3	1,5	0,005969	0	0,014755	86
100	5	4	15	10	5	0,3	1,1	0,004423	0	0,012657	89
100	5	4	15	20	5	0,3	1,5	0,006161	0	0,015456	87

Таблица 2.3. Влияние параметров метода. Функция Экли

Параметры метода								$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{усп}$
NStep	$Iter_{max}$	*M*	*s*	PR_{max}	Δ_{pr}	α	λ				
100	4	4	3	8	3	0,6	1,5	0,235162	0,000021	0,608738	70
100	10	3	3	8	3	0,6	1,5	0,194856	0,000256	0,617575	89
100	4	4	5	8	3	0,6	1,5	0,023385	0,000256	0,031809	93
100	4	4	11	8	3	0,6	1,5	0,002806	0,000065	0,003627	100
100	7	4	11	8	3	0,6	1,5	0,000106	0	0,000197	100

Анализ работы метода, имитирующего поведение стаи окуней, показал, что результат работы алгоритма существенно зависит от размера популяции.

При большой популяции за малое количество итераций, как правило, находится глобальный экстремум, причем не все окуни оказываются в точке глобального минимума, часть популяции остается в локальных минимумах.

Таким образом, данный метод позволяет находить не только глобальный минимум, но и некоторые локальные минимумы.

Алгоритм имеет несколько недостатков. На функциях типа «Кожа» производится большое количество пересчета значений целевой функции, что влияет на скорость выполнения итераций. Также при малых популяциях на функциях, у которых точки глобального минимума находятся на большом расстоянии друг от друга, есть риск того, что популяции «застрянут» в локальных минимумах.

Контролирующий параметр *NStep*, определяющий количество шагов до окончания движения. Для многоэкстремальных функций с большим расстоянием между экстремумами следует полагать $NStep = 100,...,200$ при условии $NP \geq 100$. Для функций с малым количеством локальных минимумов достаточно положить $NStep = 100$.

Максимальное число итераций $Iter_{\max}$, определяющее максимальное количество итераций. Учитывая суть алгоритма, необходимо задавать $Iter_{\max} \geq 3$ в зависимости от количества минимумов функции на множестве допустимых решений.

Для многоэкстремальных функций рекомендуется использовать:

- для малых популяций ($NP = 3 \leq s \cdot M < 20$) $Iter_{\max} = 4,...,20$;
- для средних популяций ($NP = 20 \leq s \cdot M \leq 40$) $Iter_{\max} = 4,...,15$;
- для больших популяций ($NP = s \cdot M > 40$) $Iter_{\max} = 4,...,12$.

Для функций с малым количеством локальных минимумов рекомендуется использовать:

- для малых популяций ($NP = s \cdot M \leq 20$) $Iter_{\max} = 4,...,10$;
- для средних популяций ($NP = 20 \leq s \cdot M \leq 40$) $Iter_{\max} = 3,...,8$;
- для больших популяций ($NP = s \cdot M > 20$) $Iter_{\max} = 3,...,7$.

Количество стай в популяции M. Согласно описанной стратегии метода рекомендуется выбирать $M \geq 3$.

Количество окуней в стае s следует выбирать в зависимости от количества минимумов функции и максимального числа итераций $Iter_{\max}$.

Максимальное число перекоммутаций $PR_{\max}$. Рекомендуется использовать $PR_{\max} = 5,...,20$.

Число шагов в процедуре перекоммутации $\Delta_{pr} = 5,...,30$.

Параметр распределения Леви λ. Рекомендуется использовать $\lambda \in (1;3]$.

Величина шагов α **для стаи с наихудшим лидером**. Рекомендуется использовать $\alpha \in [0,3;0,7]$.

2.4. МЕТОД, ИМИТИРУЮЩИЙ ПОВЕДЕНИЕ СТАИ СИНИЦ

2.4.1. Стратегия поиска решения

Рассматривается решение задачи (2.1), в которой дана целевая функция $f(x) = f(x_1, x_2, ..., x_n)$, определенная на множестве допустимых решений $D \subseteq R^n$.

Требуется найти условный глобальный минимум функции $f(x)$ на множестве D, т.е. такую точку $x^* \in D$, что

$$f(x^*) = \min_{x \in D} f(x),$$

где $x = (x_1, x_2, ..., x_n)^T$, $D = \{x \mid x_i \in [a_i, b_i], i = 1, 2, ..., n\}$.

Метод, имитирующий поведение стаи синиц (Tomtit Flock Optimization – TFO) – гибридный алгоритм поиска глобального условного экстремума функций многих переменных, относящийся как к методам «роевого» интеллекта, так и к биоинспирированным методам [134].

Синицы, объединяясь в стаи, подчиняются командам вожака стаи, обладая некоторой свободой выбора способа поиска пищи. От других птиц и животных их отличает особенная сплоченность, согласованность коллективных действий, интенсивность использования найденного источника пищи до ее исчезновения, четкое и дружное исполнение команд, общих для членов стаи.

При решении задачи поиска глобального условного минимума функции используются конечные наборы $I = \{x^j = (x_1^j, x_2^j, ..., x_n^j)^T, j = 1, 2, ..., NP\} \subset D$ возможных решений, называемые популяциями, где x^j – особь (синица) с номером j, NP – размер популяции.

Начальная стая (на нулевой итерации, $k = 0$) генерируется на множестве допустимых решений D при помощи равномерного закона распределения. Для каждой синицы (решения) подсчитывается значение целевой функции $f(x)$. Наилучшее по величине функции решение связывается с положением вожака стаи. Он не участвует в процедуре поиска пищи (нахождении экстремума целевой функции) на текущей итерации, а только ожидает ее результатов с целью обработки и дальнейшего хранения в матрице памяти $\left(\begin{array}{ccc|c} x_1^1 & \cdots & x_n^1 & f(x^1) \\ \vdots & \ddots & \vdots & \vdots \\ x_1^K & \cdots & x_n^K & f(x^K) \end{array}\right)$ размеров $K \times (n+1)$, где K – максимальное число записей. В матрицу заносится наилучший достигнутый результат на каждой k-й итерации вплоть до ее окончательного заполнения. Число K определяет количество итераций, выполняемых алгоритмом за один проход. Решения, хранящиеся в матрице, упорядочиваются так, что первой записи соответствует наилучшее решение $(x^1, f(x^1))$, а последующим соответствующие убыванию (невозрастанию) значений целевой функции. При достижении K записей производится очистка памяти, а наилучшее решение из полученных на данном проходе, помещается в специальное множество *Pool*, в котором хранятся наилучшие результаты проходов.

Новое положение вожака стаи генерируется случайным образом с помощью распределения Леви:

$$x_i^{1,k+1} = x_i^{1,k} + \frac{\alpha}{k+1} \cdot Levy_i(\lambda), \quad i = 1,\ldots,n,$$

где $x_i^{1,k}$ – координата положения вожака стаи на k-й итерации, α – величина шага, $\lambda \in (1,3]$. Исследования поведения животных показали, что распределение Леви наиболее точно описывает траекторию движения птиц и насекомых. Благодаря «тяжелым хвостам» распределения Леви велика вероятность значительных отклонений случайной величины от среднего значения. Поэтому согласно приведенному выражению возможны достаточно большие приращения по каждой координате вектора $x^{j,k}$. Если новое значение координаты не принадлежит множеству допустимых решений, т.е. $x_i \notin [a_i, b_i]$, то процесс его генерации повторяется.

Этот процесс описывает перелет вожака стаи с одного места на другое. За ним по его команде совершают перелет остальные синицы. Их положение моделируется с помощью равномерного распределения на параллелепипеде, центр которого определяется положением вожака, а длины сторон равны $r^k(b_i - a_i)$, $i = 1,...,n$, где $r^{k+1} = \gamma r^k$, $r^0 = 1$, γ – коэффициент уменьшения множества поиска (если $r^k < \varepsilon$, проход завершается и начинается новый). При достижении K итераций или при выполнении условия $r^k < \varepsilon$ проход считается законченным, счетчик числа проходов увеличивается на единицу: $p = p + 1$, а параметр, определяющий размеры следующего множества поиска $r^0 = \eta^p$, где η – коэффициент восстановления множества поиска. Найденные координаты определяют начальные условия для поиска на текущей итерации (начальное положение каждой синицы).

Предполагается, что каждая j-я особь (синица) обладает памятью, в которой хранятся:

- текущее число итераций k;
- текущее положение $x^{j,k}$ и соответствующее значение $f(x^{j,k})$ целевой функции;
- наилучшее положение x^{best} и соответствующее значение $f(x^{best})$ целевой функции в популяции;
- наилучшее положение $x^{j,best}$ особи за все выполненные итерации и соответствующее значение $f(x^{j,best})$ целевой функции;
- наилучшее положение $x^{j,local}$ среди особей, находящихся в окрестности j-й особи радиуса ρ, и соответствующее значение $f(x^{j,local})$ целевой функции.

Траектория движения каждой особи (для всех $j = 1,...,NP$) на промежутке $[0;T]$, в ходе которого реализуется поиск на текущей итерации, описывается решением стохастического дифференциального уравнения

$$dx^{j,k} = \boldsymbol{f}(x^{j,k}(t))dt + \sigma(x^{j,k}(t))dW + dq,\ x^{j,k}(0) = x^{j,k},\ j = 2,...,NP,$$

где $W(t)$ – стандартный винеровский случайный процесс, T – время, выделяемое вожаком стаи для поиска членам стаи на текущей итерации, dq – пуассоновская составляющая, которая может быть записана в виде:

$$dq = \sum_p \theta_p \delta(t - \tau_p)dt,$$

$\delta(t)$ – асимметричная дельта-функция, τ_p – моменты скачков. В случайные моменты времени τ_p положение синицы получает случайные приращения θ_p, образующие пуассоновский поток событий заданной интенсивности μ. Решение уравнения определяет траектории движения синиц, реализующие процедуру диффузионного поиска со скачками.

Вектор сноса $\boldsymbol{f}(x^{j,k}(t))\ \ \forall t \in [0;T]$ определяется выражением

$$\boldsymbol{f}(x^{j,k}(t)) = c_1 r_1 [x^{best} - x^{j,k}(t)],$$

т.е. учитывает информацию о наилучшем решении в популяции (положении вожака стаи).

Матрица диффузии $\sigma(x^{j,k}(t))$ учитывает информацию о наилучшем решении, полученном данной особью за все прошедшие итерации, о наилучшем решении в окрестности текущего решения, определяемой радиусом ρ:

$$\sigma(t, x^{j,k}(t)) = c_2 r_2 [x^{j,best} - x^{j,k}(t)] + c_3 r_3 [x^{j,local} - x^{j,k}(t)],$$

где c_1, c_2, c_3 – коэффициенты влияния; r_1, r_2, r_3 – случайные параметры, равномерно распределенные на отрезке $[0;1]$. Параметр c_2 характеризует процесс забывания о своей истории поиска, а параметр c_3 описывает влияние лидера среди соседей.

Решение стохастического дифференциального уравнения – случайный процесс, траектории которого имеют участки непрерывного изменения, прерывающиеся скачками заданной интенсивности. Оно описывает движение синицы, сопровождаемое относительно короткими прыжками. Это решение может быть найдено с помощью численного интегрирования с некоторым шагом h. Если какая-либо координата попала на границу области поиска или вышла за ее пределы, то она берется равной значению на этой границе. В качестве нового положения синицы $x^{j,k,search}$ выбирается наилучшее, достигнутое в ходе текущей итерации. Среди всех новых положений синиц выбирается наилучшее, оно записывается в матрицу памяти и отождествляется с окончательной позицией вожака стаи на текущей итерации, после чего начинается следующая итерация с процедуры поиска нового положения вожака стаи и начальных положений членов стаи относительно него. Метод завершает работу по достижении максимального числа проходов P.

Предлагаемый гибридный метод использует идеи, использованные при реализации известных метаэвристических алгоритмов глобальной оптимизации [54, 74, 76, 114, 120, 121, 157, 158]:

- эволюционных методов при создании начальной популяции;
- метода имитации поведения кукушек для моделирования скачка вожака стаи на основе полетов Леви;
- метода частиц в стае для описания взаимодействия особей между собой;
- модифицированного метода искусственных иммунных систем для обновления популяции;
- метода Лууса для обновления множества поиска при завершении очередного прохода.

Общая схема работы метода, имитирующего поведение стаи синиц, изображена на рис. 2.9, а его иллюстрация на рис. 2.10.

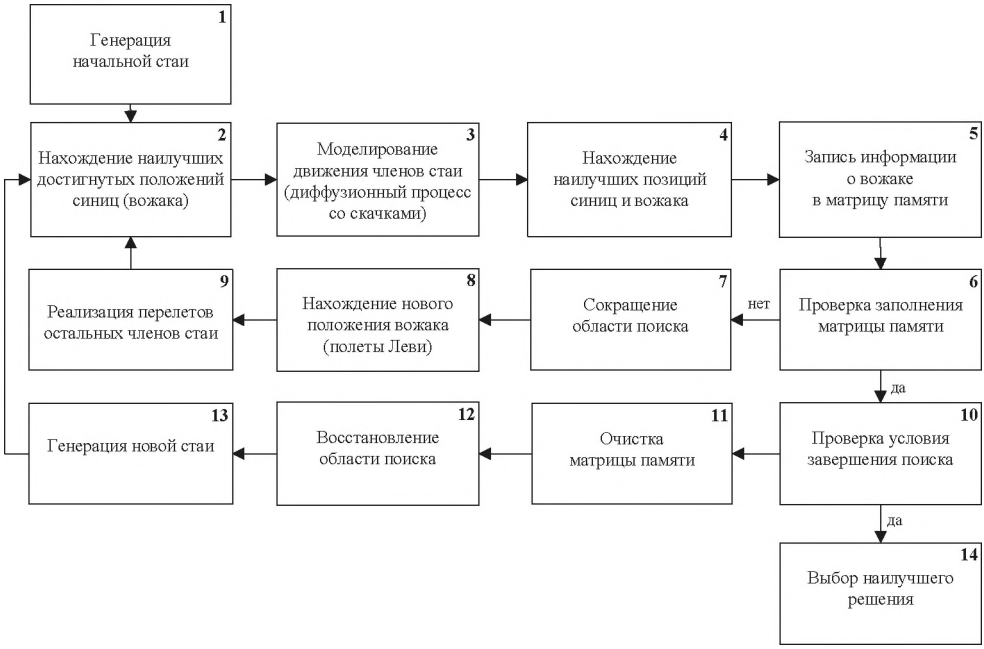


Рис. 2.9. Общая схема работы метода, имитирующего поведение стаи синиц

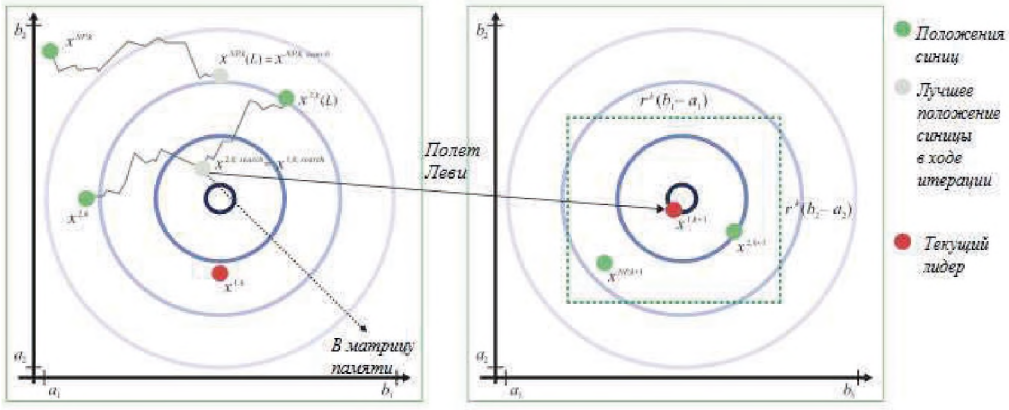


Рис. 2.10. Иллюстрация работы метода, имитирующего поведение стаи синиц

2.4.2. Алгоритм решения задачи

Шаг 1. *Создание начальной популяции.*
Шаг 1.1. Задать параметры метода:

- размер популяции NP;
- шаг при передвижении вожака стаи α;
- параметр сокращения множества поиска γ;
- параметр восстановления множества поиска η;

- параметр распределения Леви λ;
- величина радиуса ρ, определяющая окрестность члена стаи;
- параметры c_1, c_2, c_3, определяющие вектор сноса и матрицу диффузии в уравнении, описывающем процесс поиска членами стаи;
- максимальное число записей в матрице памяти K;
- шаг интегрирования дифференциального уравнения h;
- максимальное число дискретных шагов L;
- время, затрачиваемое на поиск членами стаи $T = Lh$;
- максимальное число проходов P;
- параметр интенсивности скачков μ.

Положить $p = 0$ (число проходов), $r^0 = 1$.

Шаг 1.2. Создать популяцию (стаю) $I = \{x^j = (x_1^j, x_2^j, ..., x_n^j)^T, j = 1, 2, ..., NP\} \subset D$ из NP решений (синиц) со случайно сгенерированными координатами x_i из промежутка $[a_i, b_i]$ с использованием равномерного закона распределения:

$$x_i^j = a_i + rand_i[0;1] \cdot (b_i - a_i),\ i = 1, \ldots, n;\ j = 1, \ldots, NP,$$

где $rand_i[0;1]$ – случайная величина, определяемая равномерным законом распределения на отрезке $[0;1]$.

Шаг 2. *Движение членов стаи.* Реализация процедуры диффузионного поиска со скачками.

Шаг 2.1. Положить $k = 0$ (счетчик числа итераций).

Шаг 2.2. Для каждого члена популяции вычислить значение целевой функции: $f(x^{1,k}), ..., f(x^{NP,k})$. Упорядочить членов стаи по неубыванию значений целевой функции. Наилучшему значению соответствует решение $x^{1,k}$, отождествляемое с положением вожака стаи.

Шаг 2.3. Обработать текущую информацию о членах стаи.

Для $j = 1$ принять: наилучшее положение $x^{best} = x^{1,k}$ и соответствующее значение $f(x^{best}) = f(x^{1,k})$ целевой функции в популяции.

Для всех остальных членов стаи ($j = 2, ..., NP$) найти:

- наилучшее положение $x^{j,best}$ особи за все выполненные итерации и соответствующее значение $f(x^{j,best})$ целевой функции;
- наилучшее положение $x^{j,local}$ среди особей, находящихся в окрестности j-й особи радиуса ρ, и соответствующее значение $f(x^{j,local})$ целевой функции.

Шаг 2.4. Для каждого $j = 2, ..., NP$ найти численное решение стохастического дифференциального уравнения на промежутке $[0, Lh]$ с шагом h, применяя метод Эйлера–Маруямы.

Шаг 2.4.1. Положить $x^{j,k}(0) = x^{j,k}$ и $l = 0$.

Шаг 2.4.2. Найти диффузионную составляющую решения

$$\tilde{x}^{j,k}(l+1) = x^{j,k}(l) + h\boldsymbol{f}(x^{j,k}(l)) + \sqrt{h}\sigma(x^{j,k}(l))\xi,$$

где $\boldsymbol{f}(x^{j,k}(l)) = c_1 r_1[x^{best} - x^{j,k}(l)]$, $\sigma(x^{j,k}(l)) = c_2 r_2[x^{j,best} - x^{j,k}(l)] + c_3 r_3[x^{j,local} - x^{j,k}(l)]$,

r_1, r_2, r_3 – случайные параметры, равномерно распределенные на отрезке $[0;1]$, ξ – случайная величина, имеющая стандартное нормальное распределение с нулевым математическим ожиданием и единичной дисперсией. Для ее моделирования можно воспользоваться методом Бокса–Мюллера:

$$\xi = \sqrt{-2\ln\alpha_1}\cos 2\pi\alpha_2 \text{ или } \xi = \sqrt{-2\ln\alpha_1}\sin 2\pi\alpha_2,$$

в котором α_1 и α_2 – независимые равномерно распределенные на интервале $(0;1)$ случайные величины.

Шаг 2.4.3. Проверить условие скачка: $\beta \le \mu h$, где β – равномерно распределенная на интервале $(0;1)$ случайная величина. В процессе интегрирования следует проверять принадлежность решения множеству допустимых решений: если какая-либо координата решения попала на границу области поиска или вышла за ее пределы, то она берется равной этой границе.

Если оно выполнено, то положить

$$x^{j,k}(l+1) = \tilde{x}^{j,k}(l+1) + \theta,$$

где θ – случайное приращение, координаты которого моделируется согласно равномерному закону распределения: $\theta_i \in [-\Delta_i, \Delta_i]$, $\Delta_i = \min[(b_i - \tilde{x}_i^{j,k}(l+1), (\tilde{x}_i^{j,k}(l+1) - a_i)]$.

Иначе положить $x^{j,k}(l+1) = \tilde{x}^{j,k}(l+1)$.

Шаг 2.4.4. Проверить условие окончания процесса движения члена стаи.

Если $l < L$, положить $l = l+1$ и перейти к шагу 2.4.2. Иначе перейти к шагу 2.5.

Шаг 2.5. Для каждого члена стаи ($j = 2,...,NP$) найти наилучшее решение среди полученных в ходе интегрирования: $x^{j,k}(0), x^{j,k}(1),..., x^{j,k}(L)$. Обозначить его $x^{j,k,search}$, $j = 2,...,NP$.

Шаг 2.6. Среди положений $x^{1,k}, x^{2,k,search},..., x^{NP,k,search}$ синиц (решений) выбрать наилучшее. Записать его в матрицу памяти и считать положением вожака стаи после выполнения текущей итерации $x^{1,k,search}$.

Шаг 2.7. Проверить условие окончания прохода. Если $k = K$ (матрица памяти заполнена полностью) или $r^k < \varepsilon_1$, то проход завершить, из матрицы памяти выбрать наилучшее решение, поместить его в множество *Pool* и перейти к шагу 3. Если $k < K$, положить $r^{k+1} = \gamma r^k$, $k = k+1$ и перейти к шагу 2.8.

Шаг 2.8. Найти новое положение вожака стаи

$$x^{1,k} = x^{1,k,search} + \frac{\alpha}{k+1} Levy(\lambda).$$

Для генерации случайной величины согласно распределению Леви требуется:

- для каждой координаты с помощью равномерного закона распределения на множестве $[\varepsilon; b_i - a_i]$, где $\varepsilon = 10^{-7}$ – константа различимости, сгенерировать число $R_i, i = 1,...,n$;
- найти числа $\theta_i = R_i \cdot 2\pi$ и $L_i = (R_i + \varepsilon)^{-\frac{1}{\lambda}}$, $i = 1,...,n$, где λ – параметр распределения;
- вычислить значения координат по формулам:

$$x_i = L_i \sin\theta_i,\ i = 1,\ldots,\left[\frac{n}{2}\right];\ x_i = L_i \cos\theta_i,\ i = \left[\frac{n}{2}\right]+1,\ldots,n.$$

Если полученное значение координаты не принадлежит множеству допустимых решений, т.е. $x_i \notin [a_i, b_i]$, то процесс его генерации повторяется, но не более 10 раз.

Если после 10 неудачных повторных генераций $x_i \notin [a_i, b_i]$, то сгенерировать значение x_i с помощью равномерного распределения на отрезке $[a_i, b_i]$.

Шаг 2.9. Реализовать перелет остальных синиц.

Положение синиц ($j = 2,\ldots,NP$) моделируется с помощью равномерного распределения на параллелепипеде (см. шаг 1.2), центр которого определяется положением $x^{1,k}$ вожака стаи (см. шаг 2.8), а длины сторон равны $r^k(b_i - a_i)$, $i = 1,\ldots,n$.

Если значение координаты синицы находится вне множества допустимых решений, то сгенерировать новое значение с помощью равномерного распределения:

- на множестве $\left[a_i, x_i^{1,k}\right]$ при $x_i^{j,k} < a_i$, $j = 2,\ldots,NP$;
- на множестве $\left[x_i^{1,k}, b_i\right]$ при $x_i^{j,k} > b_i$, $j = 2,\ldots,NP$.

Перейти к шагу 2.2.

Шаг 3. *Проверка условий окончания процесса поиска.* Если $p = P$, процесс завершить и перейти к шагу 4. Если $p < P$, очистить матрицу памяти, счетчик числа проходов увеличить на единицу: $p = p + 1$, положить $r^0 = \eta^p$, где η – коэффициент восстановления множества поиска.

Генерировать новую стаю синиц. Для этого выбрать из множества *Pool* наилучшее решение: $x^{1,0}$. Положение остальных синиц ($j = 2,\ldots,NP$) моделируется с помощью равномерного распределения на параллелепипеде (см. шаг 1.2), центр которого определяется положением $x^{1,0}$ вожака стаи, а длины сторон равны $r^0(b_i - a_i)$, $i = 1,\ldots,n$.

Если значение координаты синицы находится вне множества допустимых решений, то сгенерировать новое значение с помощью равномерного распределения:

- на множестве $\left[a_i, x_i^{1,0}\right]$ при $x_i^{j,k} < a_i$, $j = 2,\ldots,NP$;
- на множестве $\left[x_i^{1,0}, b_i\right]$ при $x_i^{j,k} > b_i$, $j = 2,\ldots,NP$.

Перейти к шагу 2.

Шаг 4. *Выбор решения задачи.* В качестве приближенного решения задачи выбрать из множества *Pool* наилучшее решение.

2.4.3. Программное обеспечение

На основе изложенного алгоритма разработана программа поиска глобального минимума функций [54, 134]. Среда разработки Microsoft Visual Studio 2019, язык программирования C#.

Возможности программы позволяют изучить алгоритм метода, а также влияние параметров метода на результат его работы. В список выбираемых функций включены стандартные многоэкстремальные тестовые функции двух переменных, для которых известно точное решение (табл. П.1). Поскольку сформированная программа решает задачу поиска минимума, то перед исследуемыми тестовыми функциями по-

ставлен знак минус. Программа состоит из главного окна (рис. 2.11) и окна пошаговой работы метода (рис. 2.12).

В главном окне метода пользователь выбирает оптимизируемую функцию, задает множество допустимых решений и параметры метода. Также в этом окне отображаются результаты работы метода. При необходимости можно создать отчет – текстовый файл с тремя сериями решений одной и той же задачи с одними и теми же значениями параметров. Внизу окна расположены кнопки «Описание алгоритма» и «Выход».

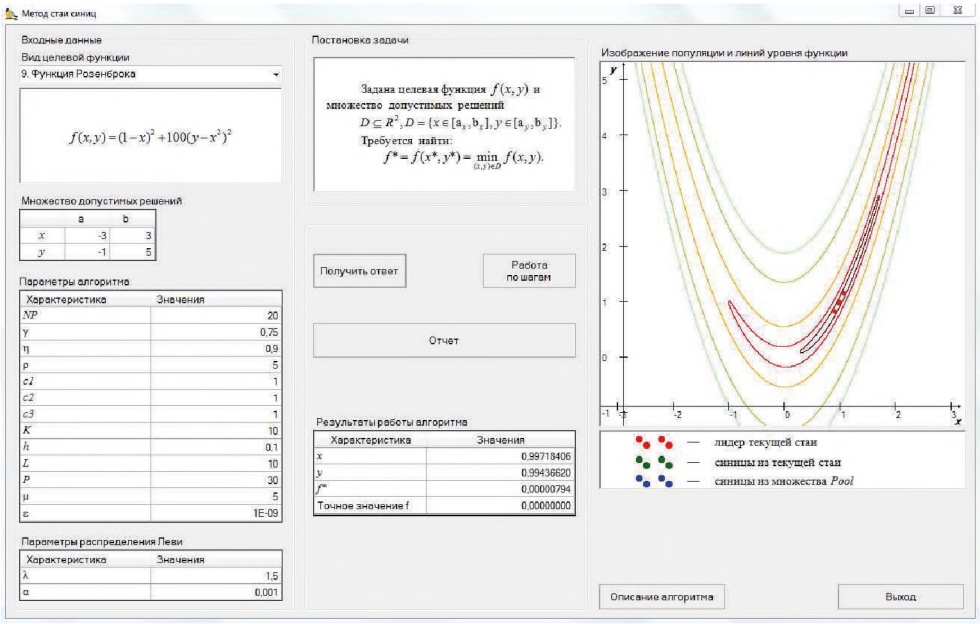


Рис. 2.11. Главное окно программы метода стаи синиц

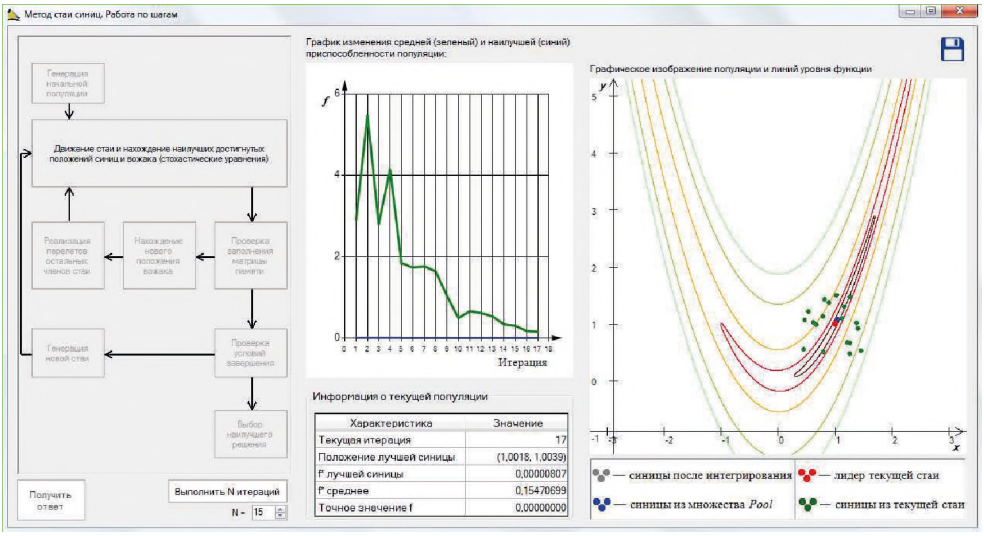


Рис. 2.12. Окно пошаговой работы метода стаи синиц

Во время пошаговой работы отображаются схема работы метода, результаты выполнения каждого шага и окончательный результат работы метода. В центральной части окна расположен график изменения средней и наилучшей популяций. В данном окне предусмотрена возможность выполнить N итераций по нажатию кнопки «Выполнить N итераций». Число N задается пользователем в ходе решения.

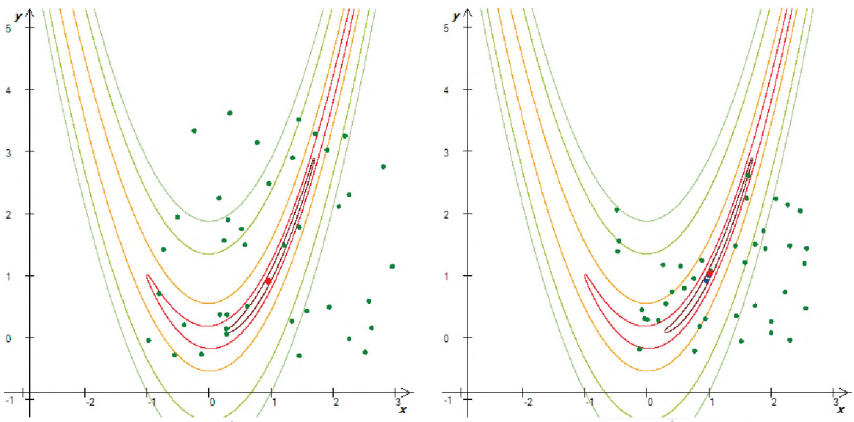
2.4.4. Тестовые примеры

Пример 2.3. Рассмотрим функцию Розенброка с обратным знаком (табл. П.1). Зададим множество допустимых решений $x \in [-3;3], y \in [-1;5]$.

Выберем следующие параметры алгоритма:

- размер популяции $NP = 40$;
- шаг при передвижении вожака стаи $\alpha = 0,001$;
- параметр сокращения множества поиска $\gamma = 0,95$;
- параметр восстановления множества поиска $\eta = 0,89$;
- параметр распределения Леви $\lambda = 1,5$;
- величина радиуса $\rho = 6$, определяющая окрестность члена стаи;
- параметры $c_1, c_2, c_3 = 10$, определяющие вектор сноса и матрицу диффузии в уравнении, описывающем процесс поиска членами стаи;
- максимальное число записей в матрице памяти $K = 6$;
- шаг интегрирования дифференциального уравнения $h = 0,1$;
- максимальное число дискретных шагов $L = 10$;
- время, затрачиваемое на поиск членами стаи $T = Lh = 1$;
- максимальное число проходов $P = 10$;
- параметр интенсивности скачков $\mu = 2$.

На рис. 2.13 представлена популяция на начальной ($p = 1$), промежуточных ($p = 4, p = 8$) и конечной ($p = 10$) итерациях.



$p = 1$ $\qquad$ $p = 4$

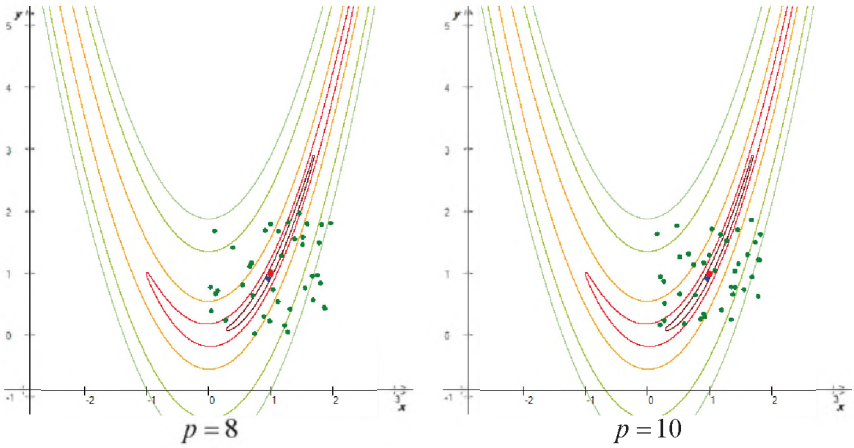


Рис. 2.13. Начальная, промежуточные и конечная итерации

Результаты работы метода:

- наилучшее решение $(x^*; y^*) = (0,9999; 0,9997)$;
- значение целевой функции $f(x^*, y^*) = 0$;
- отклонение от точного решения $\Delta = 0$.

Изменение наилучшего значения целевой функции при переходе от одной итерации поиска к другой представлено на рис. 2.14.

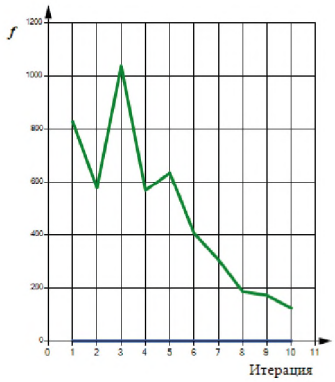


Рис. 2.14. Графики изменения среднего (зеленый) и наилучшего (синий) значений целевой функции

Пример 2.4. Рассмотрим мультифункцию с обратным знаком. Зададим множество допустимых решений $x, y \in [-2; 2]$. Выберем следующие параметры алгоритма:

- размер популяции $NP = 20$;
- шаг при передвижении вожака стаи $\alpha = 0,001$;
- параметр сокращения множества поиска $\gamma = 0,6$;
- параметр восстановления множества поиска $\eta = 0,9$;
- параметр распределения Леви $\lambda = 1,5$;
- величина радиуса $\rho = 4$, определяющая окрестность члена стаи;

- параметры $c_1, c_2, c_3 = 3$, определяющие вектор сноса и матрицу диффузии в уравнении, описывающем процесс поиска членами стаи;
- максимальное число записей в матрице памяти $K = 3$;
- шаг интегрирования дифференциального уравнения $h = 0,1$;
- максимальное число дискретных шагов $L = 4$;
- время, затрачиваемое на поиск членами стаи $T = Lh = 0,4$;
- максимальное число проходов $P = 15$;
- параметр интенсивности скачков $\mu = 3$.

На рис. 2.15 представлена популяция на начальной ($p = 1$), промежуточных ($p = 6, p = 10$) и конечной ($p = 15$) итерациях.

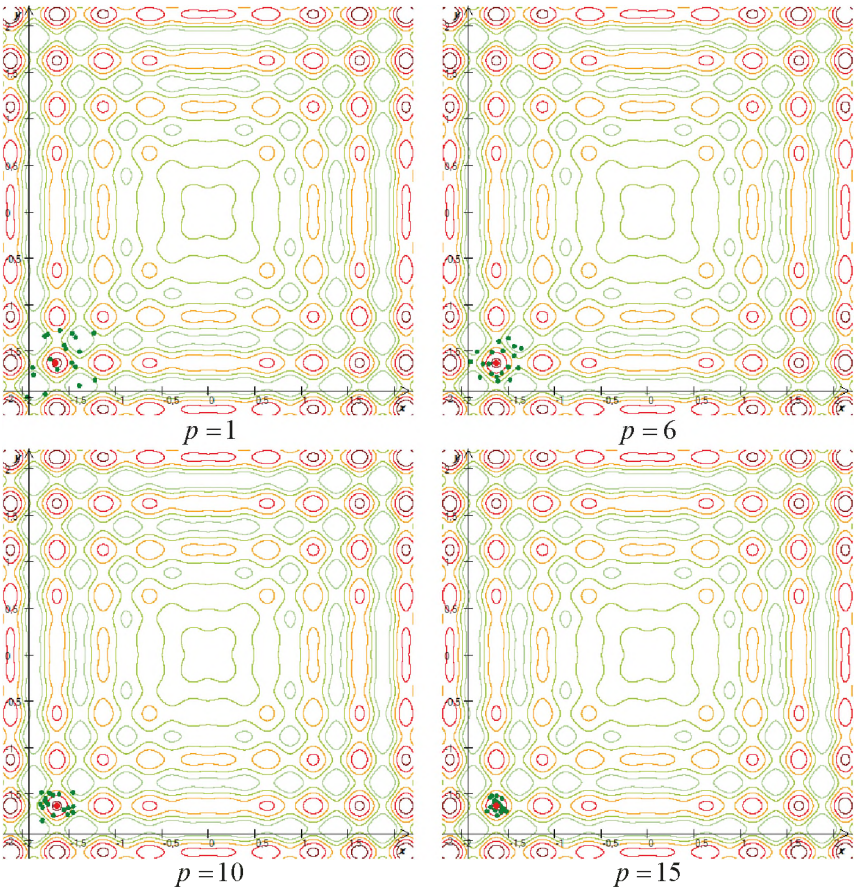


Рис. 2.15. Начальная, промежуточные и конечная итерации

Результаты работы метода:

- наилучшее решение $(x^*; y^*) = (-1,6285;\ -1,6288)$;
- значение целевой функции $f(x^*, y^*) = -4,25386952$;
- отклонение от точного решения $\Delta = 1,8 \cdot 10^{-5}$.

Изменение наилучшего значения целевой функции при переходе от одной итерации поиска к другой представлено на рис. 2.16.

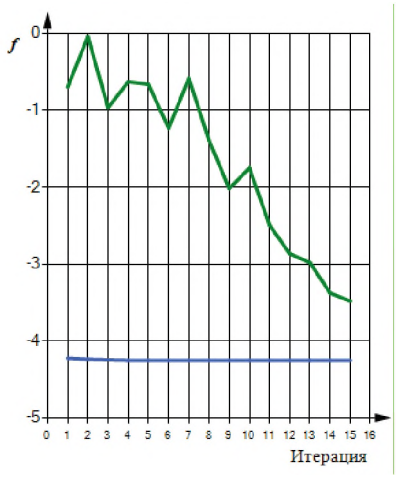


Рис. 2.16. Графики изменения среднего (зеленый) и наилучшего (синий) значений целевой функции

2.4.5. Анализ эффективности метода

АНАЛИЗ РАБОТЫ МЕТОДА ПРИ РАЗЛИЧНЫХ ЗНАЧЕНИЯХ ПАРАМЕТРОВ

В данном разделе приводится статистический анализ и сравнение работы метода стаи синиц при различных значениях его параметров. Исследуемый метод применялся к некоторым функциям из набора тестовых функций (табл. П.1). Для каждой функции проводились серии из 100 решений одной и той же задачи с одними и теми же значениями параметров.

Для полученной выборки $\left\{f^1, f^2, ..., f^{100}\right\}$ вычислялись:

- среднее значение отклонения полученного решения от точного:

$$\overline{\Delta f} = \frac{1}{100}\sum_{i=1}^{100} \Delta f_i,$$

где $\Delta f_i = |f(x^*) - f^i|$;

- наименьшее значение отклонения $\Delta f_{best} = \min_i \Delta f_i$;
- среднеквадратическое отклонение $\overline{\sigma}_f = \sqrt{\overline{S}_{100}}$, где $\overline{S}_{100} = \frac{1}{99}\sum_{i=1}^{100} (\Delta f_i - \overline{\Delta f})^2$;
- количество успехов $n_{\text{усп}}$ (попаданий лучшей точки в ε-окрестность точного решения, $\varepsilon = \frac{1}{1000}\max_{i=1,...,n} |b_i - a_i|$).

Результаты, полученные для каждой функции, представлены в табл. 2.4–2.6.

Таблица 2.4. Влияние параметров метода. Параболическая функция

Параметры метода														$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{уст}$
NP	γ	η	α	λ	*P*	*K*	*L*	μ	*h*	c_1	c_2	c_3	ρ				
20	0,6	0,9	0,001	1,5	15	3	4	3	0,1	3	3	3	4	$2\cdot10^{-5}$	0	$2\cdot10^{-5}$	100
10	0,6	0,9	0,001	1,5	15	3	4	3	0,1	3	3	3	4	$5\cdot10^{-5}$	0	$5\cdot10^{-5}$	100
10	0,7	0,9	0,001	1,5	6	3	4	3	0,1	2	2	2	5	$1{,}5\cdot10^{-3}$	0	$1{,}1\cdot10^{-5}$	100
5	0,7	0,9	0,001	1,5	6	3	8	3	0,1	2	2	2	5	$4{,}1\cdot10^{-4}$	$2\cdot10^{-5}$	$4{,}4\cdot10^{-4}$	100
5	0,7	0,9	0,001	1,5	10	3	8	3	0,1	2	2	2	5	$3{,}5\cdot10^{-4}$	0	$4\cdot10^{-4}$	100
5	0,75	0,91	0,001	1,5	10	3	8	2	0,1	3	3	3	5	$9\cdot10^{-5}$	0	$7{,}5\cdot10^{-5}$	100

Таблица 2.5. Влияние параметров метода. Функция Розенброка

Параметры метода														$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{уст}$
NP	γ	η	α	λ	*P*	*K*	*L*	μ	*h*	c_1	c_2	c_3	ρ				
40	0,95	0,89	0,001	1,5	6	6	10	2	0,1	10	10	10	6	$2{,}6\cdot10^{-4}$	0	$4{,}2\cdot10^{-4}$	95
40	0,95	0,89	0,001	1,5	6	6	10	2	0,1	5	5	5	6	$1{,}5\cdot10^{-4}$	0	$2{,}8\cdot10^{-4}$	97
40	0,95	0,89	0,001	1,5	10	6	10	2	0,1	5	5	5	6	$0{,}4\cdot10^{-4}$	0	$0{,}6\cdot10^{-4}$	100
40	0,95	0,89	0,001	1,5	6	15	10	2	0,1	5	5	5	6	$1\cdot10^{-4}$	0	$1{,}5\cdot10^{-4}$	100
40	0,95	0,89	0,001	1,3	6	15	10	2	0,1	5	5	5	6	$1{,}6\cdot10^{-4}$	0	$2{,}6\cdot10^{-4}$	98
40	0,85	0,9	0,001	1,3	6	15	10	2	0,1	5	5	5	6	$1{,}9\cdot10^{-4}$	0	$4\cdot10^{-4}$	96
50	0,95	0,89	0,001	1,5	6	6	10	2	0,1	10	10	10	6	$1{,}9\cdot10^{-4}$	0	$2{,}7\cdot10^{-4}$	97
50	0,95	0,89	0,001	1,5	10	6	10	2	0,1	10	10	10	6	$0{,}5\cdot10^{-4}$	0	$0{,}7\cdot10^{-4}$	100

Таблица 2.6. Влияние параметров метода. Мульти-функция

Параметры метода														$\overline{\Delta f}$	Δf_{best}	$\overline{\sigma}_f$	$n_{уст}$
NP	γ	η	α	λ	*P*	*K*	*L*	μ	*h*	c_1	c_2	c_3	ρ				
20	0,6	0,9	0,001	1,5	15	3	4	3	0,1	3	3	3	4	$6\cdot10^{-5}$	0	$9\cdot10^{-5}$	100
30	0,75	0,9	0,001	1,5	15	10	4	3	0,1	5	5	5	4	$1\cdot10^{-5}$	0	$1\cdot10^{-5}$	100
25	0,75	0,9	0,001	1,5	15	10	4	3	0,1	5	5	5	4	$3\cdot10^{-5}$	0	$7\cdot10^{-5}$	100
20	0,75	0,9	0,001	1,5	15	5	4	3	0,1	3	3	3	4	$2\cdot10^{-5}$	0	$5\cdot10^{-5}$	100
20	0,6	0,9	0,001	1,5	15	3	4	2	0,1	3	3	3	4	$3\cdot10^{-5}$	0	$4\cdot10^{-5}$	100

Анализ работы метода стаи синиц показал, что результат работы алгоритма зависит главным образом от параметров NP, K, L, P.

С большинством стандартных тестовых функций алгоритм успешно справляется даже при малой популяции, а график изменения средней приспособленности указывает на сходимость метода.

Алгоритм имеет ряд недостатков. Подбор большого количества параметров является трудоемкой задачей для каждой конкретной задачи. На функции "Западня" алгоритм часто сходится к локальному экстремуму. Данная проблема устранима за счет выбора большей по размеру популяции и увеличения параметров K и P.

Размер популяции NP рекомендуется полагать в диапазоне от 5 до 100. Данный параметр сильно влияет на точность решения. Для наиболее точного решения следует принимать $30 \le NP \le 100$.

Шаг при передвижении вожака стаи α рекомендуется выбирать из диапазона $0,00001 \le \alpha \le 0,001$.

Параметр сокращения множества поиска γ следует полагать в диапазоне $0,6 \le \gamma \le 0,95$.

Параметр восстановления множества поиска η следует полагать в диапазоне $0,89 \le \eta \le 0,91$.

Параметр распределения Леви λ рекомендуется выбирать $1,2 \le \lambda \le 1,5$.

Параметры $c_1, c_2, c_3 = 10$**, определяющие вектор сноса и матрицу диффузии**, следует полагать $1 \le c_i \le 10$, $i = 1,2,3$.

Максимальное число записей в матрице памяти K следует полагать тем больше, чем большая точность требуется. Рекомендуется выбирать $K \in [3;15]$.

Шаг интегрирования дифференциального уравнения h **и параметр интенсивности скачков** μ. Шаг численного интегрирования h должен быть согласован с интенсивностью пуассоновского потока событий μ: $\mu h \ll 1$. Например, $h = 1$; $\mu = 0,3$.

Максимальное число дискретных шагов L следует принимать в диапазоне $L \in [1;10]$.

Максимальное число проходов P сильно влияет на точность решения. Для наиболее точного решения следует полагать $15 \le P \le 30$.

Параметры NP, K, L, P наиболее существенно влияют на скорость работы алгоритма. При увеличении данных параметров увеличивается время работы.

ГЛАВА 3
ПРИМЕНЕНИЕ МУЛЬТИАГЕНТНЫХ МЕТОДОВ ОПТИМИЗАЦИИ В ЗАДАЧАХ ПОИСКА ОПТИМАЛЬНОГО УПРАВЛЕНИЯ НЕПРЕРЫВНЫМИ ДИНАМИЧЕСКИМИ СИСТЕМАМИ

3.1. КЛАССИФИКАЦИЯ ПОСТАНОВОК ЗАДАЧ ОПТИМАЛЬНОГО УПРАВЛЕНИЯ НЕПРЕРЫВНЫМИ ДИНАМИЧЕСКИМИ СИСТЕМАМИ

Математические модели непрерывных динамических систем управления описываются дифференциальными уравнениями различных типов. Задачи оптимального управления сводятся к проблеме минимизации значений функционалов, определенных на их траекториях. Приведем типовые формулировки постановок задач в порядке возрастания сложности.

Задача 1 (оптимальное управление отдельной траекторией).

Случай А (оптимальное программное управление). Пусть поведение модели объекта управления описывается обыкновенным дифференциальным уравнением

$$\dot{x}(t) = f(t, x(t), u(t)), \tag{3.1}$$

где x – вектор состояния системы, $x = (x_1, \ldots, x_n)^T \in R^n$; u – вектор управления, $u = (u_1, \ldots, u_q)^T \in U \subseteq R^q$, U – некоторое заданное множество допустимых значений управления; t – время, $t \in T = [t_0, t_1]$ – промежуток времени функционирования системы, моменты начала процесса t_0 и окончания процесса t_1 заданы; $f(t, x, u)$ – непрерывная вместе со своими частными производными вектор-функция, $f(t, x, u) = (f_1(t, x, u), \ldots, f_n(t, x, u))^T$, $f(t, x, u) : T \times R^n \times U \to R^n$; R^n – n-мерное евклидово пространство.

Начальное условие

$$x(t_0) = x_0 \tag{3.2}$$

задает начальное состояние системы.

Предполагается, что при управлении используется информация только о времени, т.е. система управления в данном случае является разомкнутой по состоянию и рассматривается так называемое программное управление (рис. 3.1).

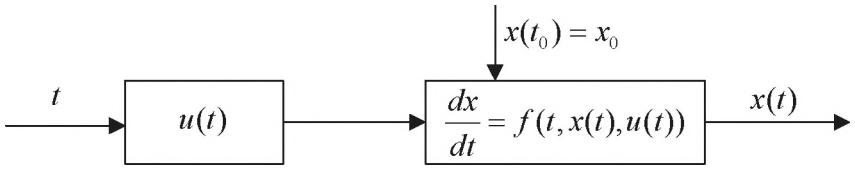


Рис. 3.1

Множество допустимых управлений $\mathcal{U}_0$ образуют кусочно-непрерывные функции $u(\cdot)$ со значениями в множестве U. В точках разрыва значение управления определяется как предел справа.

Определим множество допустимых процессов $\mathcal{D}(t_0, x_0)$ как множество пар $d = (x(\cdot), u(\cdot))$, которые включают траекторию $x(\cdot)$ и управление $u(\cdot)$ (где

$\forall t \in T:\ x(t) \in R^n,\ u(t) \in U$, функции $x(\cdot)$ непрерывны и кусочно-дифференцируемы, а $u(\cdot) \in \mathcal{U}_0$ кусочно-непрерывны), удовлетворяющие уравнению (3.1) с начальным условием (3.2) почти всюду на множестве T.

На множестве $\mathcal{D}(t_0, x_0)$ определим функционал качества управления

$$I(d) = \int_{t_0}^{t_1} f^0(t, x(t), u(t))\, dt + F(x(t_1)), \tag{3.3}$$

где $f^0(t, x, u)$, $F(x)$ – заданные непрерывно дифференцируемые функции.

Требуется найти такую пару $d^* = (x^*(\cdot), u^*(\cdot)) \in \mathcal{D}(t_0, x_0)$, что

$$I(d^*) = \min_{d \in \mathcal{D}(t_0, x_0)} I(d). \tag{3.4}$$

Искомые функции $x^*(\cdot)$ и $u^*(\cdot)$ называются соответственно оптимальной траекторией и оптимальным управлением.

З а м е ч а н и е. Если любое допустимое управление $u(\cdot) \in \mathcal{U}_0$ порождает единственную пару $d \in \mathcal{D}(t_0, x_0)$, то задача (3.4) может быть записана в эквивалентной форме:

$$I(t_0, x_0, u^*(\cdot)) = \min_{u(\cdot) \in \mathcal{U}_0} I(t_0, x_0, u(\cdot)).$$

Случай Б (оптимальное управление с полной обратной связью). Пусть поведение модели объекта управления описывается обыкновенным дифференциальным уравнением (3.1). Начальное условие записывается в форме (3.2): $x(t_0) = x_0 \in R^n$, где начальное состояние x_0 заранее не задано и может быть произвольным. На множестве допустимых процессов $\mathcal{D}(t_0, x_0)$ определен функционал качества управления (3.3).

Предполагается, что при управлении используется информация о времени t и векторе состояния x.

Множество $\mathcal{U}_n$ допустимых управлений с полной обратной связью образуют функции $\boldsymbol{u}(t, x): T \times R^n \to U$, которые для любых начальных состояний порождают соответствующие пары $d = (x(\cdot), u(\cdot)) \in \mathcal{D}(t_0, x_0)$, где программные управления $u(\cdot) \in \mathcal{U}_0$, а $\forall t \in T \quad u(t) = \boldsymbol{u}(t, x(t))$.

Применяемое в каждый момент времени $t \in T$ управление имеет вид управления с полной обратной связью по вектору состояния (рис. 3.2).

Требуется найти такую функцию $\boldsymbol{u}^*(t, x) \in \mathcal{U}_n$, что

$$I(d^*) = \min_{d \in \mathcal{D}(t_0, x_0)} I(d) \qquad \forall x_0 \in R^n, \tag{3.5}$$

где $d^* = (x^*(\cdot), u^*(\cdot) = \boldsymbol{u}^*(\cdot, x(\cdot)))$.

Функция $\boldsymbol{u}^*(t, x) \in \mathcal{U}_n$ называется оптимальным управлением с полной обратной связью. Для любого начального состояния x_0 из множества R^n она порождает соответствующую оптимальную пару, т.е. оптимальную траекторию $x^*(\cdot)$ и оптимальное программное управление $u^*(\cdot)$.

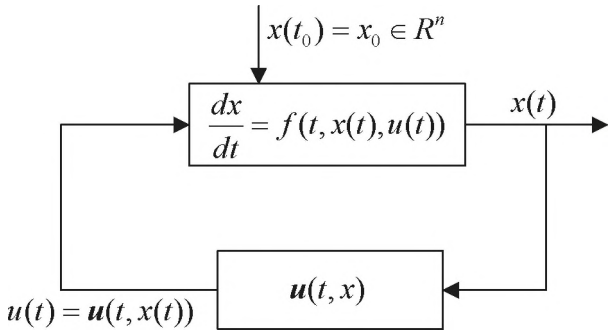


Рис. 3.2

Случай B (оптимальное управление с неполной обратной связью). Пусть поведение модели объекта управления описывается обыкновенным дифференциальным уравнением (3.1). Вектор состояния системы управления представляется в форме $x = (x^1, x^2)^T \in R^n$, $x^1 = (x_1, ..., x_m)^T$, $x^2 = (x_{m+1}, ..., x_n)^T$, $0 \le m \le n$. Предполагается, что о координатах вектора $x^1 \in R^m$ известна текущая информация, а о координатах вектора $x^2 \in R^{n-m}$ она отсутствует. Начальные условия $x(t_0)$ заданы множеством $\Omega \subseteq R^n$, размерность которого равна m, т.е.

$$x(t_0) \in \Omega = \left\{ x \,\middle|\, x^2 = y_0(x^1),\ x^1 \in R^m = B_1 \right\}, \tag{3.6}$$

где $y_{0j}(x^1)$, $j = m+1, ..., n,$ – заданные непрерывно дифференцируемые функции. При $m = 0$ множество Ω является точкой, а при $m = n$ совпадает с множеством R^n.

На множестве допустимых процессов $\mathcal{D}(t_0, x_0)$ определен функционал качества управления (3.3).

Предполагается, что при управлении используется информация только о времени t и о координатах вектора x^1, т.е. управление $u(t)$, применяемое в каждый момент времени $t \in T$, имеет вид управления $u(t) = \boldsymbol{u}(t, x^1(t))$ с неполной обратной связью по вектору состояния (рис. 3.3).

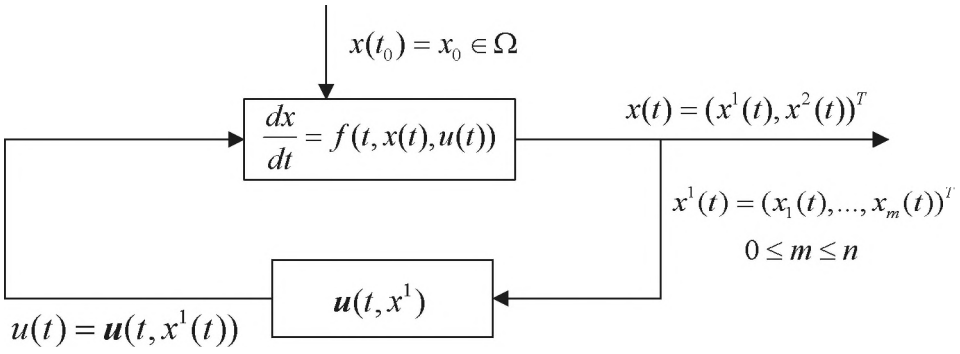


Рис. 3.3

Множество допустимых управлений $\mathcal{U}_m$ с неполной обратной связью образуют функции $\boldsymbol{u}(t,x^1)$: $T\times R^m \to U$ такие, что функции $f_i(t,x,\boldsymbol{u}(t,x^1))$, $i=1,...,n$, непрерывны вместе с частными производными по x, кусочно-непрерывны по t. При этом управление $u(t)=\boldsymbol{u}(t,x^1(t))\in U$ и кусочно-непрерывно по t.

Требуется найти такую функцию $\boldsymbol{u}^*(t,x^1)\in\mathcal{U}_m$, что

$$I(d^*)=\min_{d\in\mathcal{D}(t_0,x_0)} I(d) \qquad \forall x_0\in\Omega, \tag{3.7}$$

где $d^*=(x^*(\cdot),u^*(\cdot)=\boldsymbol{u}^*(\cdot,x^{1*}(\cdot))$.

Функция $\boldsymbol{u}^*(t,x^1)\in\mathcal{U}_m$ называется оптимальным управлением с неполной обратной связью на множестве Ω. Для каждого начального условия из множества Ω она порождает оптимальную пару, т.е. оптимальную траекторию $x^*(\cdot)$ и оптимальное программное управление $u^*(\cdot)$.

Подчеркнем, что число используемых в управлении координат вектора состояния совпадает с размерностью множества начальных условий Ω. При $m=0$ множество Ω является точкой x_0, для которой ищется оптимальное программное управление $u^*(\cdot)$, а при $m=n$ множество Ω совпадает с n-мерным пространством состояний и ищется оптимальное управление $\boldsymbol{u}^*(t,x)$ с полной обратной связью по вектору состояния.

Задача 2 (оптимальное управление пучками траекторий).

Пусть поведение модели объекта управления описывается обыкновенным дифференциальным уравнением (3.1). Вектор состояния системы представляется в форме $x=(x^1,x^2)^T\in R^n$, $x^1=(x_1,...,x_m)^T$, $x^2=(x_{m+1},...,x_n)^T$, $0\le m\le n$. Предполагается, что о координатах вектора $x^1\in R^m$ известна текущая информация, а о координатах вектора $x^2\in R^{n-m}$ она отсутствует.

Начальные условия заданы компактным множеством Ω положительной меры с кусочно-гладкой границей:

$$x(t_0)=x_0\in\Omega\subset R^n, \tag{3.8}$$

где множество Ω характеризует неопределенность задания начальных условий.

Предполагается, что при управлении используется информация только о времени t и о координатах вектора x^1, т.е. управление $u(t)$, применяемое в каждый момент времени $t\in T$, имеет вид управления $u(t)=\boldsymbol{u}(t,x^1(t))$ с неполной обратной связью по вектору состояния (см. рис. 3.3).

Множество допустимых управлений $\mathcal{U}_m$ образуют такие функции $\boldsymbol{u}(t,x^1)$, что $\forall t\in T$ управление $u(t)=\boldsymbol{u}(t,x^1(t))\in U$ кусочно-непрерывно, а функция $f(t,x,\boldsymbol{u}(t,x^1))$ такова, что решение уравнения (3.1) с начальным условием (3.8) существует и единственно.

Управление, применяемое в каждый момент времени t, имеет вид управления с неполной обратной связью: $u(t)=\boldsymbol{u}(t,x^1(t))$ (см. рис. 3.3). Если $m=0$, система управления будет разомкнутой по состоянию (см. рис. 3.1), а соответствующее управление $u(t)$ – программным, а если $m=n$, то система управления будет замкнутой с полной обратной связью, определяемой управлением $\boldsymbol{u}(t,x)$ (см. рис. 3.2).

Множество допустимых процессов $\mathcal{D}(t_0, x_0)$ – множество пар $d = (x(\cdot), u(\cdot))$, включающих траекторию $x(\cdot)$ и кусочно-непрерывное допустимое управление $u(\cdot)$, где $\forall t \in T \quad u(t) \in U$, удовлетворяющих уравнению состояния (3.1) и начальному условию (3.2).

На множестве $\mathcal{D}(t_0, x_0)$ определен функционал качества управления отдельной траекторией:

$$I(x_0, d) = \int_{t_0}^{t_1} f^0(t, x(t), u(t))\, dt + F(x(t_1)), \tag{3.9}$$

где $f^0(t, x, u)$, $F(x)$ – заданные непрерывные функции.

Каждому допустимому управлению $\boldsymbol{u}(t, x^1) \in \mathcal{U}_m$ и множеству Ω поставим в соответствие пучок (ансамбль) траекторий уравнения (3.1) [24]:

$$X(t, \boldsymbol{u}(t, x^1)) = \bigcup \{x(t, \boldsymbol{u}(t, x^1(t)), x(t_0)) \mid x(t_0) \in \Omega\},\ t \in T, \tag{3.10}$$

т.е. объединение решений уравнения (3.1) по всем возможным начальным состояниям (3.8). Пучок траекторий порождается множеством Ω и управлением $\boldsymbol{u}(t, x^1) \in \mathcal{U}_m$ (на рис. 3.4 для случая $n = 2$ изображены сечения $\Omega_{t,u}$ пучка траекторий).

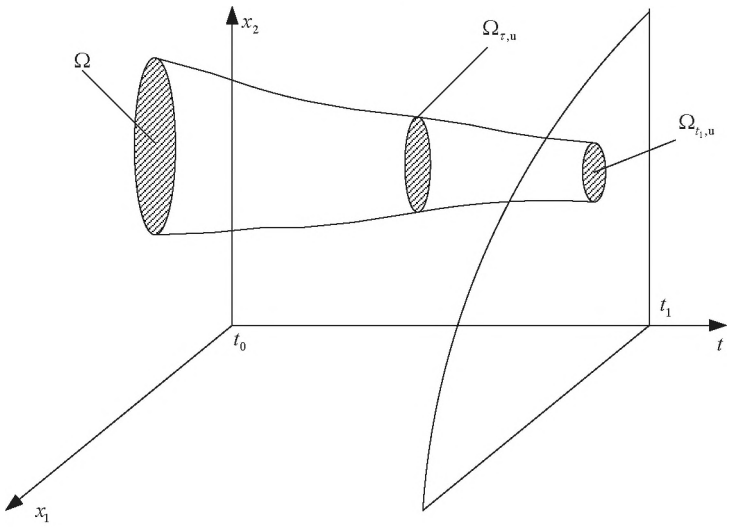


Рис. 3.4

Качество управления пучком траекторий предлагается оценивать величиной функционала

$$J[\boldsymbol{u}(t, x^1)] = \int_{\Omega} I(x_0, d) dx_0 \,/\, \text{mes}\ \Omega, \tag{3.11}$$

или

$$J[\boldsymbol{u}(t, x^1)] = \max_{x_0 \in \Omega} I(x_0, d), \tag{3.12}$$

где mes Ω – мера множества Ω.

Требуется найти управление $\boldsymbol{u}^*(t,x^1)\in \mathcal{U}_m$, минимизирующее величину функционала (3.11) или (3.12):

$$J[\boldsymbol{u}^*(t,x^1)]=\min_{\boldsymbol{u}(t,x^1)\in \mathcal{U}_m} J[\boldsymbol{u}(t,x^1)]. \tag{3.13}$$

Искомое управление называется оптимальным в среднем, когда минимизируется значение функционала (3.11), т.е. среднее значение функционала (3.9) на множестве начальных состояний Ω, или гарантирующим (минимаксным), когда минимизируется значение функционала (3.12).

Задача 3 (оптимальное управление стохастическими системами).

Поведение модели объекта управления описывается стохастическим дифференциальным уравнением Ито:

$$dX=f(t,X(t),u(t))\,dt+\sigma(t,X(t),u(t))\,dW,\quad X(t_0)=X_0, \tag{3.14}$$

где X – вектор состояния системы, $X=(X^1,X^2)^T\in R^n$, $X^1=(X_1,...,X_m)^T$, $X^2=(X_{m+1},...,X_n)^T$, $0\le m\le n$; u – вектор управления, $u\in U\subseteq R^q$, U – некоторое заданное множество, $t\in T=[t_0,t_1]$, T – промежуток времени функционирования системы, моменты времени t_0 и t_1 заданы; $W(t)$ – k-мерный стандартный винеровский случайный процесс, не зависящий от X_0 (второй член в уравнении (3.14) характеризует случайные внешние воздействия на объект); $f(t,x,u):T\times R^n\times U\to R^n$, $\sigma(t,x,u)$ – матричная функция размеров $(n\times k)$. Предполагается, что о координатах вектора $X^1\in R^m$ текущая информация известна, а о координатах вектора $X^2\in R^{n-m}$ отсутствует.

Начальное состояние X_0 является случайным и определяется заданной плотностью вероятности.

Предполагается, что при управлении используется информация только о времени t и о координатах вектора X^1, т.е. управление, применяемое в каждый момент времени $t\in T$, имеет вид управления с неполной обратной связью $u(t)=\boldsymbol{u}(t,X^1(t))$ (рис 3.5).

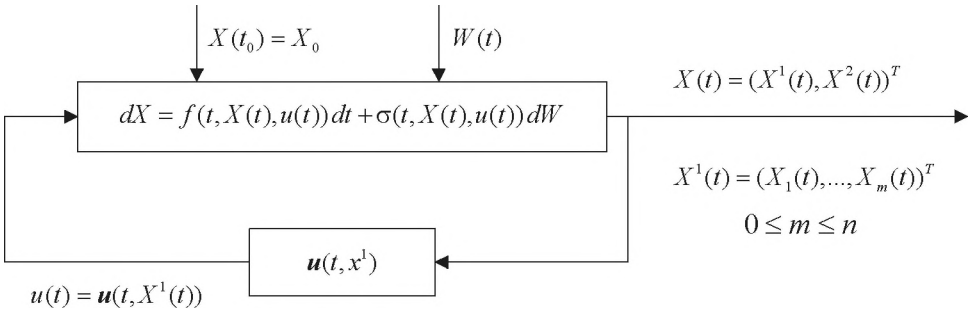
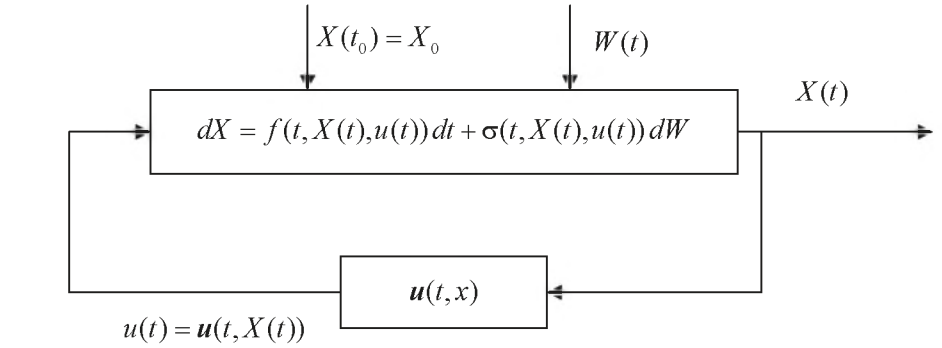


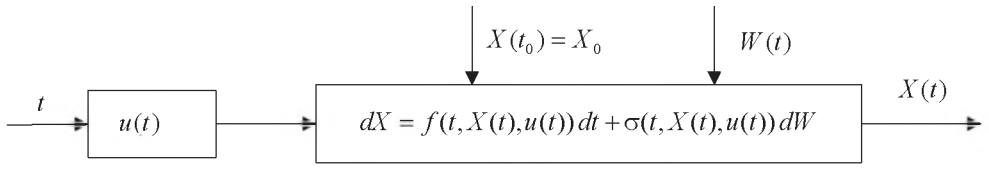
Рис. 3.5

Число m, $0\le m\le n$, определяется условиями информированности. При $m=n$ имеется информация обо всех координатах вектора X, т.е. система (см. рис. 3.5) будет системой с полной обратной связью (рис. 3.6, а), а при $m=0$ – системой, разомкнутой

по состоянию (рис. 3.6, б). В последнем случае рассматривается так называемое программное управление $u(t)$.



а



б

Рис. 3.6

Множество допустимых управлений с неполной обратной связью $\mathcal{U}_m$ образуют функции $\boldsymbol{u}(t, x^1) : T \times R^m \to U$ такие, что для всех $i = 1,...,n; j = 1,...,k$ функции $f_i^{u(\cdot)}(t,x) = f_i(t, x, \boldsymbol{u}(t, x^1))$, $\sigma_{ij}{}^{u(\cdot)}(t,x) = \sigma_{ij}(t, x, \boldsymbol{u}(t, x^1))$ удовлетворяют условиям, при которых решение уравнения (3.14) существует, единственно и является непрерывным марковским процессом [86, 87].

Определим функционал качества управления

$$J[\boldsymbol{u}(t, x^1)] = M\left\{ \int_{t_0}^{t_1} f^0(t, X(t), \boldsymbol{u}(t, X^1(t)))dt + F(X(t_1)) \right\}, \tag{3.15}$$

где непрерывные функции $f^0(t, x, u) : T \times R^n \times U \to R$, $F(x) : R^n \to R$ удовлетворяют условию полиномиального роста [86], а M – знак математического ожидания.

Требуется найти управление $\boldsymbol{u}^*(t, x^1) \in \mathcal{U}_m$, минимизирующее величину функционала (3.15):

$$J[\boldsymbol{u}^*(t, x^1)] = \min_{\boldsymbol{u}(t, x^1) \in \mathcal{U}_m} J[\boldsymbol{u}(t, x^1)]. \tag{3.16}$$

Искомое управление называется оптимальным в среднем, поскольку минимизируется среднее значение функционала, определенного на траекториях динамической системы. При $m = n$, т.е. если имеется информация обо всех координатах вектора X, формула (3.16) соответствует формулировке задачи поиска оптимального управления с полной обратной связью, а при $m = 0$ – формулировке поиска оптимального программного управления.

3.2. СПОСОБЫ ПАРАМЕТРИЗАЦИИ ЗАКОНОВ УПРАВЛЕНИЯ

Рассмотренные в параграфе 3.1 постановки задач оптимального управления связаны с нахождением закона управления, зависящего от времени и от заданного набора координат вектора состояния, доступных измерению.

Для численного решения задачи требуется задать структуру искомого закона управления, зависящую от неизвестных параметров. Таким образом, реализуется переход от поставленных вариационных задач к проблеме конечномерной оптимизации, т.е. проблеме поиска наилучших значений параметров, задающих структуру управления. Для того чтобы задать структуру управления, рекомендуется предварительно воспользоваться необходимыми или достаточными условиями оптимальности и связанными с ними соотношениями.

При решении задачи 1 используются: принцип максимума [1, 11, 22, 26, 35, 40, 56, 60, 87] для поиска программного управления, уравнение Беллмана [1, 11, 22, 26, 35, 40, 86, 87] для поиска управления с полной обратной связью, соотношения для нахождения управления с неполной обратной связью [35, 40, 52].

При решении задачи 2 используются соотношения для нахождения управления с неполной обратной связью и следующие из них частные случаи, если информация о векторе состояния отсутствует или, наоборот, доступна полностью [30, 31, 40, 52].

При решении задачи 3 используются: стохастический принцип максимума [103] для поиска программного управления, уравнение Беллмана для стохастических систем [86, 87, 116] для поиска управления с полной обратной связью, соотношения для нахождения управления с неполной обратной связью [35, 40, 52].

Замечания.

1. В задачах с параллелепипедными ограничениями на управление структура оптимального управления, как следует из необходимых или достаточных условий оптимальности, часто является релейной. Поэтому желательно задавать ее так, чтобы ограничения на управление удовлетворялись автоматически. Для задач поиска программного управления фактически требуется найти наилучшее число переключений управления и значения моментов переключения.

2. Промежуток функционирования системы будет обозначаться либо $[t_0, t_1]$, либо $[t_0, t_f]$ в зависимости от удобства обозначений других переменных задачи.

Для численного решения проблем оптимального управления можно использовать различные процедуры улучшения начального приближения в пространстве управлений или в пространстве траекторий [9, 10, 22, 26, 60, 96, 120]; различные подходы, связанные с дискретизацией задачи [9, 10, 67, 97, 116, 148]; конечно-разностные процедуры решения уравнений Беллмана [11, 22, 26, 86, 87, 116]; аппроксимации законов управления совместно с применением методов оптимизации нулевого, первого или второго порядков [11, 14, 36, 53, 59, 65, 95, 97, 120, 121].

В общем случае закон управления $\boldsymbol{u}(t, x^1)$ предлагается искать в параметрическом виде, определяемом числом коэффициентов в разложении управления по выбираемой системе базисных функций и их значениями.

Могут применяться следующие способы параметризации закона управления.

1. *Применение ортонормированных систем базисных функций*, получивших широкое распространение в спектральном методе [50, 51]. Предлагается искать закон управления в виде функции насыщения sat, гарантирующей выполнение параллелепипедных ограничений на управление вида $u_j(t) \in [a_j(t), b_j(t)], j = 1, ..., q; t \in T$:

$$\boldsymbol{u}_j(t,x^1)=\operatorname{sat}\left\{g_j\left(t,x_1,...,x_m\right)\right\},\ j=1,...,q\,. \tag{3.17}$$

Здесь

$$\operatorname{sat}v_j(t)=\begin{cases}v_j(t), & a_j(t)\le v_j(t)\le b_j(t),\\ a_j(t), & v_j(t)<a_j(t),\\ b_j(t), & v_j(t)>b_j(t),\end{cases} \tag{3.18}$$

$v_j(t)=g_j\left(t,x_1(t),...,x_m(t)\right)$, а аргументы $g_j\left(t,x_1,...,x_m\right)$ функции насыщения искать в виде линейной комбинации базисных функций:

$$g_j\left(t,x_1,...,x_m\right)=\sum_{i_0=0}^{L_0^j-1}\sum_{i_1=0}^{L_1^j-1}\cdots\sum_{i_m=0}^{L_m^j-1}c^j_{i_0i_1\ldots i_m}\,q\left(i_0,t\right)p_1\left(i_1,x_1\right)\cdots p_m\left(i_m,x_m\right), \tag{3.19}$$

где $c^j_{i_0i_1\ldots i_m}$ – неизвестные коэффициенты; $L_0^j,L_1^j,...,L_m^j$ – масштабы усечения по времени и координатам вектора состояния, используемым в управлении.

В качестве базисных функций $q\left(i_0,t\right)$, $p_k\left(i_k,x_k\right)$, $k=1,...,m$, можно, например, взять ортонормированную на отрезке $\left[0;t_f\right]$ систему нестационарных косинусоид:

$$q(i_0,t)=\begin{cases}\sqrt{\dfrac{1}{t_f}}, & i_0=0,\\ \sqrt{\dfrac{2}{t_f}}\cos\dfrac{i_0\pi t}{t_f}, & i_0=1,...,L_0-1,\end{cases}$$

$$p_k(i_k,x_k)=\begin{cases}\sqrt{\dfrac{1}{\overline{x}_k-\underline{x}_k}}, & i_k=0,\\ \sqrt{\dfrac{2}{\overline{x}_k-\underline{x}_k}}\cos\dfrac{i_k\pi(x_k-\underline{x}_k)}{\overline{x}_k-\underline{x}_k}, & i_k=1,...,L_k-1,\end{cases} \tag{3.20}$$

а также многочленов Лежандра, функции Уолша и т.д. [50, 51].

Здесь предполагается, что $t_0=0$ и известна оценка множества возможных состояний, которая представляется прямым произведением $[\underline{x}_1,\overline{x}_1]\times\cdots\times[\underline{x}_n,\overline{x}_n]$, где $\underline{x}_i,\overline{x}_i$ – нижняя и верхняя границы по каждой координате вектора состояния соответственно, определяемые физическим смыслом решаемой задачи.

Неизвестные параметры j-й координаты закона управления: матрица-столбец масштабов усечения $L^j=(L_0^j,L_1^j,...,L_m^j)^T$ и наилучших значений коэффициентов разложения $c^j_{i_0i_1\ldots i_m}$, удобно представить в виде блочной гиперстолбцовой матрицы:

$$C^j=\left[\begin{bmatrix}c^j_{00\ldots0}\\ c^j_{00\ldots1}\\ \vdots\\ c^j_{00\ldots(L_m^j-1)}\end{bmatrix}\begin{bmatrix}c^j_{10\ldots0}\\ c^j_{10\ldots1}\\ \vdots\\ c^j_{10\ldots(L_m^j-1)}\end{bmatrix}\cdots\begin{bmatrix}c^j_{(L_0^j-1)0\ldots0}\\ c^j_{(L_0^j-1)0\ldots1}\\ \vdots\\ c^j_{(L_0^j-1)0\ldots(L_m^j-1)}\end{bmatrix}\cdots\begin{bmatrix}c^j_{(L_0^j-1)(L_1^j-1)\ldots0}\\ c^j_{(L_0^j-1)(L_1^j-1)\ldots1}\\ \vdots\\ c^j_{(L_0^j-1)(L_1^j-1)\ldots(L_m^j-1)}\end{bmatrix}\right]^T,\ j=1,...,q.$$

Тогда векторы неизвестных, подлежащих поиску, можно записать в форме объединенных матриц-столбцов

$$L = (L^1,...,L^q)^T, \quad C = (C^1,....,C^q)^T \tag{3.21}$$

с ограничениями на компоненты:

$$0 \le L_i^j \le L_{\max}, \ L_i^j - \text{целые}; \ j = 1,...,q, \ i = 0,...,m, \tag{3.22}$$

$$c_{\min} \le c^j_{i_0,i_1,...,i_m} \le c_{\max}, \ i_0 = 0,1,...,L_0^j - 1;...;i_m = 0,1,...,L_m^j - 1;$$

где значения $L_{\max}$, $c_{\min}, c_{\max}$ задаются исходя из возможных требований к точности решения и лимиту вычислительных затрат.

2. *Применение финитных базисных систем функций.* При $m = 0$, т.е. в задачах поиска программного управления, закон управления ищется в виде функции насыщения sat, гарантирующей выполнение ограничений на управление:

$$u_j(t) = \text{sat}\{g_j(t)\}, \ j = 1,...,q, \tag{3.23}$$

$$g_j(t) = \sum_{i=0}^{L^j-1} c_i^j S(i,t). \tag{3.24}$$

В качестве базисной системы функций $S(i,t)$ можно использовать сплайны: $S(i,t) = Sp(t/h - i)$, $h = 1/(L^j - 1)$, $i = 0,...,L^j - 1$, – финитные функции, порожденные сплайнами, на отрезке $[0;1]$:

$$Sp(t) = \begin{cases} 2^{p-1}(1+t)^p, t \in [-1;-\frac{1}{2}], \\ 1 - 2^{p-1}|t|^p, t \in [-\frac{1}{2};\frac{1}{2}], \\ 2^{p-1}(1-t)^p, \ t \in [\frac{1}{2};1], \\ 0, \qquad t \notin [-1;1], \end{cases} \tag{3.25}$$

где $Sp(t)$ при $p = 0$ задает кусочно-постоянный сплайн, при $p = 1$ – кусочно-линейный (крышки), при $p = 2$ – квадратичный, при $p = 3$ – кубический.

Для преобразования отрезка $[t_0, t_f]$ к отрезку $[0;1]$ можно воспользоваться соотношением $t = t_0 + (t_f - t_0)\tau$, $\tau \in [0;1]$. Решение ищется в виде расширенного вектора

$$(L^1,...,L^q | c_0^1,...,c_{L^1-1}^1,...,c_0^q,...,c_{L^q-1}^q)^T \tag{3.26}$$

с ограничениями на координаты:

$$0 \le L^j \le L_{\max}, \ L^j - \text{целые}; \ j = 1,...,q, \tag{3.27}$$

$$c_{\min} \le c_i^j \le c_{\max}.$$

3. *Применение радиально-базисных функций* [73, 146]. Предлагается использовать:

а) функцию Гаусса $\varphi(r) = e^{-(\mu r)^2}$,

б) мульти-квадратичную функцию $\varphi(r)=\sqrt{1+(\mu r)^2}$,

в) обратную квадратичную функцию $\varphi(r)=1\Big/\left(1+(\mu r)^2\right)$,

г) обратную мульти-квадратичную функцию $\varphi(r)=1\Big/\sqrt{1+(\mu r)^2}$.

Здесь μ – масштабный коэффициент, x^i – базовая точка, $r=\left\|x-x^i\right\|$.

Закон управления представляется в форме (3.17) – (3.19), где

$$g_j\left(t,x_1,...,x_m\right)=\sum_{i_0=0}^{L_0^j-1}\sum_{i_1=0}^{L_1^j-1}\cdots\sum_{i_m=0}^{L_m^j-1}c^j_{i_0i_1\ldots i_m}\,\varphi(\left|t-t_{i_0}\right|)\varphi(\left|x_1-x_{1i_1}\right|)\cdots\varphi(\left|x_m-x_{mi_m}\right|, \tag{3.28}$$

$t_{i_0}, i_0=0,...,L_0^j-1$ – точки разбиения промежутка времени функционирования системы $[t_0,t_f]$, а $x_{ki_k}, i_k=0,...,L_k^j-1;\ k=1,...,m$ – базовые точки разбиения отрезка $[\underline{x}_k,\overline{x}_k]$. Вектор неизвестных можно записать в виде (3.21), (3.26).

4. *Применение разложений по многочленам Чебышёва*, используемых в различных вариантах псевдоспектрального метода [70, 90, 91].

Вариант А. Можно использовать систему многочленов Чебышева, ортогональных с весом $\rho(x)=1\Big/\sqrt{1-x^2}$ на отрезке $[-1;1]$:

$$T_0(x)=1,\ T_1(x)=x,\ T_{m+1}(x)=2xT_m(x)-T_{m-1}(x),\ m\ge 1. \tag{3.29}$$

При этом

$$\int_{-1}^{1}\frac{T_m(x)T_n(x)}{\sqrt{1-x^2}}dx=\begin{cases}0, & m\ne n,\\ \pi, & m=n=0,\\ \dfrac{\pi}{2}, & m=n\ne 0.\end{cases}$$

Тогда справедливо представление (3.17)–(3.19), где аргументы $g_j\left(t,x_1,...,x_m\right)$ функции насыщения можно искать в виде линейной комбинации базисных функций:

$$g_j\left(t,x_1,...,x_m\right)=\sum_{i_0=0}^{L_0^j}\sum_{i_1=0}^{L_1^j}\cdots\sum_{i_m=0}^{L_m^j}c^j_{i_0i_1\ldots i_m}\,T_{i_0}\left(t\right)T_{i_1}\left(x_1\right)\cdots T_{i_m}\left(x_m\right). \tag{3.30}$$

Здесь $c^j_{i_0i_1\ldots i_m}$ – неизвестные коэффициенты; $L_0^j,L_1^j,...,L_m^j$ – масштабы усечения по времени и координатам вектора состояния, используемым в управлении. Решение ищется как расширенная матрица-столбец $C_N=(C_N^1,....,C_N^q)^T$ вида (3.21) с ограничениями (3.22).

Для преобразования отрезков $[t_0,t_f],[\underline{x}_1,\overline{x}_1],...,[\underline{x}_n,\overline{x}_n]$ к стандартному отрезку $[-1;1]$ можно применить линейные преобразования:

$$t=\frac{t_f-t_0}{2}\tau+\frac{t_f+t_0}{2},\ dt=\frac{t_f-t_0}{2}d\tau,\ \tau\in[-1;1],$$

$$x_i=\frac{\underline{x}_i+\overline{x}_i}{2}+\frac{\overline{x}_i-\underline{x}_i}{2}\tilde{x}_i;\ \tilde{x}_i\in[-1;1],\ i=1,...,n.$$

В результате уравнение (3.1) и функционал (3.3) перепишем в следующей форме:

$$\dot{x}(\tau) = \frac{t_f - t_0}{2} f(\tau, x(\tau), u(\tau)), \tag{3.31}$$

$$I(x_0, d) = \frac{t_f - t_0}{2} \int_{-1}^{1} f^0(\tau, x(\tau), u(\tau)) d\tau + F(x(1)).$$

Вариант Б. Для случая $m = 0$ можно использовать многочлены Чебышева, а в качестве узлов интерполяции выбрать положения экстремумов многочлена Чебышева степени N^j и крайние точки отрезка интерполяции: $t_k = \cos(\pi k / N^j), k = 0,..., N^j$.

На отрезке $[-1;1]$ они располагаются так: $t_0 = 1, t_1, ..., t_{N^j-1}, t_{N^j} = -1$ и называются точками CGL (Chebyshev–Gauss–Lobatto). Поскольку многочлены Чебышева можно записать в форме $T_m(t) = \cos(m \arccos t)$, $m = 0,..., N^j$, то их значения в узлах найдем в виде

$$T_m(t_k) = \cos(\pi k m / N^j),\ k = 0,..., N^j;\ j = 1,..., q,$$

$$u_j^{N^j}(t) = \sum_{i=0}^{N^j} u_i^j \varphi_i(t),\ j = 1,..., q, \tag{3.32}$$

где u_i^j – неизвестные величины. В качестве базисной системы $\{\varphi_i(t)\}_{i=0}^{N^j}$ предлагается использовать [70, 90, 91]

$$\varphi_i(t) = \frac{(-1)^{i+1}(1-t^2)T'_{N^j}(t)}{(N^j)^2 c_i (t - t_i)}, i = 0,..., N^j; \quad c_i = \begin{cases} 2, i = 0, i = N^j, \\ 1, i = 1,..., N^j - 1. \end{cases}$$

При этом в узлах выполняются соотношения

$$\varphi_i(t_k) = \delta_{ik},\ u_j^{N^j}(t_k) = u_k^j,$$

где δ_{ik} – символ Кронекера.

На коэффициенты разложения накладываются ограничения, следующие из постановки задачи:

$$a_j(t_k) \le u_k^j \le b_j(t_k), k = 0,..., N^j; j = 1,..., q. \tag{3.33}$$

В задаче требуется найти координаты расширенного вектора

$$(N^1,..., N^q \,|\, u_0^1,..., u_N^1,..., u_0^q,..., u_N^q)^T, \tag{3.34}$$

содержащего блок целочисленных переменных $N^1,..., N^q$ и блок действительных переменных, на которые накладываются ограничения вида (3.33).

Таким образом, при всех описанных способах параметризации закона управления предлагается искать значения координат блочных расширенных векторов (3.21), (3.26), (3.34), содержащих блок целочисленных переменных и блок действительных переменных, на которые наложены интервальные ограничения.

Для оценки эффективности предложенного подхода создан программный комплекс [17], с помощью которого решены задачи поиска оптимального управления. Вычисления проводились на процессоре Intel Core i7-1165G7 с частотой 2,8 GHz и оперативной памятью DDR3 объемом 16 GB. В качестве среды разработки использовалась Microsoft Visual Studio 2012 и язык программирования C#.

3.3. ПОИСК ОПТИМАЛЬНОГО ПРОГРАММНОГО УПРАВЛЕНИЯ ОДНИМ КЛАССОМ НЕЛИНЕЙНЫХ СИСТЕМ, ЛИНЕЙНЫХ ПО ОГРАНИЧЕННОМУ УПРАВЛЕНИЮ

3.3.1. Постановка задачи

Рассматривается класс нелинейных непрерывных детерминированных динамических систем, линейных по ограниченному управлению:

$$\dot{x}(t) = A(t, x(t)) + B(t)u(t), \tag{3.35}$$

где x – вектор состояния системы, $x = (x_1, \ldots, x_n)^T \in R^n$; u – управление, $u \in U \subseteq R$, U – заданное множество допустимых значений управления, определяемое отрезком $[a, b]$; $t \in T = [t_0, t_1]$ – промежуток времени функционирования системы; моменты начала процесса t_0 и окончания t_1 заданы; $A(t, x), B(t)$ – непрерывные вектор-функции размеров $(n \times 1)$; R^n – n-мерное евклидово пространство.

Начальное условие

$$x(t_0) = x_0 \tag{3.36}$$

задает начальное состояние системы.

Определим множество допустимых процессов $\mathcal{D}(t_0, x_0)$ как множество пар $d = (x(\cdot), u(\cdot))$, которые включают траекторию $x(\cdot)$ и управление $u(\cdot)$ (где $\forall t \in T: x(t) \in R^n,\ u(t) \in U$, функции $x(\cdot)$ непрерывны и кусочно-дифференцируемы, а $u(\cdot)$ кусочно-непрерывны), удовлетворяющие уравнению (3.35) с заданным начальным условием (3.36).

На множестве $\mathcal{D}(t_0, x_0)$ определим функционал качества управления

$$I(d) = F(x(t_1)), \tag{3.37}$$

где $F(x)$ – заданная непрерывная функция.

Требуется найти такую пару $d^* = (x^*(\cdot), u^*(\cdot)) \in \mathcal{D}(t_0, x_0)$, что

$$I(d^*) = \min_{d \in \mathcal{D}(t_0, x_0)} I(d). \tag{3.38}$$

3.3.2. Алгоритм решения задачи

Применение необходимых условий оптимальности в форме принципа максимума для математической модели (3.35) с ограниченным управлением параллелепипедного вида, как правило, определяет релейную структуру оптимального управления. Поиск приближенного решения сводится к нахождению решения соответствующей двухточечной краевой задачи [1, 11, 22, 40, 56]. Во многих прикладных задачах при этом возникают проблемы, связанные с обеспечением сходимости вычислительных процедур.

Предлагается использовать знание структуры искомого управления, а поэтому искать программное управление в виде некоторой ступенчатой функции с неопределенными параметрами, которая должна гарантировать выполнение заданных ограничений на управление. Для нахождения параметров применяются мультиагентные алгоритмы оптимизации, изложенные в разд. 1.3 и 1.4.

Для решения задачи можно использовать два способа параметризации закона управления.

Первый способ. Сведем задачу к поиску величины параметра p и последовательности моментов переключения $\{t_{П_0},...,t_{П_p}\}$ управления:

$$u(t)=\begin{cases} a\chi(t-t_0)+(b-a)\sum\limits_{k=0}^{p}(-1)^k\chi(t-t_{П_k}),\ t_{П_0}=t_0, \\ a\chi(t-t_0)+(b-a)\sum\limits_{k=0}^{p}(-1)^{k+1}\chi(t-t_{П_k}),\ t_{П_0}=t_1, \end{cases} \tag{3.39}$$

где $\chi(t)=\operatorname{sign}(t)=\begin{cases}0, t<0,\\ 1, t\ge 0,\end{cases}$ – единичная ступенчатая функция, $t_{П_k}$ – точки переключения, $t_{П_k}\in[t_0,t_1], k=0,...,p$.

Таким образом, в процессе минимизации функционала (3.37) ищется наилучшее решение вида

$$(p\,|\,t_{П_0},t_{П_1},...,t_{П_p})^T. \tag{3.40}$$

АЛГОРИТМ ПОИСКА ОПТИМАЛЬНОГО ПРОГРАММНОГО УПРАВЛЕНИЯ НА ОСНОВЕ ПЕРЕКЛЮЧЕНИЙ

Шаг 1. *Инициализация.* Выбрать метод из семейства мультиагентных алгоритмов и задать его параметры. Задать $p=0$ в управлении (3.39), найти значения функционала при $t_{П_0}=t_0$ и $t_{П_0}=t_1$. Лучшее из них принять за I_0^*.

Положить $p=1$.

Шаг 2. *Нахождение наилучших значений точек переключения* путем применения выбранного мультиагентного алгоритма оптимизации.

Сгенерировать начальную популяцию, состоящую из NP агентов с положениями $(t_{П_0},t_{П_1},...,t_{П_p})^T_{(j)}$, $j=1,...,NP$, где $t_{П_0}\in\{t_0,t_1\}$, $t_{П_k}\in(t_0,t_1)$, $t_{П_{k+1}}>t_{П_k}$ при $k>1$.

По сгенерированным матрицам-столбцам $(t_{П_0},t_{П_1},...,t_{П_p})^T_{(j)}$, $j=1,...,NP$, сформировать управления

$$u^{(j)}(t)=\begin{cases} a\chi(t-t_0)+(b-a)\sum\limits_{k=0}^{p}(-1)^k\chi(t-t_{П_k}),\ t_{П_0}=t_0, \\ a\chi(t-t_0)+(b-a)\sum\limits_{k=0}^{p}(-1)^{k+1}\chi(t-t_{П_k}),\ t_{П_0}=t_1. \end{cases}$$

Проинтегрировать NP систем дифференциальных уравнений (3.35) при начальном условии (3.36) с управлениями $u^{(1)}(\cdot),...,u^{(NP)}(\cdot)$, например, методом Рунге–Кутты 4-го порядка. Для каждого агента получить соответствующие траектории $x^{(1)}(\cdot)=(x_1^{(1)}(\cdot),\dots,x_n^{(1)}(\cdot)),\dots,x^{(NP)}(\cdot)=(x_1^{(NP)}(\cdot),\dots,x_n^{(NP)}(\cdot))$ и вычислить значения функционала $I^{(1)},\dots,I^{(NP)}$.

Если достигнуто максимальное число итераций мультиагентного алгоритма оптимизации, то из последней популяции следует выбрать наилучшее решение (поло-

жение наилучшего агента), т.е. наилучший вектор $(t_{П_0}, t_{П_1}, ..., t_{П_p})^T_*$ моментов переключения и соответствующее значение функционала I^*_p.

Шаг 3. *Проверка условий завершения поиска.* Если $I^*_p < I^*_{p-1}$ (условие проверяется при $p \geq 1$), то положить $p = p+1$ и перейти к шагу 2. Если условие $I^*_p \geq I^*_{p-1}$ выполняется на двух последовательных итерациях, то процедура поиска параметров оптимального программного управления завершается (иначе следует положить $p = p+1$ и перейти к шагу 2).

Второй способ. Закон управления предлагается искать в виде, гарантирующем выполнение ограничений на управление:

$$u(t) = \text{sign}\{a, b, g(t)\}, \tag{3.41}$$

где $\forall t \in T \quad g(t) = \sum_{i=0}^{L-1} c_i S(i,t)$, $\text{sign}(a,b,g(t)) = \begin{cases} a, & g(t) < 0, \\ b, & g(t) \geq 0, \end{cases}$ а $\{c_0, c_1, ..., c_{L-1}\}$ – неизвестные коэффициенты разложения по базисной системе функций $\{S(i,t)\}_{i=0}^{L-1}$; L – масштаб усечения разложения.

В качестве базисной функций $\{S(i,t)\}_{i=0}^{L-1}$ в случае $t_0 = 0$ можно взять ортонормированную на отрезке $[0, t_1]$ систему нестационарных косинусоид:

$$S(i,t) = \begin{cases} \sqrt{\dfrac{1}{t_1}}, & i = 0, \\ \sqrt{\dfrac{2}{t_1}} \cos\left(\dfrac{i\pi t}{t_1}\right), & i = 1, 2, \ldots, L-1, \end{cases} \tag{3.42}$$

где t_1 – момент окончания функционирования системы.

Таким образом, в процессе минимизации функционала (3.37) ищется наилучшее решение вида

$$(L \,|\, c_0, c_1, ..., c_{L-1})^T. \tag{3.43}$$

На координаты вектора (3.43) накладываются интервальные ограничения вида

$$c_i \in [c_{i\min}, c_{i\max}], \; i = 0, ..., L-1. \tag{3.44}$$

АЛГОРИТМ ПОИСКА ОПТИМАЛЬНОГО ПРОГРАММНОГО УПРАВЛЕНИЯ НА ОСНОВЕ РАЗЛОЖЕНИЙ ПО СИСТЕМЕ БАЗИСНЫХ ФУНКЦИЙ

Шаг 1. *Инициализация.* Выбрать метод из семейства мультиагентных алгоритмов и задать его параметры. Задать начальный масштаб усечения $L = 1$ и промежутки возможных значений коэффициентов разложения (3.44).

Шаг 2. *Нахождение наилучших значений коэффициентов разложения* путем применения выбранного мультиагентного алгоритма оптимизации.

Сгенерировать начальную популяцию, состоящую из NP агентов с положениями $(c_0, c_1, ..., c_{L-1})^T_{(j)}$, $j = 1, ..., NP$, где значения коэффициентов разложения удовлетворяют условиям (3.44).

По сгенерированным матрицам-столбцам $(c_0, c_1, ..., c_{L-1})^T_{(j)}$, $j = 1, ..., NP$, сформировать управления

$$u^{(j)}(t) = \text{sign}\left\{a, b, g^{(j)}(t)\right\},$$

где $\forall t \in T \quad g^{(j)}(t) = \sum_{i=0}^{L-1} c_i^{(j)} S(i,t)$, $\text{sign}(a,b,g^{(j)}(t)) = \begin{cases} a, & g^{(j)}(t) < 0, \\ b, & g^{(j)}(t) \geq 0, \end{cases}$.

В качестве базисной функций $\{S(i,t)\}_{i=0}^{L-1}$ в случае $t_0 = 0$ взять ортонормированную на отрезке $[0, t_1]$ систему нестационарных косинусоид (3.42).

Проинтегрировать NP систем дифференциальных уравнений (3.35) при начальном условии (3.36) с управлениями $u^{(1)}(\cdot), ..., u^{(NP)}(\cdot)$, например, методом Рунге–Кутты 4-го порядка. Для каждого агента получить соответствующие траектории $x^{(1)}(\cdot) = (x_1^{(1)}(\cdot), \ldots, x_n^{(1)}(\cdot)), \ldots, x^{(NP)}(\cdot) = (x_1^{(NP)}(\cdot), \ldots, x_n^{(NP)}(\cdot))$ и вычислить значения функционала $I^{(1)}, \ldots, I^{(NP)}$.

Если достигнуто максимальное число итераций мультиагентного алгоритма оптимизации, то из последней популяции следует выбрать наилучшее решение (положение наилучшего агента), т.е. наилучший вектор $(c_0, c_1, ..., c_{L-1})^T_*$ коэффициентов разложения и соответствующее значение функционала I^*_L.

Шаг 3. *Проверка условий завершения поиска.* Если $I^*_L < I^*_{L-1}$ (условие проверяется при $L > 1$), то положить $L = L + 1$ и перейти к шагу 2. Если условие $I^*_L \geq I^*_{L-1}$ выполняется на двух последовательных итерациях, то процедура поиска параметров оптимального программного управления завершается (иначе следует положить $L = L + 1$ и перейти к шагу 2). В качестве ответа выбирается наилучшее полученное решение.

3.3.3. Модельные примеры

Пример 3.1. Параметры постановки задачи приведены в табл. 3.1.

Таблица 3.1. Постановка задачи (пример 3.1)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = x_2(t) + \sin x_1(t) + u(t) \\ \dot{x}_2 = x_1(t) \cos x_2(t) u(t) \end{cases}$
Начальное условие	$x(0) = (0;0)^T$
Временной интервал	$t \in [0;1]$
Ограничения на управление	$-1 \leq u \leq 1$
Функционал (3.37)	$I(x_0, d) = x_2(1) \to \min$

Для нахождения структуры оптимального программного управления применим принцип максимума в форме, изложенной в [26, 40].

1. Составляем гамильтониан:

$$H(t, \psi, x, u) = \psi_1 \underbrace{\left[x_2 + \sin x_1 + u\right]}_{f_1(t,x,u)} + \psi_2 \underbrace{(x_1 \cos x_2 u)}_{f_2(t,x,u)} =$$

$$= [\psi_1 + \psi_2 x_1 \cos x_2] u + \psi_1 \left[x_2 + \sin x_1\right].$$

2. Найдем максимум гамильтониана по управлению:

$$H(t,\psi(t),x,u)\to\max_{-1\le u\le 1}.$$

Отсюда следует структура оптимального управления

$$u^*(t)=\begin{cases}-1,\text{если } \psi_1(t)+\psi_2(t)x_1(t)\cos x_2(t)<0,\\ 1,\text{если } \psi_1(t)+\psi_2(t)x_1(t)\cos x_2(t)>0.\end{cases}$$

3. Проверим условия трансверсальности вида

$$\delta F(t_1)-H(t_1)\delta t_1+\sum_{i=1}^{2}\psi(t_1)\delta x(t_1)=0\ .$$

Так как терминальный член функционала $F(t_1,x)=x_2$, а момент окончания $t_1=1$, т.е. задан, то вариация терминального члена

$$\delta F=\frac{\partial F}{\partial x_1}\delta x_1+\frac{\partial F}{\partial x_2}\delta x_2=\delta x_2,$$

а вариация момента окончания $\delta t_1=0$. Следовательно, получаем

$$\delta x_2+\psi_1(1)\delta x_1+\psi_2(1)\delta x_2=\psi_1(1)\delta x_1+[1+\psi_2(1)]\delta x_2=0.$$

Поскольку правый конец траектории свободен, то вариации $\delta x_1,\delta x_2$ произвольные. Отсюда $\psi_1(1)=0,\ \psi_2(1)=-1$.

В результате из применения необходимых условий оптимальности следует вывод о том, что оптимальное управление следует искать в классе кусочно-постоянных функций с неизвестными моментами переключения со значения 1 на -1 и наоборот. Поэтому задача может быть сведена к конечномерной проблеме поиска наилучших значений моментов переключения управления.

Из найденной структуры оптимального управления следует, что на последнем промежутке знакопостоянства управление принимает значение, равное 1, если выполняется неравенство $x_1(1)\cos x_2(1)<0$, и -1, если $x_1(1)\cos x_2(1)>0$. Полученные условия можно использовать при анализе полученных численных решений.

Пример 3.1 решался численно двумя описанными выше способами. В результате применения первого способа полученное значение $p=1,\ t_{П_0}=t_0=0$.

Параметры гибридного мультиагентного метода интерполяционного поиска: $NP=30$, $I_{\max}=50$, $M_1=2$, $M_2=5$, $PRT=0{,}01$, $nstep=5$, $b_2=8$.

Параметры мультиагентного метода с применением линейных регуляторов: $NP=101$, $NMAX=50$, $P_{\max}=10$, $k_S=0{,}1$, $k_l=5$, $h=0{,}0001$.

Результаты решения примера 3.1 представлены в табл. 3.2.

В результате применения второго способа получено значение $L=2$.

Параметры гибридного мультиагентного метода интерполяционного поиска: $NP=30$, $I_{\max}=50$, $M_1=2$, $M_2=5$, $PRT=0{,}01$, $nstep=5$, $b_2=8$.

Параметры мультиагентного метода с применением линейных регуляторов: $NP=41$, $NMAX=40$, $P_{\max}=20$, $k_S=1$, $k_l=5$, $h=0{,}0001$.

Таблица 3.2. Решение примера 3.1 (первый способ)

Метод оптимизации	Значения координат $(x_1(1), x_2(1))$	Точка переключения	Значение функционала I
Гибридный мультиагентный метод интерполяционного поиска	(0,444665; – 0,13598)	0,5	– 0,13598
Мультиагентный метод с применением линейных регуляторов	(0,44999; – 0,13450)	0,5	– 0,134550
Известное решение [61]	(0,440804; – 0,13593)	0,5	– 0,13593

Результаты решения примера 3.1 представлены в табл. 3.3. Оптимальное управление и траектория изображены на рис. 3.7.

Таблица 3.3. Решение примера 3.1 (второй способ)

Метод оптимизации	Значения координат $(x_1(1), x_2(1))$	Коэффициенты разложения	Значение функционала I
Гибридный мультиагентный метод интерполяционного поиска	(0,55055; – 0,13349)	5,05; 6,51	– 0,13598
Мультиагентный метод с применением линейных регуляторов	(0,57325; – 0,13208)	4,63; 5,66	– 0,13208

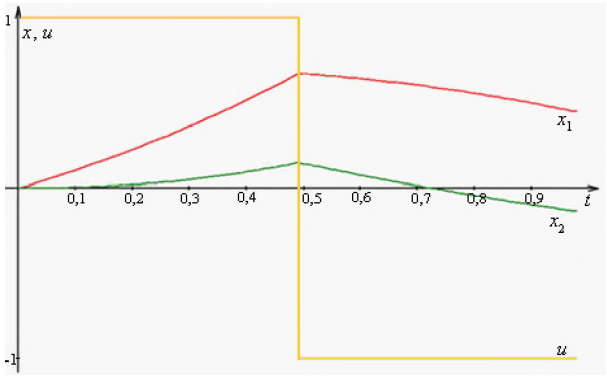


Рис. 3.7. Оптимальные управление и траектория в задаче 3.1

Пример 3.2. Параметры постановки задачи приведены в табл. 3.4.

Таблица 3.4. Постановка задачи (пример 3.2)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = x_2^2(t) + u(t) \\ \dot{x}_2 = 8\sin x_1(t) + x_1(t) - x_2(t) - u(t) \end{cases}$
Начальное условие	$x(0) = (-1;0)^T$
Временной интервал	$t \in [0;2]$
Ограничения на управление	$-1 \le u \le 2$
Функционал (3.37)	$I(x_0, d) = -x_1(2) \to \min$

Для нахождения структуры оптимального программного управления применим принцип максимума в форме, изложенной в [26, 40].

1. Составляем гамильтониан:

$$H(t,\psi,x,u)=\psi_1\underbrace{\left[x_2^2+u\right]}_{f_1(t,x,u)}+\psi_2\underbrace{(8\sin x_1+x_1-x_2-u)}_{f_2(t,x,u)}=$$

$$=[\psi_1-\psi_2]u+\psi_1 x_2^2+\psi_2(8\sin x_1+x_1-x_2).$$

2. Найдем максимум гамильтониана по управлению:

$$H(t,\psi(t),x,u)\to\max_{-1\le u\le 2}.$$

Отсюда следует структура оптимального управления

$$u^*(t)=\begin{cases}-1, \text{если } \psi_1(t)-\psi_2(t)<0,\\ 2, \text{если } \psi_1(t)-\psi_2(t)>0.\end{cases}$$

3. Проверим условия трансверсальности вида

$$\delta F(t_1)-H(t_1)\delta t_1+\sum_{i=1}^{2}\psi(t_1)\delta x(t_1)=0\ .$$

Так как терминальный член функционала $F(t_1,x)=-x_1$, а момент окончания $t_1=2$, т.е. задан, то вариация терминального члена

$$\delta F=\frac{\partial F}{\partial x_1}\delta x_1+\frac{\partial F}{\partial x_2}\delta x_2=-\delta x_1,$$

а вариация момента окончания $\delta t_1=0$. Следовательно, получаем

$$-\delta x_1+\psi_1(2)\delta x_1+\psi_2(2)\delta x_2=[\psi_1(2)-1]\,\delta x_1+\psi_2(2)\delta x_2=0.$$

Поскольку правый конец траектории свободен, то вариации $\delta x_1,\delta x_2$ произвольные. Отсюда $\psi_1(2)=1,\ \psi_2(2)=0$.

В результате из применения необходимых условий оптимальности следует вывод о том, что оптимальное управление следует искать в классе кусочно-постоянных функций с неизвестными моментами переключения со значения 2 на -1 и наоборот. Поэтому задача может быть сведена к конечномерной проблеме поиска наилучших значений моментов переключения управления.

Из найденной структуры оптимального управления следует, что на последнем промежутке знакопостоянства управление принимает значение, равное 2, так как выполняется неравенство $\psi_1(2)-\psi_2(2)=1>0$. Полученные условия можно использовать при анализе полученных численных решений.

Пример решался численно двумя описанными выше способами. В результате применения первого способа полученное значение $p=4$, $t_{П_0}=t_0=0$. Параметры гибридного мультиагентного метода интерполяционного поиска: $NP=30$, $I_{\max}=200$, $M_1=2$, $M_2=5$, $PRT=0{,}01$, $nstep=5$, $b_2=8$. Параметры мультиагентного метода с применением линейных регуляторов: $NP=801$, $NMAX=50$, $P_{\max}=10$, $k_S=0{,}1$, $k_\Lambda=5$, $h=0{,}0001$.

Результаты решения примера 3.2 представлены в табл. 3.5.

Таблица 3.5. Решение примера 3.2 (первый способ)

Метод оптимизации	Значения координат $(x_1(2), x_2(2))$	Точки переключения	Значение функционала I
Гибридный мультиагентный метод интерполяционного поиска	(16,36429; 6,06547)	(0,58; 1,35; 1,54; 1,77)	–16,36429
Мультиагентный метод с применением линейных регуляторов	(16,66516; 6,52485)	(0,59; 1,29; 1,53; 1,97)	–16,66516
Известное решение [61]	(16,76268; 6,35095)	(0,5; 1,25; 1,5; 1,8)	–16,76268

В результате применения второго способа получено значение $L = 4$.

Параметры гибридного мультиагентного метода интерполяционного поиска: $NP = 40$, $I_{\max} = 400$, $M_1 = 2$, $M_2 = 5$, $PRT = 0,01$, $nstep = 5$, $b_2 = 8$. Параметры мультиагентного метода с применением линейных регуляторов: $NP = 401$, $NMAX = 40$, $P_{\max} = 10$, $k_S = 1$, $k_I = 5$, $h = 0,0001$. Результаты решения примера 3.2 представлены в табл. 3.6.

Таблица 3.6. Решение примера 3.2 (второй способ)

Метод оптимизации	Значения координат $(x_1(2), x_2(2))$	Коэффициенты разложения	Значение функционала I
Гибридный мультиагентный метод интерполяционного поиска	(13,01829; 4,44509)	–0,23; 1,73; 1,78; 1,81	–13,01829
Мультиагентный метод с применением линейных регуляторов	(12,36497; 4,16002)	–0,28; 1,61; 1,79; 1,92	–12,36497

Оптимальное управление и траектория изображены на рис. 3.8.

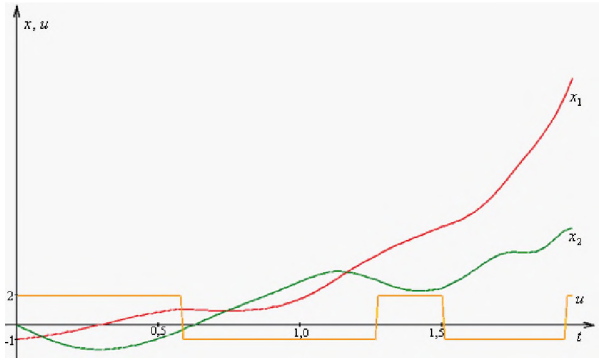


Рис. 3.8. Оптимальные управление и траектория в примере 3.2

Полученный результат соответствует выводам, сделанным из необходимых условий оптимальности (принципа максимума).

Пример 3.3. Постановка задачи приведена в табл. 3.7.

Таблица 3.7. Постановка задачи (пример 3.3)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = \dfrac{1}{\cos x_1(t) + 2} + 3\sin x_2(t) + u(t) \\ \dot{x}_2 = x_1(t) + x_2(t) + u(t) \end{cases}$
Начальное условие	$x(0) = (1;0)^T$
Временной интервал	$t \in [0;1,6]$
Ограничения на управление	$-2 \le u \le 1$
Функционал (3.37)	$I(x_0, d) = x_1(1,6) - \dfrac{1}{2} x_2(1,6) \to \min$

Для нахождения структуры оптимального программного управления применим принцип максимума в форме, используемой в [26, 40].

1. Составляем гамильтониан:

$$H(t,\psi,x,u) = \psi_1 \underbrace{\left[\frac{1}{\cos x_1 + 2} + 3\sin x_2 + u\right]}_{f_1(t,x,u)} + \psi_2 \underbrace{(x_1 + x_2 + u)}_{f_2(t,x,u)} =$$

$$= [\psi_1 + \psi_2]\, u + \psi_1 \left[\frac{1}{\cos x_1 + 2} + 3\sin x_2\right] + \psi_2 (x_1 + x_2).$$

2. Найдем максимум гамильтониана по управлению

$$H(t,\psi(t),x,u) \to \max_{-2 \le u \le 1}.$$

Отсюда следует структура оптимального управления

$$u^*(t) = \begin{cases} -2, \text{если } \psi_1(t) + \psi_2(t) < 0, \\ 1, \text{если } \psi_1(t) + \psi_2(t) > 0. \end{cases}$$

3. Проверим условия трансверсальности вида

$$\delta F(t_1) - H(t_1)\delta t_1 + \sum_{i=1}^{2} \psi(t_1)\delta x(t_1) = 0 \ .$$

Так как терминальный член функционала $F(t_1, x) = x_1 - \dfrac{x_2}{2}$, а момент окончания $t_1 = 1,6$, т.е. задан, то вариация терминального члена

$$\delta F = \frac{\partial F}{\partial x_1}\delta x_1 + \frac{\partial F}{\partial x_2}\delta x_2 = \delta x_1 - \frac{1}{2}\delta x_2,$$

а вариация момента окончания $\delta t_1 = 0$. Следовательно, получаем

$$\delta x_1 - \frac{1}{2}\delta x_2 + \psi_1(1,6)\delta x_1 + \psi_2(1,6)\delta x_2 = [1 + \psi_1(1,6)]\delta x_1 + [-\frac{1}{2} + \psi_2(1,6)]\delta x_2 = 0.$$

Поскольку правый конец траектории свободен, то вариации $\delta x_1, \delta x_2$ произвольные.

Отсюда получаем, что $\psi_1(1,6) = -1,\ \psi_2(1,6) = \frac{1}{2}$.

В результате из применения необходимых условий оптимальности следует вывод о том, что оптимальное управление следует искать в классе кусочно-постоянных функций с неизвестными моментами переключения со значения 1 на -2 и наоборот. Поэтому задача может быть сведена к конечномерной проблеме поиска наилучших значений моментов переключения управления.

Поскольку $\psi_1(1,6) + \psi_2(1,6) = -\frac{1}{2}$, то из структуры оптимального управления, найденной в пункте 2, следует, что на последнем промежутке знакопостоянства управление принимает значение, равное -2.

Из [10] известно, что в данном примере имеются два локальных минимума и один глобальный минимум (табл. 3.8).

Таблица 3.8. Известные решения примера 3.3

Решения примера 3.3	Значения координат $(x_1(1,6), x_2(1,6))$	Значение функционала I
Глобальный минимум	(3,46114; 12,88400)	–2,980856
Локальный минимум	(1,06257; 7,06928)	–2,472074
Локальный минимум	(–2,00836; –2,37696)	– 0,81988

Пример решался численно двумя способами.

В результате применения первого способа полученное значение $p = 1$, $t_{П_0} = t_0 = 0$.

Параметры гибридного мультиагентного метода интерполяционного поиска: $NP = 30$, $I_{\max} = 50$, $M_1 = 2$, $M_2 = 5$, $PRT = 0,01$, $nstep = 5$, $b_2 = 8$.

Параметры мультиагентного метода с применением линейных регуляторов: $NP = 101$, $NMAX = 50$, $P_{\max} = 10$, $k_g = 0,1$, $k_l = 5$, $h = 0,0001$.

Результаты решения примера 3.3 представлены в табл. 3.9.

Таблица 3.9. Решение примера 3.3 (первый способ)

Метод оптимизации	Значения координат $(x_1(1,6), x_2(1,6))$	Точки переключения	Значение функционала I
Гибридный мультиагентный метод интерполяционного поиска	(3,45449; 12,87062)	1,25	–2,98082
Мультиагентный метод с применением линейных регуляторов	(3,45949; 12,87562)	1,25	–2,97832
Известное решение [10]	(3,46114; 12,884)	1,26	–2,98086

В результате применения второго способа получено значение $L = 2$.

Параметры гибридного мультиагентного метода интерполяционного поиска: $NP = 30$, $I_{\max} = 50$, $M_1 = 2$, $M_2 = 5$, $PRT = 0,01$, $nstep = 5$, $b_2 = 8$.

Параметры мультиагентного метода с применением линейных регуляторов: $NP = 101$, $NMAX = 50$, $P_{\max} = 10$, $k_s = 1$, $k_l = 5$, $h = 0{,}0001$.

Результаты решения примера 3.3 представлены в табл. 3.10.

Таблица 3.10. Решение примера 3.3 (второй способ)

Метод оптимизации	Значения координат $(x_1(1{,}6), x_2(1{,}6))$	Коэффициенты разложения	Значение функционала I
Гибридный мультиагентный метод интерполяционного поиска	(3,45449; 12,87062)	0,22; 1,15	–2,98082
Мультиагентный метод с применением линейных регуляторов	(3,48523; 13,44901)	0,36; 0,99	–2,87218

Оптимальное управление и траектория изображены на рис. 3.9.

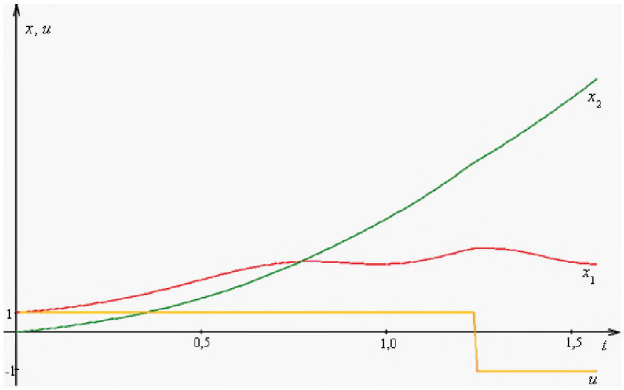


Рис. 3.9. Оптимальные управление и траектория в задаче 3.3

Полученные результаты соответствуют выводам, следующим из необходимых условий оптимальности. В связи с этим, использованный численный подход на основе мультиагентных алгоритмов показал свою эффективность. При анализе используемых способов решения видно, что алгоритм поиска оптимального управления на основе переключений показал лучшие результаты по сравнению с алгоритмом поиска, основанного на разложении по системе нестационарных косинусоид. Но, тем не менее, оба подхода с использованием мультиагентных алгоритмов дают решения, близкие к оптимальным, и могут успешно применяться для решения задач оптимального управления. Заметим, что рассмотренный в данном параграфе класс динамических систем (3.35) является достаточно узким, но часто встречающимся в модельных и прикладных задачах. Поэтому в следующих параграфах рассматриваются более общие постановки задач, для которых применяются различные способы параметризации законов управления. Процедура подбора параметров используемых мультиагентных алгоритмов оптимизации при решении конкретного примера, как правило, требует некоторых временных затрат. Однако, наличие общих рекомендаций, приведенных в параграфах 1.3.5 и 1.4.5, позволяет их существенно сократить.

3.4. ПОИСК ОПТИМАЛЬНОГО ПРОГРАММНОГО УПРАВЛЕНИЯ С ИСПОЛЬЗОВАНИЕМ БАЗИСА ФИНИТНЫХ ФУНКЦИЙ

3.4.1. Постановка задачи

Рассматривается решение задачи поиска оптимального программного управления, основанного на применении разложений по системам базисных функций и мультиагентного метода условной оптимизации. В частности, в качестве базисных функций используются кусочно-постоянные, кусочно-линейные, квадратичные и кубические сплайны, а в качестве мультиагентного метода применяется гибридный мультиагентный метод интерполяционного поиска (разд. 1.3).

На основе предложенных алгоритмов разработано программное обеспечение и исследована его эффективность при решении модельных примеров [10, 61]. Проведен сравнительный анализ влияния выбора параметров базисной системы на точность решаемой задачи.

Пусть поведение модели объекта управления описывается обыкновенным дифференциальным уравнением

$$\dot{x}(t) = f(t, x(t), u(t)) , \tag{3.45}$$

где x – вектор состояния системы, $x = (x_1, \ldots, x_n)^T \in R^n$; u – вектор управления, $u = (u_1, \ldots, u_q)^T \in U \subseteq R^q$, U – некоторое заданное множество допустимых значений управления, определяемое прямым произведением отрезков $[a_1, b_1] \times \ldots \times [a_q, b_q]$; $t \in T = [t_0, t_1]$ – промежуток времени функционирования системы; моменты начала процесса t_0 и окончания t_1 заданы; $f(t, x, u)$ – непрерывная вектор-функция; R^n – n-мерное евклидово пространство.

Начальное условие

$$x(t_0) = x_0 \tag{3.46}$$

задает начальное состояние системы.

Определим множество допустимых процессов $\mathcal{D}(t_0, x_0)$ как множество пар $d = (x(\cdot), u(\cdot))$, которые включают траекторию $x(\cdot)$ и управление $u(\cdot)$ (где $\forall t \in T: x(t) \in R^n$, $u(t) \in U$, функции $x(\cdot)$ непрерывны и кусочно-дифференцируемы, а $u(\cdot)$ кусочно-непрерывны), удовлетворяющие уравнению (3.45) с заданным начальным условием (3.46).

На множестве $\mathcal{D}(t_0, x_0)$ определим функционал качества управления

$$I(d) = F(x(t_1)), \tag{3.47}$$

где $F(x)$ – заданная непрерывная функция.

Требуется найти такую пару $d^* = (x^*(\cdot), u^*(\cdot)) \in \mathcal{D}(t_0, x_0)$, что

$$I(d^*) = \min_{d \in \mathcal{D}(t_0, x_0)} I(d). \tag{3.48}$$

3.4.2. Алгоритм решения задачи

Для частного случая систем (3.45), в которых структура оптимального программного управления согласно принципу максимума является релейной, предлагает-

ся искать приближенное решение в виде функции насыщения, которая должна гарантировать выполнение заданных ограничений на управление параллелепипедного типа, а аргументы функции насыщения – в виде линейной комбинации заданных базисных функций, которые используются в спектральном методе анализа и синтеза нелинейных систем управления [14–16]. Предварительно с помощью линейных преобразований промежуток времени функционирования системы $[t_0, t_1]$ приводится к стандартному отрезку [0;1].

Закон управления задается в виде функции насыщения sat, гарантирующей выполнение ограничений на управление:

$$u_j(t) = \mathrm{sat}\left\{g_j(t)\right\},\ j = 1,...,q; \tag{3.49}$$

где $\forall t \in T\ \ g_j(t) = \sum_{i=0}^{L^j-1} c_i^j S(i,t)$, $\mathrm{sat}\, g_j(t) = \begin{cases} a_j, & g_j(t) < a_j, \\ g_j(t), & a_j \le g_j(t) \le b_j, \\ b_j, & g_j(t) > b_j, \end{cases}$

$\left\{c_0^j, c_1^j, ..., c_{L^j-1}^j\right\}$ – неизвестные коэффициенты разложения по базисной системе функций $\{S(i,t)\}_{i=0}^{L^j-1}$; $L^j, j = 1,...,q$ – масштабы усечения разложений по каждой из координат вектора управления.

В качестве базисной системы функций $\{S(i,t)\}_{i=0}^{L^j-1}$ можно использовать сплайны:

$$S(i,t) = \widetilde{Sp}\left(\frac{t}{h} - i\right),\ h = \frac{1}{L^j - 1},\ i = 0,1,...,L^j - 1 \tag{3.50}$$

– финитные функции, порожденные сплайнами, на отрезке [0;1]:

$$\widetilde{Sp}(t) = \begin{cases} 2^{p-1}(1+t)^p, & t \in [-1;-\frac{1}{2}], \\ 1 - 2^{p-1}|t|^p, & t \in [-\frac{1}{2};\frac{1}{2}], \\ 2^{p-1}(1-t)^p, & t \in [\frac{1}{2};1], \\ 0, & t \notin [-1;1], \end{cases} \tag{3.51}$$

где $\widetilde{Sp}(t)$ при $p = 0$ задает кусочно-постоянный сплайн, при $p = 1$ – кусочно-линейный (крышки), при $p = 2$ – квадратичный, при $p = 3$ – кубический, а L^j – число базисных функций, используемых для представления j-й координаты вектора управления.

При решении задачи принимается гипотеза о том, что с ростом числа членов разложения (увеличении числа L^j) точность приближенного решения должна возрастать. Процесс увеличения чисел L^j можно завершить, если относительное приращение величины функционала станет меньше заранее заданного порогового значения, т.е. когда добавление новых слагаемых и, как следствие, усложнение закона управления не приносит существенного улучшения значения критерия оптимальности.

Текущее приближение решения задается в виде блочного вектора

$$(L^1,...,L^q\left|c_0^1,c_1^1,...,c_{L^1-1}^1\right|...\left|c_0^q,c_1^q,...,c_{L^q-1}^q\right)^T, \tag{3.52}$$

определяющего представление закона управления (3.49). На координаты вектора (3.52) накладываются интервальные ограничения вида

$$c_i^j \in \left[c_{i\min}^j, c_{i\max}^j\right], j=1,...,q;\ i=0,...,L^j-1. \tag{3.53}$$

Поскольку первый блок матрицы (3.52) содержит целочисленные переменные, а последующие блоки содержат вещественные переменные, задача параметрической минимизации функционала (3.47) на множестве допустимых векторов вида (3.52) относится к классу целочисленно-непрерывных. Ее решение связано с последовательным улучшением значений, как целочисленных переменных, так и всех остальных. Предлагается при поиске целочисленных решений применить идею последовательного покоординатного спуска с возвратом при неудачном шаге, а при поиске значений остальных переменных «заморозить» текущие значения целочисленных переменных, и применять мультиагентные методы оптимизации.

Введем обозначения матриц-столбцов

$$e_0=\begin{pmatrix}0\\ \vdots\\ 0\\ \vdots\\ 0\end{pmatrix}_{q\times 1},\ e_k=\begin{pmatrix}0\\ \vdots\\ 1\\ \vdots\\ 0\end{pmatrix}_{q\times 1}\ k-\text{я строка},\ 1\le k\le q.$$

Цикл, в котором величина масштаба усечения по каждой координате вектора управления изменяется на единицу, назовем проходом.

Описанная стратегия поиска реализуется в виде следующего алгоритма.

Шаг 1. *Задание параметров.* Задать параметр $p\in\{0,1,2,3\}$ сплайна (3.51), параметры выбранного мультиагентного алгоритма оптимизации, параметр для остановки алгоритма $\varepsilon>0$, максимальное число проходов. Задать матрицу-столбец начальных масштабов усечения $L^{(0)}=(\underbrace{1,...,1}_{q})^T$.

Шаг 2. *Инициализация прохода.* Положить $k=0$ (счетчик изменения номера координаты вектора управления).

Шаг 3. *Нахождение наилучшего вектора коэффициентов разложения* при заданных координатах вектора $L^{(k)}$ путем использования выбранного мультиагентного алгоритма.

Сформировать начальную популяцию, состоящую из NP особей, определяемых векторами $(c_0^{1(m)},c_1^{1(m)},...,c_{L^1-1}^{1(m)}\left|...\right|c_0^{q(m)},c_1^{q(m)},...,c_{L^q-1}^{q(m)})^T$, $m=1,...,NP$. Все координаты должны удовлетворять условиям (3.53).

Реализовать процедуры улучшения начальной популяции выбранным мультиагентным алгоритмом оптимизации.

Для вычисления значений целевой функции, роль которой играет функционал (3.47), следует выполнить следующие действия.

По сгенерированным коэффициентам сформировать закон управления в виде функции насыщения sat, гарантирующей выполнение ограничений на управление:

$$u_j^{(m)}(t) = \mathrm{sat}\left\{g_j^{(m)}(t)\right\},\ j = 1,...,q;\ m = 1,...,NP,$$

где $\forall t \in T\quad g_j^{(m)}(t) = \sum_{i=0}^{L^j-1} c_i^{j(m)} S(i,t)$, $\mathrm{sat}\, g_j^{(m)}(t) = \begin{cases} a_j, & g_j^{(m)}(t) < a_j, \\ g_j^{(m)}(t), & a_j \le g_j^{(m)}(t) \le b_j, \\ b_j, & g_j^{(m)}(t) > b_j. \end{cases}$

В качестве базисных функций $S(i,t)$ использовать сплайны:

$$S(i,t) = \widetilde{Sp}\left(\frac{t}{h} - i\right),\ h = \frac{1}{L^j - 1},\ i = 0,1,...,L^j - 1$$

– финитные функции, порожденные сплайнами, на отрезке $[0;1]$:

$$\widetilde{Sp}(t) = \begin{cases} 2^{p-1}(1+t)^p, & t \in [-1;-\frac{1}{2}], \\ 1 - 2^{p-1}|t|^p, & t \in [-\frac{1}{2};\frac{1}{2}], \\ 2^{p-1}(1-t)^p, & t \in [\frac{1}{2};1], \\ 0, & t \notin [-1;1], \end{cases}$$

где $\widetilde{Sp}(t)$ при $p = 0$ задает кусочно-постоянный сплайн, при $p = 1$ – кусочно-линейный (крышки), при $p = 2$ – квадратичный, при $p = 3$ – кубический.

Проинтегрировать NP систем дифференциальных уравнений (3.45) с управлениями $u^{(1)}(\cdot) = (u_1^{(1)}(\cdot),\ldots,u_q^{(1)}(\cdot)),...,u^{(NP)}(\cdot) = (u_1^{(NP)}(\cdot),...,u_q^{(NP)}(\cdot))$, например, методом Рунге–Кутты 4-го порядка. Для каждого агента получить соответствующие траектории $x^{(1)}(\cdot) = (x_1^{(1)}(\cdot),\ldots,x_n^{(1)}(\cdot)),\ldots,x^{(NP)}(\cdot) = (x_1^{(NP)}(\cdot),\ldots,x_n^{(NP)}(\cdot))$ и вычислить значения функционала $I^{(1)},\ldots,I^{(NP)}$.

Если достигнуто максимальное число итераций мультиагентного алгоритма оптимизации, то из последней популяции следует выбрать наилучшее решение (положение наилучшего агента), т.е. наилучший вектор коэффициентов разложения, и соответствующее значение функционала $I^*_{L^{(k)}}$.

Шаг 4. *Изменение вектора масштабов усечения по координатам вектора управления.*

Шаг 4.1. Если $k = 0$, то перейти к шагу 4.3.

Шаг 4.2. Проверить успешность локального поиска:

а) если $1 \le k < q$ и справедливо $I^*_{L^{(k)}} < I^*_{L^{(k-1)}}$, то перейти к шагу 4.3. Если $I^*_{L^{(k)}} \ge I^*_{L^{(k-1)}}$, то положить $L^{(k)} = L^{(k-1)}$ и перейти к шагу 4.3;

б) если $k = q$ и справедливо $I^*_{L^{(q)}} < I^*_{L^{(q-1)}}$, то принять $L^{*(q)} = L^{(q)}$. Если $I^*_{L^{(q)}} \ge I^*_{L^{(q-1)}}$, то принять $L^{*(q)} = L^{(q-1)}$.

Перейти к шагу 5.

Шаг 4.3. Положить $L^{(k+1)} = L^{(k)} + e_{k+1}, k = k+1$ и перейти к шагу 3.

Шаг 5. *Проверка условий окончания поиска.*

Шаг 5.1. Запомнить результат прохода $I^*_{L^{*(q)}}$ и соответствующий ему вектор коэффициентов разложения $L^{*(q)}$.

Шаг 5.2. Если получены результаты двух последовательных проходов, то найти значение относительного приращения величины функционала. Если оно меньше порогового значения ε, процедуру поиска завершить и перейти к шагу 6. Иначе перейти к реализации следующего прохода. Для этого в качестве начального вектора $L^{(0)}$ использовать вектор $L^{*(q)}$, полученный в результате последнего из двух сравниваемых проходов, и перейти к шагу 2. Если достигнуто максимальное число проходов, поиск завершить и перейти к шагу 6.

Шаг 6. *Выбор решения задачи.*

В качестве приближенного решения задачи выбрать наилучшее среди всех реализованных проходов.

3.4.3. Модельные примеры

Пример 3.4. Параметры постановки задачи приведены в табл. 3.11.

Таблица 3.11. Постановка задачи (пример 3.4)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = x_2(t) + \sin x_1(t) + u(t) \\ \dot{x}_2 = x_1(t)\cos x_2(t)u(t) \end{cases}$
Начальное условие	$x(0) = (0;0)^T$
Временной интервал	$t \in [0;1]$
Ограничения на управление	$-1 \le u \le 1$
Функционал (3.47)	$I(x_0, d) = x_2(1) \to \min$

В качестве базисных функций взяты четыре вида сплайнов, для каждого из которых было найдено наилучшее количество коэффициентов в разложении L и их числовые значения, на которые заданы ограничения $c_{\min} = -5$, $c_{\max} = 5$. Также определены наилучшие параметры гибридного мультиагентного алгоритма интерполяционного поиска: $NP = 20$, $I_{\max} = 10$, $M_1 = 2$, $M_2 = 5$, $PRT = 0{,}9$, $nstep = 5$, $b_2 = 8$. Величина параметра для остановки алгоритма $\varepsilon = 0{,}05$, величина шага интегрирования дифференциальных уравнений $h = 0{,}05$.

Результаты численного эксперимента приведены в табл. 3.12. Для случая кусочно-постоянного сплайна количество коэффициентов в разложении $L = 3$, для кусочно-линейного – $L = 4$, для квадратичного – $L = 4$, для кубического – $L = 4$.

Таблица 3.12. Решение примера 3.4

Вид сплайна	Значения координат $(x_1(1), x_2(1))$	Коэффициенты разложения	Значение функционала I
Кусочно-постоянный сплайн	(0,47; – 0,1362)	– 4,88; – 2,23; 4,31	– 0,13616
Кусочно-линейный сплайн	(0,4607; – 0,1354)	–5; –5; 4,72; 5	– 0,13539
Квадратичный сплайн	(0,4476; – 0,136)	–5; – 4,82; 4,35; 4,82	– 0,13604
Кубический сплайн	(0,4598; –0,1361)	–5; – 4,93; 3,63; 3,73	– 0,13612

На рис. 3.10 изображены графики траекторий и управления для случая кусочно-постоянного сплайна. Известное решение примера 3.4 представлено в [61]: значения координат $\left(x_1(1), x_2(1)\right) = \left(0,440804; -0,13593\right)$, значение функционала $I = -\ 0,13593$.

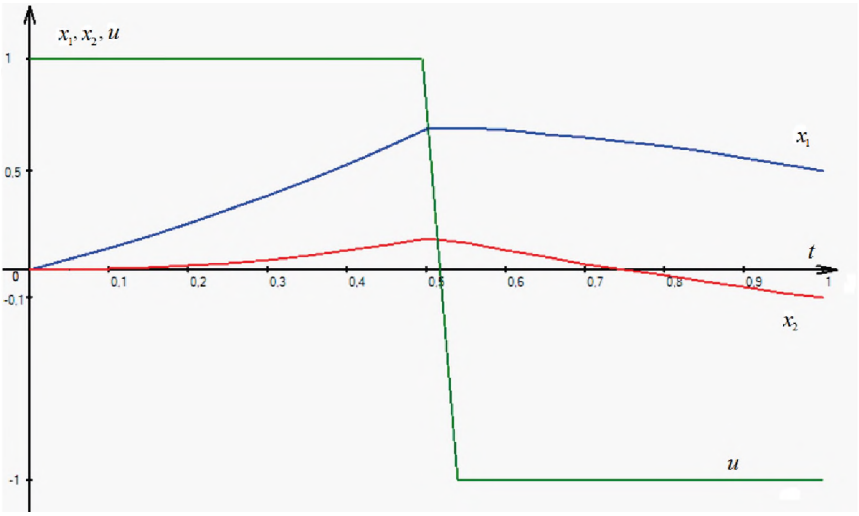


Рис. 3.10. Оптимальные управление и траектория в задаче 3.4

Пример 3.5. Параметры постановки задачи приведены в табл. 3.13.

Таблица 3.13. Постановка задачи (пример 3.5)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = x_2^2(t) + u(t) \\ \dot{x}_2 = 8\sin x_1(t) + x_1(t) - x_2(t) - u(t) \end{cases}$
Начальное условие	$x(0) = (-1;0)^T$
Временной интервал	$t \in [0;2]$
Ограничения на управление	$-1 \le u \le 2$
Функционал (3.47)	$I(x_0, d) = -x_1(2) \to \min$

Решим описанную задачу поиска оптимального программного управления, используя сплайны (3.51) разных степеней. Для каждого сплайна найдено наилучшее количество коэффициентов в разложении L и их значения, на которые заданы ограничения $c_{\min} = -50$, $c_{\max} = 50$. Также определены наилучшие параметры гибридного мультиагентного алгоритма интерполяционного поиска: $NP = 20$, $I_{\max} = 10$, $M_1 = 2$, $M_2 = 5$, $PRT = 0,9$, $nstep = 5$, $b_2 = 8$. Величина параметра для остановки алгоритма $\varepsilon = 0,05$, величина шага интегрирования дифференциальных уравнений $h = 0,05$.

Результаты решения задачи 3.5 приведены в табл. 3.14. Для случая кусочно-постоянного сплайна найденное количество коэффициентов в разложении $L = 8$, для кусочно-линейного – $L = 7$, для квадратичного – $L = 6$, для кубического – $L = 3$. На рис. 3.11 изображены полученные графики траекторий и управления для случая кубического сплайна.

Известное решение примера 3.5 представлено в [61]: значения координат $\left(x_1(2), x_2(2)\right) = \left(16,76268; 6,35095\right)$, значение функционала $I = -16,76268$.

Таблица 3.14. Решение примера 3.5

Вид сплайна	Значения координат $(x_1(2), x_2(2))$	Коэффициенты разложения	Значение функционала I
Кусочно-постоянный сплайн	(15,7198; 6,5035)	–39,32; –34,53; –5,29; 12,77; 14,9; 16,76; 35; 48,29	–15,719801
Кусочно-линейный сплайн	(15,7205; 6,50396)	–21,32; –19,88; 7,77; 16,07; 25,78; 29,52; 37,77	–15,720474
Квадратичный сплайн	(15,7205; 6,5039)	–50; –9,59; 17,7; 19,24; 20,35; 28,29	–15,720477
Кубический сплайн	(15,7205; 6,5039)	–24,92; 11,29; 15,63	–15,720477

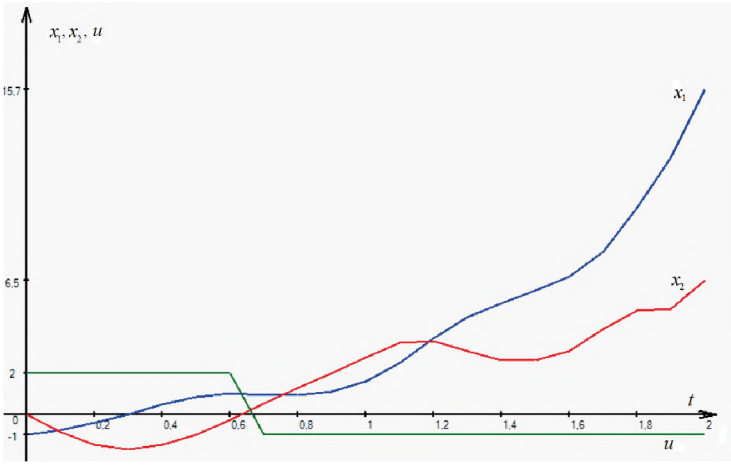


Рис. 3.11. Оптимальные управление и траектория в задаче 3.5

Пример 3.6. Постановка задачи приведена в табл. 3.15.

Таблица 3.15. Постановка задачи (пример 3.6)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = \dfrac{1}{\cos x_1(t) + 2} + 3\sin x_2(t) + u(t) \\ \dot{x}_2 = x_1(t) + x_2(t) + u(t) \end{cases}$
Начальное условие	$x(0) = (1;0)^T$
Временной интервал	$t \in [0;1,6]$
Ограничения на управление	$-2 \le u \le 1$
Функционал (3.47)	$I(x_0, d) = x_1(1,6) - \dfrac{1}{2} x_2(1,6) \to \min$

Для решения задачи выберем следующую базисную систему: финитные функции порожденные сплайнами (3.51) и, следовательно, рассмотрим четыре случая решения примера 3.6. Для каждой выбранной базисной функции найдено наилучшее количество коэффициентов в разложении L и их значения. На числовые значения ко-

эффициентов наложены ограничения $c_{\min} = -50$, $c_{\max} = 50$. Также определены наилучшие параметры гибридного мультиагентного алгоритма интерполяционного поиска: $NP = 20$, $I_{\max} = 10$, $M_1 = 2$, $M_2 = 5$, $PRT = 0{,}9$, $nstep = 5$, $b_2 = 8$. Параметр для остановки алгоритма $\varepsilon = 0{,}05$, величина шага интегрирования $h = 0{,}05$.

Результаты численного эксперимента приведены в табл. 3.16. Для случая кусочно-постоянного сплайна количество коэффициентов в разложении $L = 6$, для кусочно-линейного – $L = 5$, для квадратичного – $L = 4$, для кубического – $L = 3$. На рис. 3.12 изображены графики траекторий и управления для случая квадратичного сплайна.

Таблица 3.16. Решение примера 3.6

Вид сплайна	Значения координат $(x_1(1{,}6), x_2(1{,}6))$	Коэффициенты разложения	Значение функционала I
Кусочно-постоянный сплайн	(3,4923; 12,9461)	–39,88; –38,07; –28,92; –3,97; –1,67; 42,16	–2,98073
Кусочно-линейный сплайн	(3,4923; 12,9461)	–18,91; –9,83; –7,93; –6,24; 43	–2,98073
Квадратичный сплайн	(3,4923; 12,9461)	–16,35; –15,97; –13,85; 45,72	–2,98073
Кубический сплайн	(3,4923; 12,9461)	– 48,23; – 42,74; 20,06	–2,98073

Известное решение примера 3.6 представлено в [10]: значения координат $(x_1(1{,}6), x_2(1{,}6)) = (3{,}46114; 12{,}884)$, значение функционала $I = -2{,}98086$.

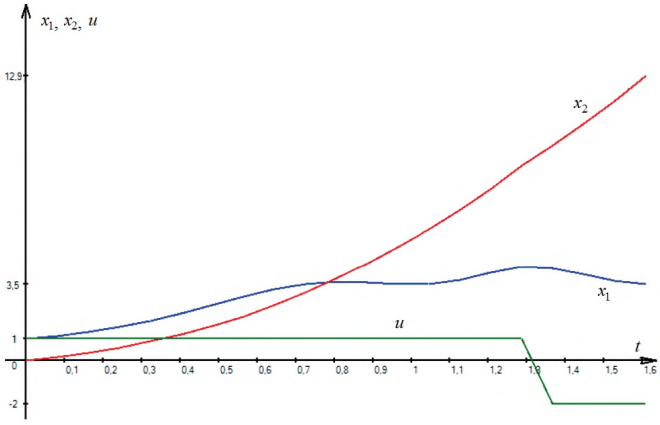


Рис. 3.12. Оптимальные управление и траектория в задаче 3.6

Исходя из полученных результатов, описанный подход на основе использования финитных функций показал свою эффективность при решении модельных примеров. Применение кусочно-постоянного, кусочно-линейного, квадратичного и кубического сплайнов позволяет найти программное управление, близкое к оптимальному. При решении примера 3.4 значение функционала оказалось немного лучше, чем в известном решении. Также подобраны параметры гибридного мультиагентного алгоритма интерполяционного поиска, при которых было достигнуто наилучшее значение функционала в рассмотренных модельных примерах, что позволяет их использовать при решении прикладных задач оптимального управления.

3.5. ПОИСК ОПТИМАЛЬНОГО ПРОГРАММНОГО УПРАВЛЕНИЯ НА ОСНОВЕ РАДИАЛЬНО-БАЗИСНЫХ ФУНКЦИЙ

3.5.1. Постановка задачи

Рассматривается способ решения задачи поиска оптимального программного управления, основанный на применении разложений по системе радиально-базисных функций и мультиагентного метода условной оптимизации. В качестве радиально-базисных функций рассматриваются: функция Гаусса, мульти-квадратичная функция, обратная квадратичная функция и обратная мульти-квадратичная функция, которые используются при решении задач интерполяции, синтезе нелинейных систем управления и в нейронных сетях [73, 146]. В качестве мультиагентного метода применяется гибридный мультиагентный метод интерполяционного поиска (разд. 1.3).

На основе предложенных алгоритмов разработано программное обеспечение и исследована его эффективность при решении модельных примеров [10, 61]. Проведен сравнительный анализ влияния выбора параметров базисной системы на точность решаемой задачи.

Пусть поведение модели объекта управления описывается обыкновенным дифференциальным уравнением

$$\dot{x}(t) = f(t, x(t), u(t)), \tag{3.54}$$

где x – вектор состояния системы, $x = (x_1, \ldots, x_n)^T \in R^n$; u – вектор управления, $u = (u_1, \ldots, u_q)^T \in U \subseteq R^q$, U – некоторое заданное множество допустимых значений управления, определяемое прямым произведением отрезков $[a_1, b_1] \times \ldots \times [a_q, b_q]$; $t \in T = [t_0, t_f]$ – промежуток времени функционирования системы; моменты начала процесса t_0 и окончания t_f заданы; $f(t, x, u)$ – непрерывная вектор-функция; R^n – n-мерное евклидово пространство.

Начальное условие

$$x(t_0) = x_0 \tag{3.55}$$

задает начальное состояние системы.

Определим множество допустимых процессов $\mathcal{D}(t_0, x_0)$ как множество пар $d = (x(\cdot), u(\cdot))$, которые включают траекторию $x(\cdot)$ и управление $u(\cdot)$ (где $\forall t \in T: x(t) \in R^n$, $u(t) \in U$, функции $x(\cdot)$ непрерывны и кусочно-дифференцируемы, а $u(\cdot)$ кусочно-непрерывны), удовлетворяющие уравнению (3.54) с заданным начальным условием (3.55).

На множестве $\mathcal{D}(t_0, x_0)$ определим функционал качества управления

$$I(d) = F(x(t_f)), \tag{3.56}$$

где $F(x)$ – заданная непрерывная функция.

Требуется найти такую пару $d^* = (x^*(\cdot), u^*(\cdot)) \in \mathcal{D}(t_0, x_0)$, что

$$I(d^*) = \min_{d \in \mathcal{D}(t_0, x_0)} I(d). \tag{3.57}$$

3.5.2. Алгоритм решения задачи

Для частного случая систем (3.54), в котором структура оптимального программного управления согласно принципу максимума является релейной, предлагается искать приближенное решение в виде функции насыщения, которая должна гарантировать выполнение заданных ограничений на управление параллелепипедного типа, а аргументы функции насыщения – в виде линейной комбинации заданных радиально-базисных функций.

Закон управления задается в виде функции насыщения sat, гарантирующей выполнение ограничений на управление:

$$u_j(t) = \mathrm{sat}\left\{g_j(t)\right\},\ j = 1,...,q, \tag{3.58}$$

где $\forall t \in T\ \ g_j(t) = \sum_{i=0}^{L^j-1} c_i^j \varphi(|t - t^i|)$, $\mathrm{sat}\, g_j(t) = \begin{cases} a_j, & g_j(t) < a_j, \\ g_j(t), & a_j \le g_j(t) \le b_j, \\ b_j, & g_j(t) > b_j, \end{cases}$

$\left\{c_0^j, c_1^j, ..., c_{L^j-1}^j\right\}$ – неизвестные коэффициенты разложения по базисной системе функций $\{\varphi(|t - t^i|)\}_{i=0}^{L^j-1}$; $L^j \ge 1, j = 1,...,q$ – масштабы усечения разложений по каждой из координат вектора управления.

В качестве радиально-базисной системы функций $\{\varphi(|t - t^i|)\}_{i=0}^{L^j-1}$ можно использовать:

– функцию Гаусса

$$\varphi(r) = e^{-(\mu r)^2}, \tag{3.59}$$

– мульти-квадратичную функцию

$$\varphi(r) = \sqrt{1 + (\mu r)^2}, \tag{3.60}$$

– обратную квадратичную функцию

$$\varphi(r) = 1 \Big/ \left(1 + (\mu r)^2\right), \tag{3.61}$$

– обратную мульти-квадратичную функцию

$$\varphi(r) = 1 \Big/ \sqrt{1 + (\mu r)^2}. \tag{3.62}$$

Здесь μ – масштабный коэффициент, t^i – базовая точка, $r = |t - t^i|$. Базовые точки задаются следующим образом: $t^i = t_0 + ih,\ i = 0,...,L^j - 1$, где $h = \dfrac{t_f - t_0}{L^j - 1}$, $j = 1,...,q$.

При решении задачи принимается гипотеза о том, что с ростом числа членов разложения (увеличении числа L^j) точность приближенного решения должна возрастать. Процесс увеличения чисел L^j можно завершить, если относительное приращение величины функционала станет меньше заранее заданного порогового значения, т.е. когда добавление новых слагаемых и, как следствие, усложнение закона управления не приносит существенного улучшения значения критерия оптимальности.

Текущее приближение решения задается в виде блочного вектора

$$(L^1,...,L^q\,|\,c_0^1,c_1^1,...,c_{L^1-1}^1\,|...|\,c_0^q,c_1^q,...,c_{L^q-1}^q)^T, \tag{3.63}$$

определяющего представление закона управления (3.58). На координаты вектора (3.63) накладываются интервальные ограничения вида

$$c_i^j \in \left[c_{i\min}^j, c_{i\max}^j\right], j=1,...,q;\ i=0,...,L^j-1. \tag{3.64}$$

Поскольку первый блок матрицы (3.63) содержит целочисленные переменные, а последующие блоки содержат вещественные переменные, задача параметрической минимизации функционала (3.56) на множестве допустимых векторов вида (3.63) относится к классу целочисленно-непрерывных. Ее решение связано с последовательным улучшением значений, как целочисленных переменных, образующих вектор масштабов усечения $L=(L^1,...,L^q)$, так и действительных переменных, образующих вектор $C=(c_0^1,c_1^1,...,c_{L^1-1}^1\,|...|\,c_0^q,c_1^q,...,c_{L^q-1}^q)^T$. Предлагается при поиске целочисленных решений реализовать идею последовательного покоординатного спуска с возвратом при неудачном шаге, а при поиске значений остальных переменных "заморозить" текущие значения целочисленных переменных и применить мультиагентные методы оптимизации (разд. 1.3 –1.5).

Введем обозначения матриц-столбцов

$$e_0=\begin{pmatrix}0\\ \vdots\\ 0\\ \vdots\\ 0\end{pmatrix}_{q\times 1},\quad e_k=\begin{pmatrix}0\\ \vdots\\ 1\\ \vdots\\ 0\end{pmatrix}_{q\times 1}\quad k-\text{я строка},\ 1\le k\le q.$$

Цикл, в котором величина масштаба усечения по каждой координате вектора управления изменяется на единицу, назовем проходом.

Описанная стратегия поиска реализуется в виде следующего алгоритма.

Шаг 1. *Задание параметров.* Выбрать тип радиально-базисных функций, параметры выбранного мультиагентного алгоритма оптимизации, параметр для остановки алгоритма ε, максимальное число проходов, масштабный коэффициент μ и положение базовых точек t^i, $i=0,...,L^j-1$; $j=1,...,q$. Задать матрицу-столбец начальных масштабов усечения $L^{(0)}=(\underbrace{1,...,1}_{q})^T$.

Шаг 2. *Инициализация прохода.* Положить $k=0$ (счетчик числа итераций изменения вектора масштабов усечения).

Шаг 3. *Нахождение наилучшего вектора коэффициентов разложения* при заданных координатах вектора $L^{(k)}$ путем использования выбранного мультиагентного алгоритма.

Сформировать начальную популяцию, состоящую из NP агентов, определяемых векторами $C^{(m)}=(c_0^{1(m)},c_1^{1(m)},...,c_{L^1-1}^{1(m)}\,|...|\,c_0^{q(m)},c_1^{q(m)},...,c_{L^q-1}^{q(m)})^T$, $m=1,...,NP$. Все координаты векторов должны удовлетворять условиям (3.64).

Реализовать процедуры улучшения начальной популяции выбранным мультиагентным алгоритмом оптимизации.

Для вычисления значений целевой функции, роль которой играет функционал (3.56), следует выполнить следующие действия.

По сгенерированным коэффициентам сформировать закон управления в виде функции насыщения sat, гарантирующей выполнение ограничений на управление:

$$u_j^{(m)}(t) = \mathrm{sat}\left\{g_j^{(m)}(t)\right\},\ j = 1,...,q;\ m = 1,...,NP,$$

где $\forall t \in T\ \ g_j^{(m)}(t) = \sum_{i=0}^{L^j-1} c_i^{j(m)} \varphi(|t - t^i|)$, $\mathrm{sat}\, g_j^{(m)}(t) = \begin{cases} a_j, & g_j^{(m)}(t) < a_j, \\ g_j^{(m)}(t), & a_j \le g_j^{(m)}(t) \le b_j, \\ b_j, & g_j^{(m)}(t) > b_j. \end{cases}$

В качестве радиально-базисных функций $\varphi(|t - t^i|)$ можно использовать:

а) функцию Гаусса $\varphi(r) = e^{-(\mu r)^2}$,

б) мульти-квадратичную функцию $\varphi(r) = \sqrt{1 + (\mu r)^2}$,

в) обратную квадратичную функцию $\varphi(r) = 1/\left(1 + (\mu r)^2\right)$,

г) обратную мульти-квадратичную функцию $\varphi(r) = 1/\sqrt{1 + (\mu r)^2}$.

Здесь μ – масштабный коэффициент, t^i – базовая точка, $r = |t - t^i|$. Базовые точки задаются следующим образом: $t^i = t_0 + ih,\ i = 0,...,L^j - 1$, где $h = \dfrac{t_f - t_0}{L^j - 1},\ j = 1,...,q$.

Проинтегрировать NP систем дифференциальных уравнений (3.54) с управлениями $u^{(1)}(\cdot) = (u_1^{(1)}(\cdot),\ldots,u_q^{(1)}(\cdot)),...,u^{(NP)}(\cdot) = (u_1^{(NP)}(\cdot),...,u_q^{(NP)}(\cdot))$, например, методом Рунге–Кутты 4-го порядка. Для каждого агента получить соответствующие траектории $x^{(1)}(\cdot) = (x_1^{(1)}(\cdot),\ldots,x_n^{(1)}(\cdot)),\ldots,x^{(NP)}(\cdot) = (x_1^{(NP)}(\cdot),\ldots,x_n^{(NP)}(\cdot))$ и вычислить значения функционала $I^{(1)},\ldots,I^{(NP)}$.

Если достигнуто максимальное число итераций мультиагентного алгоритма оптимизации, то из последней популяции следует выбрать наилучшее решение (положение наилучшего агента), т.е. наилучший вектор коэффициентов разложения, и соответствующее значение функционала $I^*_{L^{(k)}}$.

Шаг 4. *Изменение вектора масштабов усечения по координатам вектора управления.*

Шаг 4.1. Если $k = 0$, то перейти к шагу 4.3. Иначе перейти к шагу 4.2.

Шаг 4.2. Проверить успешность локального поиска:

а) если $1 \le k < q$ и справедливо $I^*_{L^{(k)}} < I^*_{L^{(k-1)}}$, то перейти к шагу 4.3. Если $I^*_{L^{(k)}} \ge I^*_{L^{(k-1)}}$, то положить $L^{(k)} = L^{(k-1)}$ и перейти к шагу 4.3;

б) если $k = q$ и справедливо $I^*_{L^{(q)}} < I^*_{L^{(q-1)}}$, то принять $L^{*(q)} = L^{(q)}$. Если $I^*_{L^{(q)}} \ge I^*_{L^{(q-1)}}$, то принять $L^{*(q)} = L^{(q-1)}$.

Перейти к шагу 5.

Шаг 4.3. Положить $L^{(k+1)} = L^{(k)} + e_{k+1}, k = k + 1$ и перейти к шагу 3.

Шаг 5. *Проверка условий окончания поиска.*

Шаг 5.1. Запомнить результат прохода $I^*_{L^{*(q)}}$ и соответствующий ему вектор коэффициентов разложения $L^{*(q)}$.

Шаг 5.2. Если получены результаты двух последовательных проходов, то найти значение относительного приращения величины функционала. Если оно меньше порогового значения ε, процедуру поиска завершить и перейти к шагу 6. Иначе перейти к реализации следующего прохода. Для этого в качестве начального вектора $L^{(0)}$ использовать вектор $L^{*(q)}$, полученный в результате последнего из двух сравниваемых проходов, и перейти к шагу 2. Если достигнуто максимальное число проходов, поиск завершить и перейти к шагу 6.

Шаг 6. *Окончание поиска.*

В качестве полученного приближенного решения выбрать наилучший результат среди реализованных проходов.

3.5.3. Модельные примеры

Пример 3.7. Параметры постановки задачи приведены в табл. 3.17.

Таблица 3.17. Постановка задачи (пример 3.7)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = x_2(t) + \sin x_1(t) + u(t) \\ \dot{x}_2 = x_1(t)\cos x_2(t)\,u(t) \end{cases}$
Начальное условие	$x(0) = (0;0)^T$
Временной интервал	$t \in [0;1]$
Ограничения на управление	$-1 \le u \le 1$
Функционал (3.56)	$I(d) = x_2(1) \to \min$

Решим пример 3.7 с помощью рассмотренного алгоритма. Для этого требуется выбрать одну из указанных радиально-базисных функций: функцию Гаусса, мультиквадратичную функцию, обратную квадратичную функцию или обратную мультиквадратичную функцию, а также задать параметры мультиагентного алгоритма условной оптимизации. Выбраны значения параметра остановки алгоритма $\varepsilon = 0{,}05$ и шага интегрирования $h = 0{,}001$, максимальное число проходов равно 10.

Рассмотрено четыре случая поиска оптимального программного управления с разными радиально-базисными функциями (3.59) – (3.62). Для каждого случая найдено наилучшее количество L коэффициентов разложения и их числовые значения, а также определены наилучшие параметры гибридного мультиагентного алгоритма интерполяционного поиска: $NP = 20$, $I_{\max} = 10$, $M_1 = 2$, $M_2 = 5$, $PRT = 0{,}9$, $nstep = 5$, $b_2 = 8$. Масштабный коэффициент для радиально-базисных функций $\mu = 2L$, максимального числа проходов 10. На значения коэффициентов в разложении наложены ограничения $c_{\min} = -5$, $c_{\max} = 5$.

Результаты численного эксперимента приведены в табл. 3.18. При использовании функции Гаусса количество коэффициентов в разложении $L = 6$, мультиквадратичной функции – $L = 6$, обратной квадратичной функции – $L = 4$ и обратной мульти-квадратичной функции – $L = 6$.

На рис. 3.13 изображены графики траекторий и управления при применении мульти-квадратичной функции (для остальных трех случаев графики аналогичные).

Известное решение примера 3.7 представлено в [61]: значения координат $(x_1(1), x_2(1)) = (0{,}440804;\ -0{,}13593)$, значение функционала $I = -\,0{,}13593$.

Таблица 3.18. Результаты решения примера 3.7

Радиально-базисные функции	Значения координат $x_1(1), x_2(1))$	Коэффициенты разложения	Значение функционала I
Функция Гаусса	0,4512; – 0,1361	– 5; – 4,65; – 4,64; 3,91; 4,24; 5	– 0,136065
Мульти-квадратичная функция	0,47; – 0,1362	– 4,06; – 3,28; – 2,34; 2,69; 3,4; 4,29	– 0,136163
Обратная квадратичная функция	0,4403; – 0,1354	– 5; – 5; 5; 5	– 0,135446
Обратная мульти-квадратичная функция	0,4649; – 0,1357	– 4,38; – 4,38; – 4,13; 3,85; 4,33; 4,66	– 0,135674

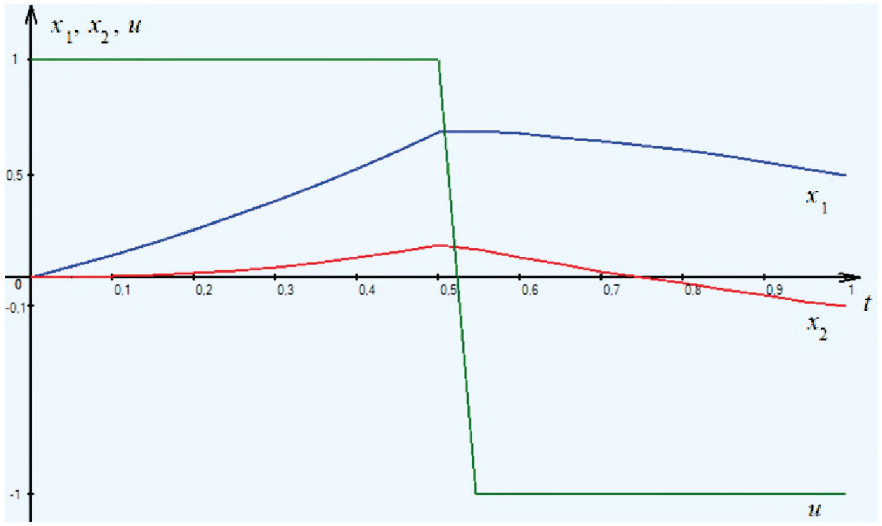


Рис. 3.13. Оптимальные управление и траектория в примере 3.7

Пример 3.8. Параметры постановки задачи приведены в табл. 3.19.

Таблица 3.19. Постановка задачи (пример 3.8)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = x_2^2(t) + u(t) \\ \dot{x}_2 = 8\sin x_1(t) + x_1(t) - x_2(t) - u(t) \end{cases}$
Начальное условие	$x(0) = (-1;0)^T$
Временной интервал	$t \in [0;2]$
Ограничения на управление	$-1 \le u \le 2$
Функционал (3.56)	$I(d) = -x_1(2) \to \min$

Решим пример 3.8 с помощью рассмотренного алгоритма. Аналогично решению примера 3.7 рассмотрено четыре случая поиска оптимального программного управления с разными радиально-базисными функциями (3.59) – (3.62): функцией Гаусса, мульти-квадратичной функцией, обратной квадратичной функцией и обратной мульти-квадратичной функцией. Для каждого случая найдено наилучшее количество

L коэффициентов разложения и их числовые значения, а также определены наилучшие параметры гибридного мультиагентного алгоритма интерполяционного поиска: $NP = 20$, $I_{\max} = 10$, $M_1 = 2$, $M_2 = 5$, $PRT = 0{,}9$, $nstep = 5$, $b_2 = 8$. Выбраны значения параметра остановки алгоритма $\varepsilon = 0{,}05$, шага интегрирования $h = 0{,}001$, масштабного коэффициента для радиально-базисных функций $\mu = 2L$, максимального числа проходов 10. На значения коэффициентов в разложении наложены ограничения $c_{\min} = -50$, $c_{\max} = 50$.

Результаты численного эксперимента приведены в табл. 3.20. При использовании функции Гаусса количество коэффициентов в разложении $L = 6$, мульти-квадратичной функции – $L = 4$, обратной квадратичной функции – $L = 4$ и обратной мульти-квадратичной функции – $L = 4$.

Таблица 3.20. Результаты решения примера 3.8

Радиально-базисные функции	Значения координат $(x_1(2), x_2(2))$	Коэффициенты разложения	Значение функционала I
Функция Гаусса	15,7656; 6,5485	– 23,65; – 15,66; 48,75; 60,69; 62,5; 75,89	– 15,765599
Мульти-квадратичная функция	15,7656; 6,5485	– 66,58; – 46,35; – 18,68; 42,49	– 15,765599
Обратная квадратичная функция	15,7656; 6,54853	– 76,53; 0,91; 84,76; 87,32	– 15,765598
Обратная мульти-квадратичная функция	15,7656; 6,5485	– 70,73; 2,41; 51,72; 53,79	– 15,765599

На рис. 3.14 изображены графики траектории и управления, полученные при применении функции Гаусса (для других радиально-базисных функций графики аналогичные). Известное решение примера 3.8 представлено в [61]: значения координат $(x_1(2), x_2(2)) = (16{,}76268;\ 6{,}35095)$, значение функционала $I = -16{,}76268$.

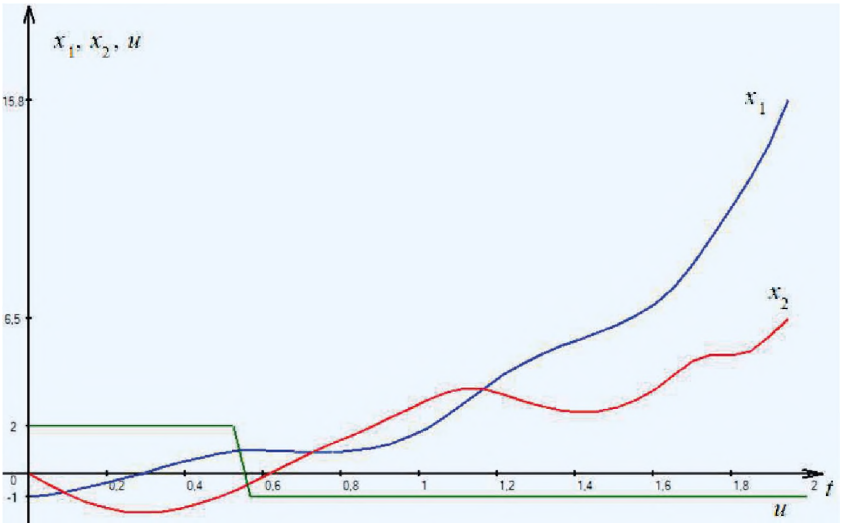


Рис. 3.14. Оптимальные управление и траектория в примере 3.8

Пример 3.9. Постановка задачи приведена в табл. 3.21.

Таблица 3.21. Постановка задачи (пример 3.9)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = \dfrac{1}{\cos x_1(t) + 2} + 3\sin x_2(t) + u(t) \\ \dot{x}_2 = x_1(t) + x_2(t) + u(t) \end{cases}$
Начальное условие	$x(0) = (1;0)^T$
Временной интервал	$t \in [0;1,6]$
Ограничения на управление	$-2 \le u \le 1$
Функционал (3.56)	$I(d) = x_1(1,6) - \dfrac{1}{2} x_2(1,6) \to \min$

Для решения примера 3.9 требуется выбрать одну из указанных радиально-базисных функций: функцию Гаусса, мульти-квадратичную функцию, обратную квадратичную функцию или обратную мульти-квадратичную функцию, а также выбрать мультиагентный алгоритм условной оптимизации и задать его параметры.

Рассмотрено четыре случая поиска оптимального программного управления с разными радиально-базисными функциями (3.59) – (3.62). Для каждого случая найдено наилучшее количество L коэффициентов разложения и их числовые значения, а также определены наилучшие параметры гибридного мультиагентного алгоритма интерполяционного поиска: $NP = 20$, $I_{\max} = 10$, $M_1 = 2$, $M_2 = 5$, $PRT = 0,9$, $nstep = 5$, $b_2 = 8$. Выбраны значения параметра остановки алгоритма $\varepsilon = 0,05$, шага интегрирования $h = 0,001$, масштабного коэффициента для радиально-базисных функций $\mu = 2L$, максимального числа проходов 10. На значения коэффициентов в разложении наложены ограничения $c_{\min} = -50$, $c_{\max} = 50$.

Результаты численного эксперимента приведены в табл. 3.22. При использовании функции Гаусса количество коэффициентов в разложении $L = 8$, мульти-квадратичной функции – $L = 4$, обратной квадратичной функции – $L = 4$ и обратной мульти-квадратичной функции – $L = 4$. На рис. 3.15 изображены графики траектории и управления при применении функции Гаусса.

Таблица 3.22. Результаты решения примера 3.9

Радиально-базисные функции	Значения координат $(x_1(1,6), x_2(1,6))$	Коэффициенты разложения	Значение функционала I
Функция Гаусса	3,471; 12,9043	– 50; – 50; – 45,89; – 42,99; – 35,41; – 28,12; 25,59; 26	– 2,981125
Мульти-квадратичная функция	3,471; 12,9043	– 29,19; 21,29; 29,14; 39,95	– 2,981125
Обратная квадратичная функция	3,4607; 12,8833	– 41,92; – 25,35; – 12,98; 37,14	– 2,980991
Обратная мульти-квадратичная функция	3,4554; 12,8726	– 49,69; – 32,17; – 4,81; 41,89	– 2,980883

Известное решение примера 3.9 представлено в [10]: значения координат $(x_1(1,6), x_2(1,6)) = (3,46114; 12,884)$, значение функционала $I = -2,98086$.

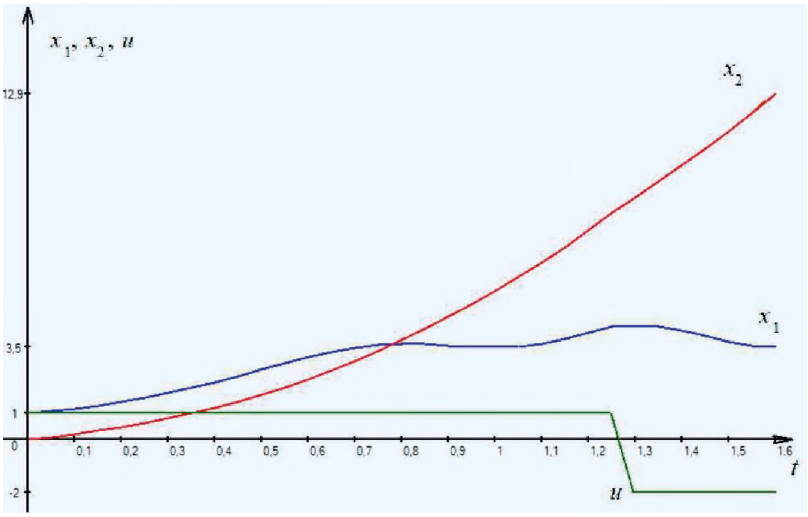


Рис. 3.15. Оптимальные управление и траектория в примере 3.9

Исходя из полученных результатов, описанный подход поиска оптимального управления на основе радиально-базисных функций с использованием мультиагентных алгоритмов показал свою эффективность при решении модельных примеров. Применение функции Гаусса, мульти-квадратичной функции, обратной квадратичной функции и обратной мульти-квадратичной функции позволяет найти программное управление, близкое к оптимальному. Найдены искомые траектории и управление, количество коэффициентов в разложении закона управления и значения этих коэффициентов, а также подобраны параметры гибридного мультиагентного алгоритма интерполяционного поиска.

При решении примеров 3.7 и 3.9 было достигнуто практически оптимальное значение функционала качества управления. В примере 3.8 значение функционала оказалось хуже, чем в известном решении [61]. Такой же результат получен при решении этого примера с помощью разложения и по системе косинусоид (разд. 3.3), и по базисной системе финитных функций (разд. 3.4). Наибольшая величина относительного отклонения по величине функционала составила 6%. Для примера 3.8 удалось вычислить значение функционала, близкое к оптимальному, только используя систему кусочно-постоянных функций с вычисляемым числом переключений (разд. 3.3). Это может быть связано с тем, что наилучшее количество переключений в примере 3.8 равно четырем, в то время как в примерах 3.7 и 3.9 оно равно единице. Заметим, что для получения решения хорошего качества, достаточно использовать сравнительно небольшое число членов разложения. При этом вычислительные затраты следует считать соответствующими сложности решаемых примеров.

3.6. ПОИСК ОПТИМАЛЬНОГО ПРОГРАММНОГО УПРАВЛЕНИЯ ПУЧКАМИ ТРАЕКТОРИЙ НА ОСНОВЕ ПСЕВДОСПЕКТРАЛЬНОГО МЕТОДА

3.6.1. Постановка задачи

Поведение нелинейной непрерывной детерминированной модели объекта управления описывается уравнением

$$\dot{x}(t) = f(t, x(t), u(t)), \tag{3.65}$$

где x – вектор состояния системы, $x \in R^n$; t – непрерывное время, $t \in T = [t_0, t_f]$, моменты t_0 начала процесса и t_f окончания процесса управления считаются заданными; u – вектор управления, $u = \left(u_1, ..., u_q\right)^T \in U \subseteq R^q$; U – множество допустимых значений управления, представляющее собой прямое произведение отрезков $[a_j, b_j]$, $j = 1, ..., q$; $f(t, x, u) = (f_1(t, x, u), ..., f_n(t, x, u))^T$ – непрерывная вектор-функция.

Начальные условия заданы компактным множеством Ω положительной меры с кусочно-гладкой границей:

$$x(t_0) = x_0 \in \Omega \subset R^n, \tag{3.66}$$

где множество Ω характеризует неопределенность задания начальных условий.

Предполагается, что при управлении используется информация только о текущем времени t, т.е. информация о векторе состояния отсутствует, а система управления является разомкнутой по состоянию (применяется программное управление).

Множество допустимых управлений $\mathcal{U}_0$ образуют такие кусочно-непрерывные функции $u(\cdot)$, что $\forall t \in T \;\; u(t) \in U$, а функция $f(t, x, u)$ такова, что решение уравнения (3.65) с начальным условием (3.66) существует и единственно.

Множество допустимых процессов $\mathcal{D}(t_0, x_0)$ – множество пар $d = (x(\cdot), u(\cdot))$, включающих траекторию $x(\cdot)$ и допустимое управление $u(\cdot)$, удовлетворяющих уравнению состояния (3.65) и начальному условию (3.66).

На множестве $\mathcal{D}(t_0, x_0)$ определен функционал качества управления отдельной траекторией:

$$I(x_0, d) = \int_{t_0}^{t_f} f^0(t, x(t), u(t))\, dt + F(x(t_f)), \tag{3.67}$$

где $f^0(t, x, u)$, $F(x)$ – заданные непрерывные функции.

Каждому допустимому управлению $u(\cdot) \in \mathcal{U}_0$ и множеству Ω поставим в соответствие пучок (ансамбль) траекторий уравнения (3.65) [24]:

$$X(t, u(t)) = \bigcup \{x(t, u(t)), x(t_0)) \mid x(t_0) \in \Omega\}, \; t \in T, \tag{3.68}$$

т.е. объединение решений уравнения (3.65) по всем возможным начальным состояниям (3.66). Пучок траекторий порождается множеством Ω и управлением $u(\cdot) \in \mathcal{U}_0$.

Качество управления пучком траекторий предлагается оценивать величиной функционала

$$J[u(\cdot)] = \int_{\Omega} I(x_0, d)dx_0 / \text{mes}\ \Omega \tag{3.69}$$

или

$$J[u(\cdot)] = \max_{x_0 \in \Omega} I(x_0, d), \tag{3.70}$$

где $\text{mes}\ \Omega$ – мера множества Ω.

Требуется найти управление $u^*(\cdot) \in \mathcal{U}_0$, минимизирующее величину функционала (3.69) или (3.70):

$$J[u^*(\cdot)] = \min_{u(\cdot) \in \mathcal{U}_0} J[u(\cdot)]. \tag{3.71}$$

Искомое управление называется оптимальным в среднем, когда минимизируется значение функционала (3.69), т.е. среднее значение функционала (3.67) на множестве начальных состояний Ω, или гарантирующим (минимаксным), когда минимизируется значение функционала (3.70).

В данном разделе предлагается прямой подход к решению поставленной задачи с параметризацией закона управления. Для этого используется псевдоспектральный метод [70, 90, 91], в котором управление аппроксимируется глобальным многочленом, а коллокация выполняется в специально выбранных точках (точках коллокации). В этом методе фиксируется набор точек коллокации, а степень используемого многочлена может изменяться. В качестве базисных функций обычно рассматриваются полиномы Чебышева или Лагранжа, а управление представляется в виде ряда по базисным функциям с неизвестными коэффициентами. В силу особых свойств полиномов Чебышева [84] в качестве ограничений на коэффициенты используются ограничения, наложенные на управление. Решение представляется в виде значений в $(N+1)$–й точках коллокации, которые однозначно определяют полином, аппроксимирующий решение. Наличие естественных ограничений на коэффициенты относится к преимуществам псевдоспектрального метода по сравнению со спектральным [50, 51], при применении которого никакой априорной информации о коэффициентах разложения, как правило, не имеется.

Для решения задачи параметрической оптимизации предлагается использовать два мультиагентных метода: гибридный мультиагентный алгоритм интерполяционного поиска (разд. 1.3) и метод, основанный на использовании линейных регуляторов управления движением агентов (разд. 1.4).

На базе этих мультиагентных алгоритмов и идей псевдоспектрального метода сформирован алгоритм поиска оптимального программного управления пучком траекторий в условиях неопределенности задания начального состояния. Приведены решения двух модельных примеров. Проведено сравнение эффективности применяемых мультиагентных алгоритмов оптимизации.

3.6.2. Алгоритм решения задачи

Для решения поставленной задачи определим способ вычисления критерия оптимальности управления пучком траекторий, а также способ параметризации закона управления.

ПРИБЛИЖЕННОЕ ВЫЧИСЛЕНИЕ ВЕЛИЧИНЫ ФУНКЦИОНАЛА

Пусть множество возможных начальных состояний Ω представляет собой параллелепипед, определенный прямым произведением отрезков $[\alpha_i, \beta_i]$, $i = 1,...,n$, т.е. $\Omega = [\alpha_1, \beta_1] \times ... \times [\alpha_n, \beta_n]$. Все отрезки $[\alpha_i, \beta_i], i = 1,...,n$, с помощью шага Δx_i разбиваются на S_i отрезков, а параллелепипед Ω делится на $S = S_1 \cdots S_n$ элементарных подмножеств Ω_k, $k = 1,...,S$. В каждом элементарном подмножестве Ω_k задается начальное состояние x_0^k – центр параллелепипеда Ω_k.

Для каждого начального состояния $x_0^k, k = 1,...,S$, из множества начальных состояний Ω следует проинтегрировать систему дифференциальных уравнений (3.65) с управлением $u(\cdot)$ одним из численных методов, например методом Рунге–Кутты 4-го порядка. В результате получаются пары $d^k, k = 1,...,S$, образованные управлением $u(\cdot)$ и соответствующими траекториями $x^k(\cdot)$. Следовательно, можно найти приближенное значение функционалов (3.69) или (3.70) соответственно:

$$J[u(\cdot)] \cong \frac{1}{S} \sum_{k=1}^{S} I\left(x_0^k, d^k\right) \tag{3.72}$$

или

$$J[u(\cdot)] \cong \max_{x_0^k, k=1,...,S} I\left(x_0^k, d^k\right). \tag{3.73}$$

ПАРАМЕТРИЗАЦИЯ ЗАКОНА УПРАВЛЕНИЯ

Процедура поиска решения задачи (3.71) сводится к проблеме поиска наилучших значений параметров, которые задают структуру управления. Управление ищется в параметрическом виде, который определяется числом коэффициентов в разложении управления по системе базисных функций и их значениями. В качестве базисных функций предлагается использовать полиномы Чебышева. Применение разложений по многочленам Чебышева широко используется в псевдоспектральных методах [70, 90, 91].

Закон управления предлагается искать в виде функции насыщения sat, гарантирующей выполнение параллелепипедных ограничений на управление:

$$u_j(t) = \text{sat}\left\{g_j(t)\right\}, \; j = 1,...,q, \;\; t \in T, \tag{3.74}$$

где

$$\text{sat}\, g_j(t) = \begin{cases} g_j(t), & a_j \le g_j(t) \le b_j, \\ a_j, & g_j(t) < a_j, \\ b_j, & g_j(t) > b_j, \end{cases} \tag{3.75}$$

где аргументы $g_j(t)$ функции насыщения предлагается искать в виде линейной комбинации базисных функций.

Используем систему многочленов Чебышева первого рода $T_m(x)$, ортогональных с весом $\rho(x) = 1/\sqrt{1-x^2}$ на отрезке $[-1; 1]$:

$$T_0(x) = 1, \; T_1(x) = x, \ldots, T_{m+1}(x) = 2xT_m(x) - T_{m-1}(x), \; m \ge 1. \tag{3.76}$$

При этом справедливы следующие свойства:

$$\int_{-1}^{1} \frac{T_m(x)T_n(x)}{\sqrt{1-x^2}}\,dx = \begin{cases} 0, & m \neq n, \\ \pi, & m = n = 0, \\ \pi/2, & m = n \neq 0. \end{cases}$$

Тогда аргументы функции насыщения находятся по формуле

$$g_j(t) = \sum_{i=0}^{N^j} u_i^j T_i(t)\,, \tag{3.77}$$

где u_i^j – неизвестные коэффициенты; N^j – масштаб усечения.

Решение ищется в форме расширенной матрицы-столбца $U_N = (U_{N^1}^1, \ldots, U_{N^q}^q)^T$ вида

$$U_{N^j}^j = \left[N^j \;\middle|\; u_0^j, u_1^j, \ldots, u_{N^j}^j \right]^T, \tag{3.78}$$

с ограничениями

$$0 \le N^j \le N_{\max}, \quad N^j - \text{ целые}; \quad j = 1, \ldots, q\,, \tag{3.79}$$

$$a_j \le u_i^j \le b_j,$$

где значение $N_{\max}$ задается исходя из возможных требований к точности решения и лимиту вычислительных затрат, a_j и b_j – параметры ограничений на управление, известные из постановки задачи.

Таким образом, ищется решение на множестве векторов, полученных путем конкатенации векторов, состоящих из коэффициентов разложения для управлений $u_1(\cdot), \ldots, u_q(\cdot)$ и масштабов усечения.

Так как полиномы Чебышева рассматриваются на отрезке $[-1;1]$, то требуется преобразовать отрезок $[t_0, t_f]$ к стандартному отрезку $[-1;1]$, применив линейное преобразование:

$$t = \frac{t_f - t_0}{2}\tau + \frac{t_f + t_0}{2}, \quad dt = \frac{t_f - t_0}{2}d\tau, \quad \tau \in [-1;1].$$

В результате уравнение (3.65) и функционал (3.67) перепишем в следующей форме:

$$\dot{x}(\tau) = \frac{t_f - t_0}{2} f(\tau, x(\tau), u(\tau)), \tag{3.80}$$

$$I(x_0, d) = \frac{t_f - t_0}{2} \int_{-1}^{1} f^0(\tau, x(\tau), u(\tau))d\tau + F(x(1)).$$

Используя многочлены Чебышева, в качестве узлов интерполяции выберем положения экстремумов многочлена Чебышева степени N^j и крайние точки отрезка интерполяции: $t_k = \cos(\pi k / N^j), k = 0, \ldots, N^j$. На отрезке $[-1;1]$ они располагаются следующим образом: $1 = t_0 > t_1 > \ldots > t_{N^j-1} > t_{N^j} = -1$ и называются точками коллокации Чебышева–Гаусса–Лобатто (CGL (Chebyshev–Gauss–Lobatto)).

Поскольку многочлены Чебышева можно записать в форме $T_m(t) = \cos(m \arccos t),\ m = 0,...,N^j$, то их значения в узлах найдем в виде

$$T_m(t_k) = \cos(\pi k m / N^j),\ k = 0,...,N^j;\ j = 1,...,q,$$

В качестве базисной системы $\{\varphi_i(t)\}_{i=0}^{N^j}$ предлагается использовать [70, 90, 91]:

$$\varphi_i(t) = \frac{(-1)^{i+1}(1-t^2)T'_{N^j}(t)}{(N^j)^2 c_i (t - t_i)}, i = 0,...,N^j;\ \ c_i = \begin{cases} 2, & i = 0,\ i = N^j, \\ 1, & i = 1,...,N^j - 1. \end{cases}$$

Графики функций базисной системы при $N^j = 8$ изображены на рис. 3.16.

Представление закона управления (3.74) запишем в виде

$$u_j^{N^j}(t) = \text{sat}\left[\sum_{i=0}^{N^j} u_i^j \varphi_i(t)\right], j = 1,...,q,\ \ t \in T, \tag{3.81}$$

где u_i^j – неизвестные величины. При этом в узлах интерполяции выполняются соотношения $\varphi_i(t_k) = \delta_{ik},\ u_j^{N^j}(t_k) = u_k^j$, где δ_{ik} – символ Кронекера. На коэффициенты разложения накладываются ограничения, следующие из постановки задачи: $a_j \le u_k^j \le b_j, k = 0,...,N^j; j = 1,...,q.$

В задаче требуется найти координаты расширенного вектора

$$(N^1,...,N^q \,|\, u_0^1,...,u_{N^1}^1,...,u_0^q,...,u_{N^q}^q)^T, \tag{3.82}$$

содержащего блок целочисленных переменных $N = (N^1,...,N^q)^T$ и блок действительных переменных $C = (u_0^1,...,u_{N^1}^1,...,u_0^q,...,u_{N^q}^q)^T$, на которые накладываются ограничения вида (3.79).

Таким образом, предлагается искать значения координат блочных расширенных векторов (3.82), содержащих блок целочисленных переменных и блок действительных переменных, на которые наложены интервальные ограничения.

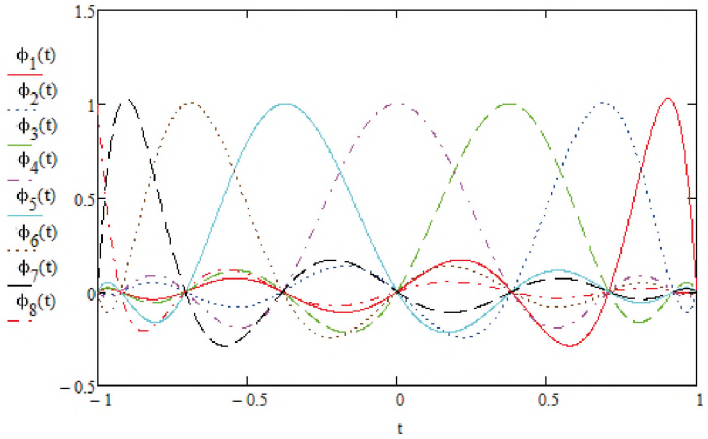


Рис. 3.16. Графики системы базисных функций для многочлена Чебышева 8-й степени

З а м е ч а н и е. Если требуется решить более общую задачу поиска оптимального управления пучком траекторий с неполной обратной связью $\boldsymbol{u}(t,x^1)$, рассмотренную в разд. 3.1, то закон управления ищется в виде функции насыщения sat, гарантирующей выполнение параллелепипедных ограничений на управление:

$$\boldsymbol{u}_j(t,x^1)=\operatorname{sat}\left\{g_j\left(t,x_1,...,x_m\right)\right\},\ j=1,...,q,$$

где

$$\operatorname{sat}v_j(t)=\begin{cases}v_j(t), & a_j\le v_j(t)\le b_j,\\ a_j, & v_j(t)<a_j,\\ b_j, & v_j(t)>b_j.\end{cases}$$

Здесь $v_j(t)=g_j\left(t,x_1(t),...,x_m(t)\right)$, а аргументы $g_j\left(t,x_1,...,x_m\right)$ функции насыщения предлагается искать в виде линейной комбинации базисных функций. Используется система многочленов Чебышева первого рода $T_m(x)$, определенная соотношениями (3.76).

Тогда аргументы функции насыщения находятся по формуле

$$g_j\left(t,x_1,...,x_m\right)=\sum_{i_0=0}^{N_0^j}\sum_{i_1=0}^{N_1^j}\cdots\sum_{i_m=0}^{N_m^j}u^j_{i_0i_1\ldots i_m}T_{i_0}\left(t\right)T_{i_1}\left(x_1\right)\cdots T_{i_m}\left(x_m\right),$$

где $u^j_{i_0i_1\ldots i_m}$ – неизвестные коэффициенты; $N_0^j,N_1^j,...,N_m^j$ – масштабы усечения по времени и координатам вектора состояния, используемым в управлении. Решение предлагается искать в форме расширенной матрицы-столбца $U_N=(U_N^1,.....,U_N^q)^T$ вида

$$U_N^j=\left[\begin{bmatrix}N_0^j\\N_1^j\\\vdots\\N_m^j\end{bmatrix}\left\|\begin{bmatrix}u^j_{00\ldots0}\\u^j_{00\ldots1}\\\vdots\\u^j_{00\ldots N_m^j}\end{bmatrix}\begin{bmatrix}u^j_{10\ldots0}\\u^j_{10\ldots1}\\\vdots\\u^j_{10\ldots N_m^j}\end{bmatrix}\right.\cdots\begin{bmatrix}u^j_{N_0^j0\ldots0}\\u^j_{N_0^j0\ldots1}\\\vdots\\u^j_{N_0^j0\ldots N_m^j}\end{bmatrix}\cdots\left.\begin{bmatrix}u^j_{N_0^jN_1^j\ldots0}\\u^j_{N_0^jN_1^j\ldots1}\\\vdots\\u^j_{N_0^jN_1^j\ldots N_m^j}\end{bmatrix}\right]^T$$

с ограничениями

$$0\le N_i^j\le N_{\max},\ \ N_i^j-\text{ целые};\ j=1,...,q;\ i=0,1,...,m,$$

$$a_j\le u^j_{i_0,i_1,...,i_m}\le b_j,$$

где значение $N_{\max}$ задается исходя из возможных требований к точности решения и лимиту вычислительных затрат, a_j и b_j – параметры ограничений на управление, известные из постановки задачи.

Предполагается, что известна оценка множества возможных состояний, которая представляется прямым произведением $[\underline{x}_1,\overline{x}_1]\times\cdots\times[\underline{x}_n,\overline{x}_n]$, где $\underline{x}_i,\overline{x}_i$ – нижняя и верхняя границы по каждой координате соответственно, определяемые физическим смыслом решаемой задачи. Так как полиномы Чебышева рассматриваются на отрезке $[-1;1]$, то требуется преобразовать отрезки $[t_0,t_f],[\underline{x}_1,\overline{x}_1],...,[\underline{x}_n,\overline{x}_n]$ к стандартному отрезку $[-1;1]$, применив линейные преобразования:

$$t = \frac{t_f - t_0}{2}\tau + \frac{t_f + t_0}{2}, \quad dt = \frac{t_f - t_0}{2}d\tau, \quad \tau \in [-1;1],$$

$$x_i = \frac{\underline{x}_i + \overline{x}_i}{2} + \frac{\overline{x}_i - \underline{x}_i}{2}\tilde{x}_i; \quad \tilde{x}_i \in [-1;1], \quad i = 1,...,n.$$

Таким образом, ищется решение на множестве векторов, полученных путем конкатенации векторов, состоящих из коэффициентов разложения координат управления $\boldsymbol{u}(t,x^1) = (\boldsymbol{u}_1(t,x^1),\ldots,\boldsymbol{u}_q(t,x^1))^T$ и векторов масштабов усечения по времени и координатам вектора состояния.

Рассмотрим процедуру нахождения оптимального программного управления пучком траекторий. Описанный способ параметризации управления (3.81) приводит к необходимости решения смешанной целочисленно-непрерывной задачи условной оптимизации:

$$f(N^*,C^*) = \min_{N\in N,\, C\in C} f(N,C), \tag{3.83}$$

где $f(N,C)$ – заданная непрерывная целевая функция, величина которой находится по формулам (3.72) или (3.73) в зависимости от поставленной задачи, а множества допустимых решений имеют вид

$$\boldsymbol{N} = \{N = (N^1,...,N^q) \,|\, N^j \in \{0,...,N_{\max}\}, j = 1,...,q\},$$

$$\boldsymbol{C} = \{C = (u_0^1,....,u_{N^1}^1,...,u_0^q,....,u_{N^q}^q)^T \,|\, u_i^j \in [a_j,b_j],\ a_j \le b_j,\ j = 1,...,q,\ i = 0,...,N^j\}.$$

Для ее решения используются гибридные метаэвристические алгоритмы, относящиеся к группе мультиагентных. Поскольку вектор подбираемых параметров содержит как целочисленные, так и непрерывные координаты, то формируется алгоритм, реализующий два вида поиска: процедуру целочисленного линейного поиска для целочисленных переменных и процедуру поиска по непрерывным переменным, основанную на мультиагентных алгоритмах условной оптимизации.

В связи с этим, предлагается последовательная схема поиска оптимального программного управления, включающая описанные ниже шаги.

Шаг 1. *Приведение исходной постановки задачи к стандартному виду.* Отрезок $[t_0,t_f]$ привести к стандартному отрезку $[-1;1]$ по формуле

$$t = \frac{t_f - t_0}{2}\tau + \frac{t_f + t_0}{2}, \quad dt = \frac{t_f - t_0}{2}d\tau, \quad \tau \in [-1;1].$$

Тогда модель объекта управления будет иметь вид

$$\dot{x}(t) = \frac{\tau_f - \tau_0}{2} f(t,x(t),u(t)), \quad x(-1) = x_0,$$

$$I(x_0,d) = \frac{\tau_f - \tau_0}{2}\int_{-1}^{1} f^0(t,x(t),u(t))dt + F(x(1),\tau_f).$$

Задать параметры выбранного мультиагентного алгоритма, максимальную степень многочлена Чебышева $N_{\max}$, максимальное число итераций $ITER$ и максимальное число проходов P. Положить $it = 0$, $p = 0$ (счетчики числа итераций и проходов).

Шаг 2. *Генерирование начального приближения.*

Для целочисленных переменных:

$$N^j = \mathrm{INT}\left[\mathrm{rand}[0;1]\ (N_{\max}^j + 1)\right], \quad 0 \le N^j \le N_{\max}, \quad j = 1,...,q,$$

где $\mathrm{rand}[0;1]$ – число, генерируемое согласно равномерному закону распределения, $\mathrm{INT}[.]$ – операция выделения целой части числа.

Для непрерывных переменных требуется сгенерировать начальную популяцию, состоящую из NP агентов, координаты u_i^j которых определены на отрезках $[a_j, b_j]$:

$$u_i^j = a_j + \mathrm{rand}[0;1](b_j - a_j),\ j = 1,...,q;\ i = 0,...,N^j.$$

Шаг 3. *Реализация процедуры поиска на множестве* N *по целочисленным переменным при известных значениях координат вектора* C. Применяется процедура целочисленного линейного поиска, при этом значения координат вектора C в блочном векторе (N, C) считаются фиксированными.

Шаг 3.1. Упорядочить координаты вектора N случайным образом. Положить $New = false$ – индикатор наличия улучшения, $k = 1$.

Шаг 3.2. Пусть на k-м месте стоит i-я переменная. "Просканировать" множество решений вида $Z = N + ke_i$, $k = 0, \pm 1, \pm 2, ...$, где $0 \le Z \le N^{\max}$, e_i – единичный орт размеров $(q \times 1)$, составленный из всех нулей и одной единицы на i-м месте; $N^{\max}$ – вектор размеров $(q \times 1)$, все координаты которого равны $N_{\max}$, знак неравенства понимается покоординатно.

Шаг 3.3. Если среди полученных в результате сканирования решений имеется наилучшее, удовлетворяющее условию $f(Z, C) < f(N, C)$, то заменить N на Z и положить $New = true$.

Шаг 3.4. Если $k < q$, то положить $k = k + 1$ и перейти к шагу 3.2. Если $k = q$ и если $New = true$, положить $New = false$, $j = 1$ и перейти к шагу 3.2. Иначе поиск завершить и положить $p = p + 1$ (счетчик числа проходов).

Шаг 4. *Реализация процедуры поиска на множестве* C *по непрерывным переменным при фиксированных координатах вектора* N.

Шаг 4.1. Разбить промежуток времени $[t_0, t_f]$: $t_0 = 1, t_1, ..., t_{N-1}, t_{N^j} = -1$,

где $t_k = \cos\dfrac{\pi k}{N^j}$, $k = 0, 1, \ldots, N^j$; N^j – количество промежуточных точек отрезка (известно из шага 3).

Шаг 4.2. По сгенерированным коэффициентам (см. шаг 2) сформировать управление в виде функции насыщения sat, гарантирующей выполнение ограничений на управление:

$$u_j^{N^j}(t) = \mathrm{sat}\left\{g_j(t)\right\},\ j = 1,...,q,\ t \in T,$$

$$g_j(t) = \sum_{i=0}^{N^j} u_i^j \varphi_i(t),\ u_j^{N^j}(t_k) = u_k^j.$$

В качестве базисной системы функций $\{\varphi_i(t)\}_{i=0}^{N^j}$ рассматриваются многочлены Чебышева. Базисная система имеет следующие свойства:

– внутри интервала:

$$\varphi_i(t) = \frac{(-1)^{i+1}(1-t^2)T'_{N^j}(t)}{(N^j)^2 c_i (t - t_i)}, i = 0, 1, ..., N^j;\ c_i = \begin{cases} 2, & i = 0, i = N^j, \\ 1, & i = 1, ..., N^j - 1, \end{cases}$$

где $T_0(x) = 1$, $T_1(x) = x$, $T_{m+1}(x) = 2xT_m(x) - T_{m-1}(x)$, $m \ge 1$;

– в узлах:

$$\varphi_i(t_k) = \delta_{ik} = \begin{cases} 1, & i = k, \\ 0, & i \neq k. \end{cases}$$

Шаг 4.3. Проинтегрировать NP систем дифференциальных уравнений (см. шаг 1) с управлениями $u^1(\cdot),\ldots,u^{NP}(\cdot)$, например, методом Рунге–Кутты 4-го порядка для каждого начального состояния x_0^k из множества Ω. Для каждого агента и для каждого начального состояния получить соответствующие траектории $(x_1^{1,x_0^k}(\cdot),\ldots,x_n^{1,x_0^k}(\cdot)),\ldots,(x_1^{NP,x_0^k}(\cdot),\ldots,x_n^{NP,x_0^k}(\cdot))$ и вычислить значения функционала $I^{1,x_0^k},\ldots,I^{NP,x_0^k}$. Найти значения критерия качества управления пучком по формуле (3.72), соответствующее оптимальному в среднем управлению (или значение, соответствующее гарантирующему управлению, по формуле (3.73)): $J^1,\ldots,J^{NP}$.

Шаг 4.4. Выполнить очередную итерацию выбранного метода минимизации функционала (3.72) (или (3.73)). Получить новые положения агентов $1',\ldots,NP'$ (векторы значений коэффициентов разложения по системе базисных функций).

Шаг 4.5. Проверка критериев окончания поиска по непрерывным переменным.

Если $it < ITER$, то положить $it = it + 1$ и перейти к шагу 4.2.

Если $it \geq ITER$, завершить процедуру поиска значений непрерывных переменных и перейти к шагу 5.

Шаг 5. *Завершение поиска.* Если $p < P$, перейти к шагу 3. Если $p = P$, поиск завершить. Выбрать в качестве приближенного решения наилучшего агента в популяции, т.е. вектор (N, C) с найденными коэффициентами $u_i^j, i = 0,...,N^j$ и значениями координат вектора $N = (N^1,...,N^q)$, соответствующие ему управление и пучок траекторий, а также значение J функционала (3.72) (или (3.73)).

3.6.3. Модельные примеры

Пример 3.10. Исходная постановка задачи [61] приведена в табл. 3.23.

Таблица 3.23. Постановка задачи (пример 3.10)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = x_2(t) + \sin x_1(t) + u(t) \\ \dot{x}_2 = x_1(t)\cos x_2(t)u(t) \end{cases}$
Временной интервал	$t \in [0;1]$
Ограничения на управление	$-1 \leq u \leq 1$
Функционал (3.67)	$I(x_0, d) = x_2(1)$

Поставленная задача приводится к виду, удобному для применения описанного псевдоспектрального алгоритма с использованием полиномов Чебышева. Формула преобразования времени ($t_0 = 0,\ t_f = 1$):

$$t = \frac{t_f - t_0}{2}\tau + \frac{t_f + t_0}{2} = \frac{1}{2}\tau + \frac{1}{2}, \quad dt = \frac{t_f - t_0}{2}d\tau = \frac{1}{2}d\tau,\ \tau \in [-1;1].$$

Тогда отрезок времени будет иметь вид $t \in [-1;1]$, а система дифференциальных уравнений перепишется в форме

$$\begin{cases} \dot{x}_1 = \dfrac{1}{2}\left(x_2(t) + \sin x_1(t) + u(t)\right), \\ \dot{x}_2 = \dfrac{1}{2}\left(x_1(t)\cos x_2(t)u(t)\right). \end{cases}$$

Выбраны следующие значения параметров гибридного алгоритма интерполяционного поиска: $NP = 30$, $I_{\max} = 100$, $M_1 = 5$, $M_2 = 2$, $PRT = 0{,}9$, $nstep = 10$, $b_2 = 20$, и метода, основанного на применении линейных регуляторов управления движением агентов: $NP = 13$, $NMAX = 100$, $P_{\max} = 20$, $k_s = 100$, $k_l = 5$, $h = 0{,}0001$. Множество начальных состояний $\Omega = [-0{,}05; 0{,}05] \times [-0{,}05; 0{,}05]$, $S_1 = S_2 = 5$. На коэффициенты в разложении (3.81) наложены ограничения вида $u_i^1 \in [-1{,}2; 1{,}2]$ поскольку $j = 1$.

В табл. 3.24 приведены результаты численного решения задачи минимизации критерия (3.72), т.е. поиска оптимального в среднем программного управления, а в табл. 3.25 – задачи минимизации критерия (3.73), т.е. поиска гарантирующего управления. В алгоритме применяется целочисленный линейный поиск и два мультиагентных методов оптимизации.

Таблица 3.24. Результаты решения примера 3.10 (оптимальное в среднем управление)

Результаты	Гибридный мультиагентный алгоритм интерполяционного поиска	Мультиагентный алгоритм, основанный на применении линейных регуляторов управления движением агентов
Количество узлов	$N = 10$	$N = 12$
Отрезки значений координат $x_1(1), x_2(1)$	[0,2596; 0,6102], [– 0,1761; – 0,0911]	[0,2603; 0,6109], [– 0,1767; – 0,0917]
Значения коэффициентов в разложении (3.81)	– 1,2; – 1,2; – 1,2; – 1,2; – 1,2; 1,2; 1,2; 1,2; 1,2; 1,2	– 1,2; – 1,2; – 1,2; – 1,2; – 1,2; – 1,2; 1,2; 1,2; 1,2; 1,2; 1,2; 1,2
Значение функционала (3.72)	– 0,133572	– 0,134232
Затраченное время	106 секунд	152 секунд

Таблица 3.25. Результаты решения примера 3.10 (гарантирующее управление)

Результаты	Гибридный мультиагентный алгоритм интерполяционного поиска	Мультиагентный алгоритм, основанный на применении линейных регуляторов управления движением агентов
Количество узлов	$N = 10$	$N = 12$
Отрезки значений координат $x_1(1), x_2(1)$	[0,2919; 0,6426], [– 0,1748; – 0,0915]	[0,2603; 0,6109], [– 0,17617 – 0,0917]
Значения коэффициентов в разложении (3.81)	– 1,2; – 1,2; – 1,2; – 1,2; – 0,993; 1,2; 1,2; 1,2; 1,2; 1,2	– 1,2; – 1,2; – 1,2; – 1,2; – 1,2; –1,2; 1,2; 1,2; 1,2; 1,2; 1,2; 1,2; 1,2
Значение функционала (3.73)	– 0,091459	– 0,091749
Затраченное время	120 секунд	285 секунд

Исходя из полученных результатов, можно сделать вывод о том, что оба алгоритма успешно справились с поставленной задачей, но гибридный мультиагентный алгоритм интерполяционного поиска уступает мультиагентному алгоритму, основанному на применении линейных регуляторов управления движением агентов, в смысле полученного значения функционала, однако при этом тратится больше ресурсов на вычисления.

На рис. 3.17 изображены найденные оптимальное в среднем управление и соответствующий ему пучок траекторий для случая поиска решения мультиагентным алгоритмом, основанным на применении линейных регуляторов управления движением агентов.

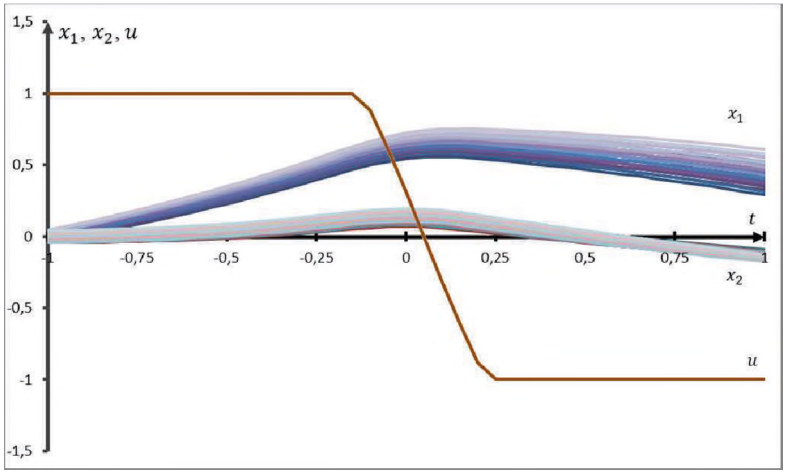


Рис. 3.17. Графики управления и пучка траекторий в примере 3.10

Результаты, приведенные в табл. 3.24 и 3.25, сравнивались с решением [61], полученным при известном фиксированном начальном состоянии, принадлежащем множеству Ω: координаты вектора состояния в конце промежутка времени функционирования системы $x_1(1) = 0,440804$; $x_2(1) = -0,13593$ принадлежат соответствующим отрезкам, указанным в третьих строках таблиц, значение $I = -0,13593$ функционала (3.67) принадлежит множеству значений функционала для траекторий, образующих пучок.

Пример 3.11. Исходная постановка задачи [10] приведена в табл. 3.26.

Таблица 3.26. Постановка задачи (пример 3.11)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = \dfrac{1}{\cos x_1(t) + 2} + 3\sin x_2(t) + u(t) \\ \dot{x}_2 = x_1(t) + x_2(t) + u(t) \end{cases}$
Временной интервал	$t \in [0; 1,6]$
Ограничения на управление	$-2 \le u \le 1$
Функционал (3.67)	$I(x_0, d) = -x_1(1,6) + \dfrac{1}{2} x_2(1,6)$

Поставленная задача приводится к виду, удобному для применения описанного псевдоспектрального алгоритма с использованием полиномов Чебышева. Формула преобразования времени ($t_0 = 0,\ t_f = 1,6$):

$$t = \frac{t_f - t_0}{2}\tau + \frac{t_f + t_0}{2} = \frac{4}{5}\tau + \frac{4}{5},$$

$$dt = \frac{t_f - t_0}{2}d\tau = \frac{4}{5}d\tau,\ \ \tau \in [-1;1].$$

Тогда будет справедливо, что $t \in [-1;1]$, а система дифференциальных уравнений перепишется в виде

$$\begin{cases} \dot{x}_1 = \dfrac{4}{5}\left(\dfrac{1}{\cos x_1(t) + 2} + 3\sin x_2(t) + u(t) \right), \\ \dot{x}_2 = \dfrac{4}{5}\left(x_1(t) + x_2(t) + u(t) \right). \end{cases}$$

Выбраны следующие значения параметров гибридного алгоритма интерполяционного поиска: $NP = 30,\ I_{\max} = 100,\ M_1 = 5,\ M_2 = 2,\ PRT = 0,9$, $nstep = 10,\ b_2 = 20,$ и метода, основанного на применении линейных регуляторов управления движением агентов: $NP = 13$, $NMAX = 100,\ P_{\max} = 10,\ k_s = 100,\ k_l = 5,\ h = 0,0001$. Множество начальных состояний $\Omega = [0,95;1,05] \times [-0,05;0,05]$, $S_1 = S_2 = 5$. На коэффициенты в разложении (3.81) наложены ограничения вида $u_i^1 \in [-2,2;\ 1,1]$ поскольку $j = 1$.

Результаты решения задачи при поиске оптимального в среднем и гарантирующего управлений представлены в табл. 3.27 и 3.28 соответственно. Указано найденное наилучшее количество точек коллокации и значения коэффициентов в представлении законов управления. Вычисления реализованы с использованием двух выбранных мультиагентных алгоритмов.

Таблица 3.27. Результаты решения примера 3.11 (оптимальное в среднем управление)

Результаты	Гибридный мультиагентный алгоритм интерполяционного поиска	Мультиагентный алгоритм, основанный на применении линейных регуляторов управления движением агентов
Количество узлов	$N = 13$	$N = 12$
Отрезки значений координат $x_1(1,6), x_2(1,6)$	[3,4351; 3,5566], [12,4475; 13,3215]	[3,3571; 3,4392], [12,2323; 13,1006]
Значения коэффициентов в разложении (3.81)	– 2,2; – 2,2; – 2,2; – 2,2; 0,49; 1,1; 1,1; 1,1; 1,1; 1,1; 1,1; 1,1; 1,1; 1,1	– 2,2; – 2,2; – 2,2; – 2,2; 1,1; 1,1; 1,1; 1,1; 1,1; 1,1; 1,1; 1,1
Значение функционала (3.72)	– 2,961588	– 2,946605
Затраченное время	564 секунд	187 секунд

Таблица 3.28. Результаты решения примера 3.11 (гарантирующее управление)

Результаты	Гибридный мультиагентный алгоритм интерполяционного поиска	Мультиагентный алгоритм, основанный на применении линейных регуляторов управления движением агентов
Количество узлов	$N = 12$	$N = 10$
Отрезки значений координат $x_1(1,6), x_2(1,6)$	[3,4963; 3,6461], [12,5654; 13,4434]	[3,5253; 3,6854], [12,6057; 13,4832]
Значения коэффициентов в разложении (3.81)	–2,2; –2,2; –2,2; –1,06; 1,1; 1,1; 1,1; 1,1; 1,1; 1,1; 1,1; 1,1	–2,2; –2,2; –2,2; 1,1; 1,1; 1,1; 1,1; 1,1; 1,1; 1,1
Значение функционала (3.73)	–2,786368	–2,777532
Затраченное время	168 секунд	25 секунд

Исходя из полученных результатов, можно заключить, что при использовании гибридного мультиагентного алгоритма интерполяционного поиска достигается меньшее значение функционала по сравнению с другим мультиагентным алгоритмом, но при этом затрачивается большее время процессора. В целом оба алгоритма успешно справились с решением задачи. На рис. 3.18 приведены графики гарантирующего программного управления и соответствующего пучка траекторий.

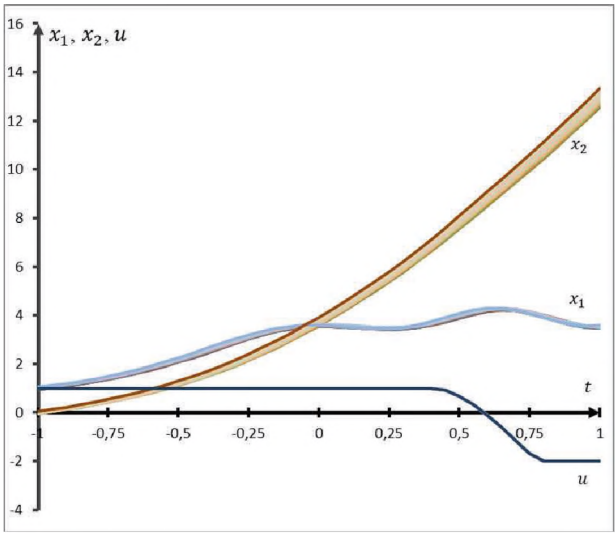


Рис. 3.18. Графики управления и пучка траекторий в примере 3.11

Результат сравнивался с решением, полученным в [10] при известном фиксированном начальном состоянии: координаты вектора состояния в конце времени функционирования системы $x_1(1,6) = 3,46114$; $x_2(1,6) = 12,884$ принадлежат соответствующим отрезкам, указанным в третьих строках таблиц 3.27 и 3.28, значение $I = -0,13593$ функционала (3.67) принадлежит множеству значений функционала для траекторий, образующих пучок.

3.7. ПОИСК ОПТИМАЛЬНОГО УПРАВЛЕНИЯ ПУЧКАМИ ТРАЕКТОРИЙ С НЕПОЛНОЙ ОБРАТНОЙ СВЯЗЬЮ

3.7.1. Постановка задачи

Рассматривается задача поиска оптимального управления пучками траекторий непрерывных детерминированных систем с неполной обратной связью. К данному классу задач относятся проблемы управления, в которых имеется неопределенность задания начальных условий в виде некоторого множества, а также предполагается, что доступна мгновенная информация только о части координат вектора состояния системы, которая не накапливается. Описан способ параметризации задачи на основе разложения по системе базисных функций, а именно системе нестационарных косинусоид. Для решения задачи параметрической оптимизации предлагается использовать два мультиагентных метода: гибридный мультиагентный алгоритм интерполяционного поиска (разд. 1.3) и метод, основанный на использовании линейных регуляторов для управления движением агентов (разд. 1.4).

Поведение нелинейной непрерывной детерминированной модели объекта управления описывается уравнением

$$\dot{x}(t) = f(t, x(t), u(t)), \tag{3.84}$$

где x – вектор состояния системы, $x \in R^n$; t – непрерывное время, $t \in T = [t_0, t_f]$, моменты t_0 начала процесса и t_f окончания процесса управления считаются заданными; u – вектор управления, $u = \left(u_1, ..., u_q\right)^T \in U(t) \subseteq R^q$; $U(t)$ – множество допустимых значений управления, представляющее собой прямое произведение отрезков $[a_j(t), b_j(t)]$, $j = 1, ..., q$; $f(t, x, u) = (f_1(t, x, u), ..., f_n(t, x, u))^T$ – непрерывная вектор-функция.

Начальные условия заданы компактным множеством Ω положительной меры с кусочно-гладкой границей:

$$x(t_0) = x_0 \in \Omega \subset R^n, \tag{3.85}$$

где множество Ω характеризует неопределенность задания начальных условий.

Предполагается, что при управлении используется информация о времени t и о части координат вектора состояния x (без ограничения общности считается, что это первые m координат). Таким образом, о компонентах вектора $x^1 \in R^m$ известна текущая информация, а о компонентах вектора $x^2 \in R^{n-m}$ она отсутствует, при этом $x = (x^1, x^2)^T \in R^n$, $0 \le m \le n$, т.е. $x^1 = (x_1, ..., x_m)^T$, $x^2 = (x_{m+1}, ..., x_n)^T$. Если $m = 0$, информация о векторе состояния отсутствует, а если $m = n$, то имеется полная информация о векторе состояния.

З а м е ч а н и е. Если заданы уравнения измерительной системы вида

$$z(t) = h(t, x(t)),$$

где $h(t, x)$ – непрерывно дифференцируемая функция по всем переменным, то с учетом (3.84) можно записать дифференциальное уравнение

$$\dot{z}(t) = \frac{\partial h(t, x(t))}{\partial t} + \sum_{i=1}^{n} \frac{\partial h(t, x(t))}{\partial x_i} f_i(t, x(t), u(t)), \; z(t_0) = z_0 = h(t_0, x_0), x_0 \in \Omega.$$

Тогда полученное уравнение вместе с уравнением (3.84) описывают динамическую систему, которую характеризует расширенный вектор состояния (z, x). В нем о координатах вектора z имеется полная информация, а информация о координатах вектора x не известна, т.е. в этом случае проблема может быть сведена к рассматриваемой.

Множество допустимых управлений $\mathcal{U}_m$ образуют такие функции $\boldsymbol{u}(t, x^1)$, что $\forall t \in T$ управление $u(t) = \boldsymbol{u}(t, x^1(t)) \in U(t)$ кусочно-непрерывно, а функция $f(t, x, \boldsymbol{u}(t, x^1))$ такова, что решение уравнения (3.84) с начальным условием (3.85) существует и единственно.

Управление, применяемое в каждый момент времени t, имеет вид управления с неполной обратной связью: $u(t) = \boldsymbol{u}(t, x^1(t))$. Если $m = 0$, система управления будет разомкнутой по состоянию, а соответствующее управление $u(t)$ – программным, а если $m = n$, то система управления будет замкнутой с полной обратной связью, определяемой управлением $\boldsymbol{u}(t, x)$.

Множество допустимых процессов $\mathcal{D}(t_0, x_0)$ – множество пар $d = (x(\cdot), u(\cdot))$, включающих траекторию $x(\cdot)$ и кусочно-непрерывное допустимое управление $u(\cdot)$, где $\forall t \in T \;\; u(t) \in U(t)$, удовлетворяющих уравнению состояния (3.84) и начальному условию (3.85).

На множестве $\mathcal{D}(t_0, x_0)$ определен функционал качества управления отдельной траекторией:

$$I(x_0, d) = \int_{t_0}^{t_f} f^0(t, x(t), u(t))\, dt + F(x(t_f)), \tag{3.86}$$

где $f^0(t, x, u)$, $F(x)$ – заданные непрерывные функции.

Каждому допустимому управлению $\boldsymbol{u}(t, x^1) \in \mathcal{U}_m$ и множеству Ω поставим в соответствие пучок (ансамбль) траекторий уравнения (3.84) [24]:

$$X(t, \boldsymbol{u}(t, x^1)) = \bigcup \{x(t, \boldsymbol{u}(t, x^1(t)), x(t_0)) \mid x(t_0) \in \Omega\}, \; t \in T, \tag{3.87}$$

т.е. объединение решений уравнения (3.84) по всем возможным начальным состояниям (3.85). Пучок траекторий порождается множеством Ω и управлением $\boldsymbol{u}(t, x^1) \in \mathcal{U}_m$.

Качество управления пучком траекторий предлагается оценивать величиной функционала

$$J[\boldsymbol{u}(t, x^1)] = \int_{\Omega} I(x_0, d) dx_0 \,/\, \text{mes}\, \Omega, \tag{3.88}$$

где $\text{mes}\, \Omega$ – мера множества Ω.

Требуется найти управление $\boldsymbol{u}^*(t, x^1) \in \mathcal{U}_m$, минимизирующее величину функционала (3.88):

$$J[\boldsymbol{u}^*(t, x^1)] = \min_{\boldsymbol{u}(t, x^1) \in \mathcal{U}_m} J[\boldsymbol{u}(t, x^1)]. \tag{3.89}$$

Искомое управление называется оптимальным в среднем, так как минимизируется среднее значение функционала (3.86) на множестве возможных начальных состояний Ω.

3.7.2. Алгоритм решения задачи

В основе алгоритма лежит переход к задаче параметрической целочисленно-непрерывной оптимизации. Рассмотрим сначала процедуру вычисления критерия (3.88) оптимальности управления пучком траекторий, а затем способ параметризации закона управления с неполной обратной связью.

ПРИБЛИЖЕННОЕ ВЫЧИСЛЕНИЕ ВЕЛИЧИНЫ ФУНКЦИОНАЛА

Предположим, что множество начальных состояний Ω представляет собой параллелепипед, определенный прямым произведением отрезков $[\alpha_i,\beta_i]$, $i=1,...,n$, т.е. $\Omega=[\alpha_1,\beta_1]\times...\times[\alpha_n,\beta_n]$. Все отрезки $[\alpha_i,\beta_i], i=1,...,n$, с помощью шага Δx_i разбиваются на N_i отрезков, а параллелепипед Ω делится на $N=N_1\cdots N_n$ элементарных подмножеств Ω_k, $k=1,...,N$. В каждом элементарном подмножестве Ω_k задается начальное состояние x_0^k – центр параллелепипеда Ω_k.

Для каждого начального состояния $x_0^k, k=1,...,N$, из множества начальных состояний Ω следует проинтегрировать систему дифференциальных уравнений (3.84) с управлением $(\boldsymbol{u}_1(t,x^1),\ldots,\boldsymbol{u}_q(t,x^1))^T$ одним из численных методов, например методом Рунге–Кутты 4-го порядка. В результате получаются пары $d^k, k=1,...,N$, образованные и управлением $u^k(t)=(u_1^k(t)=\boldsymbol{u}_1(t,x^{1,k}(t)),\ldots,u_q^k(t)=\boldsymbol{u}_q(t,x^{1,k}(t)))^T \quad \forall t\in T,$ и соответствующими траекториями $x^k(t)=(x^{1,k}(t),x^{2,k}(t))^T$. Следовательно, можно найти приближенное значение критерия качества управления пучком траекторий:

$$J\left[\boldsymbol{u}(t,x^1)\right]\cong\frac{1}{N}\sum_{k=1}^{N}I\left(x_0^k,d^k\right). \tag{3.90}$$

ПАРАМЕТРИЗАЦИЯ ЗАКОНА УПРАВЛЕНИЯ С НЕПОЛНОЙ ОБРАТНОЙ СВЯЗЬЮ

Реализуем переход от задачи (3.89) к задаче конечномерной оптимизации, т.е. проблеме поиска наилучших значений параметров, задающих структуру управления. Управление $\boldsymbol{u}(t,x^1)$ ищется в параметрическом виде, определяемом числом коэффициентов в разложении управления по выбираемой системе базисных функций и их значениями.

Для параметризации закона управления рассматривается применение ортонормированных систем базисных функций, получивших широкое распространение в спектральном методе [50, 51]. Предлагается искать закон управления в виде функции насыщения sat, гарантирующей выполнение параллелепипедных ограничений на управление:

$$\boldsymbol{u}_j(t,x^1)=\operatorname{sat}\left\{g_j\left(t,x_1,...,x_m\right)\right\},\ j=1,...,q. \tag{3.91}$$

Здесь

$$\operatorname{sat}v_j(t)=\begin{cases}v_j(t), & a_j(t)\le v_j(t)\le b_j(t),\\ a_j(t), & v_j(t)<a_j(t),\\ b_j(t), & v_j(t)>b_j(t),\end{cases} \tag{3.92}$$

где $\nu_j(t) = g_j\left(t, x_1(t), ..., x_m(t)\right)$, а аргументы $g_j\left(t, x_1, ..., x_m\right)$ функции насыщения предлагается искать в виде линейной комбинации базисных функций:

$$g_j\left(t, x_1, ..., x_m\right) = \sum_{i_0=0}^{L_0^j-1} \sum_{i_1=0}^{L_1^j-1} \cdots \sum_{i_m=0}^{L_m^j-1} c^j_{i_0 i_1 \ldots i_m} q\left(i_0, t\right) p_1\left(i_1, x_1\right) \cdots p_m\left(i_m, x_m\right), \tag{3.93}$$

где $c^j_{i_0 i_1 \ldots i_m}$ – неизвестные коэффициенты; $L_0^j, L_1^j, ..., L_m^j$ – масштабы усечения по времени и координатам вектора состояния, используемым в управлении.

В качестве базисных функций $q\left(i_0, t\right)$, $p_k\left(i_k, x_k\right), k = 1, ..., m,$ возьмем ортонормированную на отрезке $\left[0; t_f\right]$ систему нестационарных косинусоид:

$$q(i_0, t) = \begin{cases} \sqrt{\dfrac{1}{t_f}}, & i_0 = 0, \\ \sqrt{\dfrac{2}{t_f}} \cos \dfrac{i_0 \pi t}{t_f}, & i_0 = 1, ..., L_0 - 1, \end{cases}$$

$$p_k(i_k, x_k) = \begin{cases} \sqrt{\dfrac{1}{\overline{x}_k - \underline{x}_k}}, & i_k = 0, \\ \sqrt{\dfrac{2}{\overline{x}_k - \underline{x}_k}} \cos \dfrac{i_k \pi (x_k - \underline{x}_k)}{\overline{x}_k - \underline{x}_k}, & i_k = 1, ..., L_k - 1. \end{cases} \tag{3.94}$$

Здесь предполагается, что $t_0 = 0$ и известна оценка множества возможных состояний, которая представляется прямым произведением $[\underline{x}_1, \overline{x}_1] \times \cdots \times [\underline{x}_n, \overline{x}_n]$, где $\underline{x}_i, \overline{x}_i$ – нижняя и верхняя границы по каждой координате соответственно, определяемые физическим смыслом решаемой задачи.

Неизвестные параметры j-й координаты закона управления: матрица-столбец масштабов усечения $L^j = (L_0^j, L_1^j, ..., L_m^j)^T$ и наилучших значений коэффициентов разложения $c^j_{i_0, i_1, ..., i_m}$, удобно представить в виде блочной гиперстолбцовой матрицы:

$$C^j = \left[\begin{bmatrix} c^j_{00...0} \\ c^j_{00...1} \\ \vdots \\ c^j_{00...(L_m^j-1)} \end{bmatrix} \begin{bmatrix} c^j_{10...0} \\ c^j_{10...1} \\ \vdots \\ c^j_{10...(L_m^j-1)} \end{bmatrix} \cdots \begin{bmatrix} c^j_{(L_0^j-1)0...0} \\ c^j_{(L_0^j-1)0...1} \\ \vdots \\ c^j_{(L_0^j-1)0...(L_m^j-1)} \end{bmatrix} \cdots \begin{bmatrix} c^j_{(L_0^j-1)(L_1^j-1)...0} \\ c^j_{(L_0^j-1)(L_1^j-1)...1} \\ \vdots \\ c^j_{(L_0^j-1)(L_1^j-1)...(L_m^j-1)} \end{bmatrix} \right]^T, \quad j = 1, ..., q.$$

Тогда векторы неизвестных, подлежащих поиску, можно записать в форме объединенных матриц-столбцов

$$L = (L^1, ..., L^q)^T, \quad C = (C^1, ..., C^q)^T \tag{3.95}$$

с ограничениями на компоненты:

$$0 \le L_i^j \le L_{\max}, \quad L_i^j - \text{целые}; \quad j = 1, ..., q; \; i = 0, ..., m, \tag{3.96}$$

$$c_{\min} \le c^j_{i_0, i_1, ..., i_m} \le c_{\max}, \; i_0 = 0, 1, ..., L_0^j - 1; ..., i_m = 0, 1, ..., L_m^j - 1;$$

где значения $L_{\max}$, $c_{\min}, c_{\max}$ задаются исходя из возможных требований к точности решения и лимиту вычислительных затрат.

Исходя из описанного способа параметризации управления, предлагается искать значения координат блочной матрицы (L,C), содержащей блок L целочисленных переменных и блок C действительных переменных, на координаты которой наложены интервальные ограничения (3.96).

Таким образом, предлагается перейти к решению смешанной целочисленно-непрерывной задачи условной оптимизации:

$$f(L^*,C^*) = \min_{L\in\boldsymbol{L},C\in\boldsymbol{C}} f(L,C), \tag{3.97}$$

где

$$\boldsymbol{L} = \{L = (L^1,...,L^q)^T \,|\, L_i^j \in \{0,...,L_{\max}\}, j = 1,...,q; i = 0,1,...,m\},$$

$$\boldsymbol{C} = \{C = (C^1,....,C^q)^T \,|\, c^j_{i_0,i_1,...,i_m} \in [c_{\min}, c_{\max}], i_0 = 0,1,...,L_0^j - 1;...; i_m = 0,1,...,L_m^j - 1; j = 1,...,q\},$$

значение целевой функции $f(L,C)$ вычисляется по формуле (3.90).

Для решения задачи (3.97) предлагается применить модифицированную процедуру целочисленного линейного поиска и гибридные мультиагентные алгоритмы, описанные в разд. 1.3 и 1.4.

Опишем пошаговый алгоритм поиска оптимального управления с неполной обратной связью, используя предложенный подход.

Шаг 1. *Задание параметров.* Выбрать мультиагентный алгоритм оптимизации и задать его параметры. Задать количество координат m, информация о которых используется в управлении, ограничение на максимальное значение масштаба усечения $L_{\max}$, ограничения $c_{\min}$ и $c_{\max}$ на значения коэффициентов в разложении (3.93), максимальное число проходов P. Указать нижнюю и верхнюю границы $\underline{x}_i, \overline{x}_i$ по каждой координате, а также множество начальных состояний $\Omega = [\alpha_1,\beta_1]\times...\times[\alpha_n,\beta_n]$. Положить $it = 0$ (счетчик числа итераций), $p = 0$ (счетчик числа проходов).

Шаг 2. *Генерирование начального приближения.*

Для целочисленных переменных:

$$L_i^j = \mathrm{INT}\left[\mathrm{rand}[0;1]\ (L_{\max}^j + 1)\right],\ 0 \le L_i^j \le L_{\max},\ i = 0,...,m;\ j = 1,...,q,$$

где $\mathrm{rand}[0;1]$ – число, генерируемое согласно равномерному закону распределения, $\mathrm{INT}[.]$ – операция выделения целой части числа.

Для непрерывных переменных требуется сгенерировать начальную популяцию, состоящую из NP агентов, значения координат $c^j_{i_0,i_1,...,i_m}$ которых определены на интервале $[c_{\min}, c_{\max}]$:

$$c^j_{i_0,i_1,...,i_m} = c_{\min} + \mathrm{rand}[0;1](c_{\max} - c_{\min}),\ j = 1,...,q; i_0 = 0,1,...,L_0^j - 1;...; i_m = 0,1,...,L_m^j - 1.$$

Шаг 3. *Реализация процедуры поиска на множестве $\boldsymbol{L}$ по целочисленным переменным при известных значениях координат вектора C.* Значения координат вектора C в блочном векторе (L,C) считаются фиксированными.

Шаг 3.1. Упорядочить координаты вектора L случайным образом. Положить $New = false$ – индикатор наличия улучшения, $j = 1$.

Шаг 3.2. Пусть на j-м месте стоит r-я переменная. "Просканировать" множество решений вида $Z = L + ke_r, k = 0, \pm 1, \pm 2, ..., 0 \le Z \le L^{\max}$, где e_r – единичный орт размеров $(q(m+1)\times 1)$, составленный из всех нулей и одной единицы на r-м месте; знак неравенства понимается покоординатно, $L^{\max}$ – вектор размеров $(q(m+1)\times 1)$, все координаты которого равны $L_{\max}$.

Шаг 3.3. Если среди полученных в результате сканирования решений имеется наилучшее, удовлетворяющее условию $f(Z,C) < f(L,C)$, то заменить L на Z и положить $New = true$.

Шаг 3.4. Если $j < q(m+1)$, то положить $j = j+1$ и перейти к шагу 3.2. Если $j = q(m+1)$ и если $New = true$, то положить $New = false$, $j = 1$ и перейти к шагу 3.2. Иначе поиск завершить и положить $p = p+1$ (счетчик числа проходов).

Шаг 4. *Реализация процедуры поиска на множестве C по непрерывным переменным при фиксированных координатах вектора L.*

Шаг 4.1. Для каждого из NP агентов по сгенерированным коэффициентам, входящим в C, сформировать управление в виде функции насыщения sat, гарантирующей выполнение ограничений на управление:

$$\boldsymbol{u}_j(t,x^1) = \mathrm{sat}\left\{g_j\left(t,x_1,...,x_m\right)\right\},\ j = 1,...,q,$$

$$g_j\left(t,x_1,...,x_m\right) = \sum_{i_0=0}^{L_0^j-1}\sum_{i_1=0}^{L_1^j-1}\cdots\sum_{i_m=0}^{L_m^j-1} c^j_{i_0 i_1 \ldots i_m}\, q\left(i_0,t\right)p_1\left(i_1,x_1\right)\cdots p_m\left(i_m,x_m\right).$$

В качестве базисных функций $q\left(i_0,t\right)$, $p_k\left(i_k,x_k\right), k = 1,...,m$ возьмем ортонормированную на отрезке $\left[0;t_f\right]$ систему нестационарных косинусоид:

$$q(i_0,t) = \begin{cases} \sqrt{\dfrac{1}{t_f}}, & i_0 = 0, \\ \sqrt{\dfrac{2}{t_f}}\cos\dfrac{i_0\pi t}{t_f}, & i_0 = 1,...,L_0^j - 1, \end{cases}$$

$$p_k(i_k,x_k) = \begin{cases} \sqrt{\dfrac{1}{\overline{x}_k - \underline{x}_k}}, & i_k = 0, \\ \sqrt{\dfrac{2}{\overline{x}_k - \underline{x}_k}}\cos\dfrac{i_k\pi(x_k - \underline{x}_k)}{\overline{x}_k - \underline{x}_k}, & i_k = 1,...,L_k^j - 1. \end{cases}$$

Шаг 4.2. Проинтегрировать NP систем дифференциальных уравнений (3.84) с управлениями $\boldsymbol{u}^1(t,x^1),\ldots,\boldsymbol{u}^{NP}(t,x^1)$, например, методом Рунге–Кутты 4-го порядка для каждого начального состояния x_0^k из множества Ω. Для каждого агента и для каждого начального состояния получить соответствующие траектории $(x_1^{1,x_0^k}(\cdot),\ldots,x_n^{1,x_0^k}(\cdot)), k = 1,...,N;\ldots;(x_1^{NP,x_0^k}(\cdot),\ldots,x_n^{NP,x_0^k}(\cdot)), k = 1,...,N$ и вычислить значения

функционала $I^{1,x_0^k},\ldots,I^{NP,x_0^k}$. $k=1,\ldots,N$. Найти значения критерия качества управления пучком траекторий по формуле (3.90): $J^1,\ldots,J^{NP}$.

Шаг 4.3. Выполнить очередную итерацию выбранного мультиагентного метода минимизации функционала (3.90). Получить новые положения агентов $1',\ldots,NP'$ (векторы значений коэффициентов разложения по системе базисных функций).

Шаг 4.4. Проверка критериев окончания поиска по непрерывным переменным.

Если $it < ITER$, то положить $it = it + 1$ и перейти к шагу 4.1.

Если $it \geq ITER$, завершить процедуру поиска значений непрерывных переменных и перейти к шагу 5.

Шаг 5. *Завершение поиска.* Если $p < P$, перейти к шагу 3. Если $p = P$, поиск завершить. Выбрать в качестве приближенного решения наилучшего агента в популяции, т.е. вектор L с найденными значениями его координат и вектор C с координатами $c^j_{i_0,i_1,\ldots,i_m}$, $i_0 = 0,\ldots,L_0^j - 1;\ldots;i_m = 0,\ldots,L_m^j - 1$, соответствующие ему управление и пучок траекторий, а также значение J функционала (3.90).

3.7.3. Модельные примеры

Пример 3.12. Постановка задачи приведена в табл. 3.29.

Таблица 3.29. Постановка задачи (пример 3.12)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = x_2(t) + \sin x_1(t) + u(t) \\ \dot{x}_2 = x_1(t)\cos x_2(t)u(t) \end{cases}$
Временной интервал	$t \in [0;1]$
Ограничения на управление	$-1 \leq u \leq 1$
Функционал (3.86)	$I(x_0, d) = x_2(1)$

Множество начальных состояний $\Omega = [-0,05;\ 0,05] \times [-0,05;\ 0,05]$, $N_1 = N_2 = 5$.

Требуется минимизировать среднее значение функционала (3.86) на множестве начальных состояний Ω.

В качестве базисных функций использовалась система нестационарных косинусоид (3.94). Наложены ограничения на значения коэффициентов $c^j_{i_0,i_1,\ldots,i_m}$ в управлении: $c_{\min} = -5$, $c_{\max} = 5$ и на масштабы усечения $L_{\max} = 5$.

Использованы следующие параметры выбранных мультиагентных алгоритмов:

– мультиагентного алгоритма, основанного на применении линейных регуляторов для управления движением агентов (разд.1.4): $NP = 9$, $NMAX = 10$, $P_{\max} = 5$, $k_s = 0{,}1$, $k_l = 5$, $h = 0{,}0001$;

– гибридного мультиагентного алгоритма интерполяционного поиска (разд. 1.3): $NP = 30$, $I_{\max} = 10$, $M_1 = 5$, $M_2 = 5$, $PRTVector = 0{,}01$, $nstep = 5$, $b_2 = 8$.

Решены задачи синтеза оптимального управления с полной и неполной обратной связью, включая случай $m = 0$ (программное управление). Для каждого случая определялось оптимальное количество коэффициентов в разложении L_0, L_1, L_2 и значения коэффициентов. Результаты решения задачи приведены в табл. 3.30, 3.31.

Таблица 3.30. Результаты решения примера 3.12 гибридным мультиагентным методом интерполяционного поиска

Закон управления	$u(t)$	$\boldsymbol{u}(t, x_1)$	$\boldsymbol{u}(t, x_2)$	$\boldsymbol{u}(t, x_1, x_2)$
Масштабы усечения	$L_0 = 2, L_1 = 0,$ $L_2 = 0$	$L_0 = 2, L_1 = 2,$ $L_2 = 0$	$L_0 = 2, L_1 = 0,$ $L_2 = 1$	$L_0 = 2, L_1 = 2,$ $L_2 = 1$
Отрезки значений координат $x_1(1), x_2(1)$	[0,267; 0,617], [– 0,176; – 0,091]	[0,399; 0,444], [– 0,176; – 0,093]	[0,281; 0,632], [– 0,176; – 0,092]	[0,465; 0,504], [–0,177;–0,092]
Значение функционала (3.90)	– 0,1337	– 0,13435	– 0,1343	– 0,13467
Значения коэффициентов в разложении (3.93)	0,02; 3,51	0,27; 2,18; 2,26; 2,34	0,06; 3,51	0,89; 2,4; 2,47; 2,5

Таблица 3.31. Результаты решения примера 3.12 мультиагентным методом, основанным на применении линейных регуляторов

Закон управления	$u(t)$	$\boldsymbol{u}(t, x_1)$	$\boldsymbol{u}(t, x_2)$	$\boldsymbol{u}(t, x_1, x_2)$
Масштабы усечения	$L_0 = 2, L_1 = 0,$ $L_2 = 0$	$L_0 = 2, L_1 = 2,$ $L_2 = 0$	$L_0 = 2, L_1 = 0,$ $L_2 = 1$	$L_0 = 2, L_1 = 2,$ $L_2 = 1$
Отрезки значений координат $x_1(1), x_2(1)$	[0,271; 0,621], [– 0,176; – 0,091]	[0,444; 0,485], [– 0,177; – 0,093]	[0,279; 0,63], [– 0,176; – 0,092]	[0,428; 0,469], [–0,177; – 0,093]
Значение функционала (3.90)	– 0,13356	– 0,13472	– 0,134326	– 0,13522
Значения коэффициентов в разложении (3.93)	0,05; 3,38	0,7; 2,5; 2,5; 2,5	0,06; 3,54	0,65; 2,5; 2,5; 2,5

Исходя из полученных численных результатов, лучшее значение критерия оптимальности управления пучком траекторий получается при управлении с полной обратной связью, а наихудшее при использовании программного управления. Примененные мультиагентные алгоритмы не уступают друг другу в точности. Найденные наилучшие пучок траекторий и управление для случая $m = 2$ изображены на рис. 3.19.

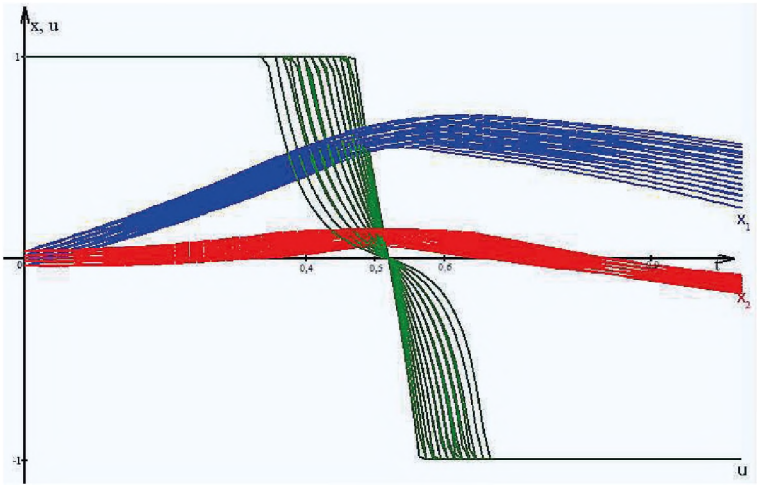


Рис. 3.19. Графики управления и траекторий, полученные в примере 3.12

В разд. 3.6 найдено оптимальное в среднем программное управление пучком траекторий с использованием многочленов Чебышева и тех же мультиагентных алгоритмов (разд. 1.3, 1.4). При этом получены отрезки значений координат в конце времени функционирования системы: $x_1(1) \in [0,2596; 0,6102]$; $x_2(1) \in [-0,176; -0,0911]$, значение функционала (3.90) равно – 0,133572. Применение обеих базисных систем (нестационарных косинусоид и многочленов Чебышева) дает близкие результаты.

Кроме того, задача 3.12 решена с помощью метаэвристического мультиагентного метода, имитирующего поведение стаи криля [64, 89, 159], входящего в качестве модуля в последовательно-параллельный гибридный метод, описанный в разд. 1.6, с параметрами $NP = 5$, $I_{\max} = 100$, $\mu = 0,5$, $N_{\max} = 0,01$, $V_f = 0,02$, $D_{\max} = 0,005$. Численные результаты приведены в табл. 3.32. С ростом степени информированности о координатах вектора состояния, как и в случаях применения вышеописанных методов, значение критерия качества управления пучком траекторий улучшается.

Таблица 3.32. Результаты решения примера 3.12 методом, имитирующим поведение стаи криля

Закон управления	$u(t)$	$\boldsymbol{u}(t, x_1)$	$\boldsymbol{u}(t, x_2)$	$\boldsymbol{u}(t, x_1, x_2)$
Масштабы усечения	$L_0 = 2, L_1 = 0,$ $L_2 = 0$	$L_0 = 2, L_1 = 1,$ $L_2 = 0$	$L_0 = 2, L_1 = 0,$ $L_2 = 1$	$L_0 = 2, L_1 = 1,$ $L_2 = 1$
Отрезки значений координат $x_1(1), x_2(1)$	[–0,371; 0,802], [–0,205; –0,04]	[0,364; 0,707], [–0,199; –0,07]	[0,122; 0,53], [–0,25; –0,049]	[0,381; 0,652], [–0,25; –0,059]
Величина функционала (3.90)	– 0,14229	– 0,15246	– 0,15413	– 0,15858
Значения коэффициентов в разложении (3.93)	0,58; 2,56	– 0,01; 3,01	– 0,7; 3,43	– 0,03; 2,04

Пример 3.13. Постановка задачи приведена в табл. 3.32. Множество начальных состояний $\Omega = [0,95;\ 1,05] \times [-0,05;\ 0,05]$, $N_1 = N_2 = 5$.

Требуется минимизировать среднее значение функционала (3.86) на заданном множестве начальных состояний Ω.

Таблица 3.33. Постановка задачи (пример 3.13)

Система дифференциальных уравнений	$\begin{cases} \dot{x}_1 = \dfrac{1}{\cos x_1(t) + 2} + 3\sin x_2(t) + u(t) \\ \dot{x}_2 = x_1(t) + x_2(t) + u(t) \end{cases}$
Временной интервал	$t \in [0;\ 1,6]$
Ограничения на управление	$-2 \le u \le 1$
Функционал (3.86)	$I(x_0, d) = -x_1(1,6) + \dfrac{1}{2} x_2(1,6)$

Использованы следующие параметры мультиагентных алгоритмов:

– мультиагентного алгоритма, основанного на использовании линейных регуляторов управления движением агентов (разд. 1.4): $NP = 9$, $NMAX = 10$, $P_{\max} = 5$, $k_s = 0,1$, $k_l = 5$, $h = 0,0001$;

– гибридного мультиагентного алгоритма интерполяционного поиска (разд. 1.3): $NP = 20$, $I_{\max} = 10$, $M_1 = 2$, $M_2 = 5$, $PRTVector = 0,01$, $nstep = 5$, $b_2 = 8$.

В качестве базисных функций применялась система нестационарных косинусоид (3.94). Наложены ограничения на значения коэффициентов $c^j_{i_0 i_1 \ldots i_m}$ в управлении: $c_{\min} = 0$, $c_{\max} = 350$ и на масштабы усечения $L_{\max} = 5$.

Решены задачи синтеза оптимального управления с полной и неполной обратной связью, включая случай $m = 0$ (программное управление). Для каждого случая определялось наилучшее количество коэффициентов в разложении L_0, L_1, L_2 и значения коэффициентов. Результаты решения примера приведены в табл. 3.34, 3.35.

Таблица 3.34. Результаты решения примера 3.13 гибридным мультиагентным методом интерполяционного поиска

Закон управления	$u(t)$	$\boldsymbol{u}(t, x_1)$	$\boldsymbol{u}(t, x_1)$	$\boldsymbol{u}(t, x_1, x_2)$
Масштабы усечения	$L_0 = 2, L_1 = 0,$ $L_2 = 0$	$L_0 = 2, L_1 = 1,$ $L_2 = 0$	$L_0 = 2, L_1 = 0,$ $L_2 = 1$	$L_0 = 2, L_1 = 1,$ $L_2 = 1$
Отрезки значений координат $x_1(1,6), x_2(1,6)$	[3,364; 3,452], [12,276; 13,147]	[3,412; 3,521], [12,391; 13,264]	[3,406; 3,511], [12,343; 13,217]	[3,4628; 3,6], [12,528; 13,403]
Значение функционала (3.90)	–2,96042	–2,96212	–2,94643	–2,96575
Значения коэффициентов в (3.93)	13,76; 13,83	22,87; 31,52	24,41; 35,29	117,21; 151,69

На основе анализа полученных численных результатов можно заключить, что наилучшее значение критерия получается при управлении с полной обратной связью,

а наихудшее – при программном управлении. При применении алгоритма, основанного на использовании линейных регуляторов для управления движением агентов, был получен наилучший результат. Найденные наилучшие управление и соответствующий пучок траекторий для случая $m = 2$ изображены на рис. 3.20.

Таблица 3.35. Результаты решения примера 3.13 мультиагентным методом, основанным на использовании линейных регуляторов

Закон управления	$u(t)$	$\boldsymbol{u}(t, x_1)$	$\boldsymbol{u}(t, x_2)$	$\boldsymbol{u}(t, x_1, x_2)$
Масштабы усечения	$L_0 = 2, L_1 = 0,$ $L_2 = 0$	$L_0 = 2, L_1 = 1,$ $L_2 = 0$	$L_0 = 2, L_1 = 0,$ $L_2 = 1$	$L_0 = 2, L_1 = 1,$ $L_2 = 1$
Отрезки значений координат $x_1(1,6), x_2(1,6)$	[3,324; 3,396], [12,177; 13,045]	[3,356; 3,439], [12,269; 13,137]	[3,401; 3,505], [12,381; 13,252]	[3,419; 3,532], [12,43; 13,302]
Значение функционала (3.90)	–2,95377	–2,96564	–2,970204	–2,97337
Значения коэффициентов в (3.93)	30,69; 31,16	173,756; 239,405	71,40; 96,33	216,79; 284,65

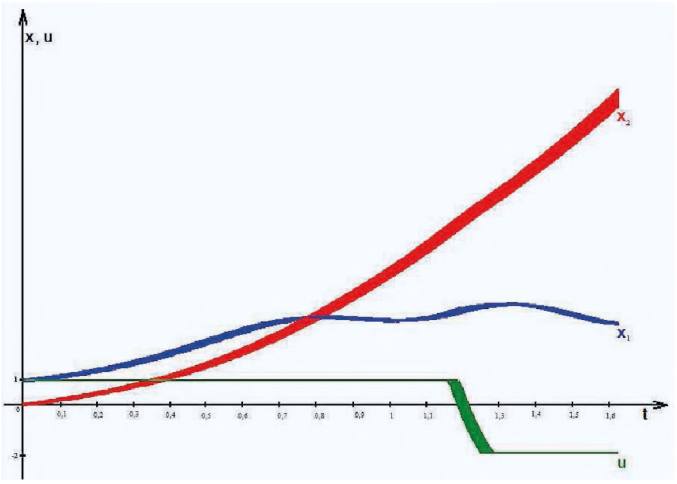


Рис. 3.20. Графики управления и траекторий, полученные в примере 3.13

В разд. 3.6 получены результаты решения рассматриваемой задачи в условиях неопределенности задания начального состояния системы с использованием псевдоспектрального метода на основе многочленов Чебышева. Получены следующие результаты поиска оптимального в среднем программного управления: значение функционала качества (3.90) управления пучком равно – 2,961588, отрезки значений координат $x_1(1) \in [3,4351; 3,5566]$; $x_2(1) \in [12,4475; 13,3215]$, что свидетельствует о близости полученных результатов.

Кроме того, задача 3.13 решена с помощью метаэвристического мультиагентного метода, имитирующего поведение стаи криля [64, 89, 159], входящего в качестве

модуля в последовательно-параллельный гибридный метод, описанный в разд. 1.6, с параметрами $NP=5$, $I_{\max}=100$, $\mu=0,5$, $N_{\max}=0,01$, $V_f=0,02$, $D_{\max}=0,005$. Численные результаты приведены в табл. 3.36. С ростом степени информированности о координатах вектора состояния, как и в случаях применения вышеописанных методов, значение критерия качества управления пучком траекторий улучшается.

Таблица 3.36. Результаты решения примера 3.13 методом, имитирующим поведение стаи криля

Законы управления	$u(t)$	$\boldsymbol{u}(t,x_1)$	$\boldsymbol{u}(t,x_2)$	$\boldsymbol{u}(t,x_1,x_2)$
Масштабы усечения	$L_0=2, L_1=0,$ $L_2=0$	$L_0=2, L_1=1,$ $L_2=0$	$L_0=2, L_1=0,$ $L_2=1$	$L_0=2, L_1=1,$ $L_2=1$
Отрезки значений координат $x_1(1,6), x_2(1,6)$	[3,413; 3,67], [11,365; 12,57]	[3,498; 3,735], [11,708; 12,67]	[3,679; 3,887], [12,197;13,17]	[3,62; 3,765], [11,92; 13,318]
Величина функционала (3.90)	–2,46526	–2,51188	–2,53738	–2,56689
Значения коэффициентов в (3.93)	95,81; 96,11	46,66; 61,06	136,46; 159,51	158,1; 196,16

Пример 3.14. Постановка задачи приведена в табл. 3.37.

Таблица 3.37. Постановка задачи (пример 3.14)

Система дифференциальных уравнений	$\begin{cases}\dot{x}_1 = x_2^2(t)+u(t)\\ \dot{x}_2 = 8\sin x_1(t)+x_1(t)-x_2(t)-u(t)\end{cases}$
Временной интервал	$t\in[0;2]$
Ограничения на управление	$-1\le u\le 2$
Функционал (3.86)	$I(x_0,d)=-x_1(2)\to\min$

Множество начальных состояний $\Omega=[-1,05;-0,95]\times[-0,05;\,0,05]$, $N_1=N_2=5$.

Требуется минимизировать среднее значение функционала (3.86) на заданном множестве начальных состояний Ω.

Использованы следующие параметры мультиагентных алгоритмов:

– мультиагентного алгоритма, основанного на использовании линейных регуляторов управления движением агентов (разд. 1.4): $NP=25$, $NMAX=10$, $P_{\max}=5$, $k_s=1$, $k_l=10$, $h=0,0001$;

– гибридного мультиагентного алгоритма интерполяционного поиска (разд. 1.3): $NP=20$, $I_{\max}=10$, $M_1=1$, $M_2=7$, $PRTVector=0,9$, $nstep=5$, $b_2=8$.

В качестве базисных функций применялась система нестационарных косинусоид (3.94). Наложены ограничения на значения коэффициентов $c^j_{i_0 i_1 \dots i_m}$ в управлении: $c_{\min}=-200$, $c_{\max}=200$ и на масштабы усечения $L_{\max}=5$. Решены задачи синтеза оптимального управления с полной и неполной обратной связью, включая случай $m=0$ (программное управление). Для каждого случая определялось наилучшее количество коэффициентов в разложении L_0, L_1, L_2 и значения коэффициентов. Результаты решения примера приведены в табл. 3.38, 3.39.

Таблица 3.38. Результаты решения примера 3.14 гибридным мультиагентным методом интерполяционного поиска

Закон управления	$u(t)$	$\boldsymbol{u}(t, x_1)$	$\boldsymbol{u}(t, x_2)$	$\boldsymbol{u}(t, x_1, x_2)$
Масштабы усечения	$L_0 = 3, L_1 = 0,$ $L_2 = 0$	$L_0 = 3, L_1 = 3,$ $L_2 = 0$	$L_0 = 3, L_1 = 0,$ $L_2 = 3$	$L_0 = 4, L_1 = 2,$ $L_2 = 2$
Отрезки значений координат $x_1(2), x_2(2)$	[15,73; 16,11], [6,119; 6,193]	[16,51; 16,92], [6,432; 6,459]	[16,27; 16,78], [6,291; 6,322]	[16,49; 16,87], [6,276; 6,303]
Значение функционала (3.90)	–15,9101	–16,7081	–16,53089	–16,6678
Значения коэффициентов в разложении (3.93)	–32,82; 82,99; 118,42	–69,08; –43,71; –17,3; 6,66; 10,44; 27,41; 51,52; 115,74; 148,07	–97,7; –83,14; 45,69; 46,61; 46,74; 47,32; 52,03; 113,68; 124,24	–134,4; –130,3; –60,9; –15,06; 109,81; 140,24; 144,03; 167,49; 184,32; 189,11; 190,01; 199,97; 200; 200; 200; 200

Таблица 3.39. Результаты решения примера 3.14 мультиагентным методом, основанным на использовании линейных регуляторов

Закон управления	$u(t)$	$\boldsymbol{u}(t, x_1)$	$\boldsymbol{u}(t, x_2)$	$\boldsymbol{u}(t, x_1, x_2)$
Масштабы усечения	$L_0 = 2, L_1 = 0,$ $L_2 = 0$	$L_0 = 2, L_1 = 2,$ $L_2 = 0$	$L_0 = 2, L_1 = 0,$ $L_2 = 2$	$L_0 = 2, L_1 = 3,$ $L_2 = 2$
Отрезки значений координат $x_1(2), x_2(2)$	[15,573; 15,972], [6,499; 6,593]	[15,6; 15,976], [6,114; 6,179]	[15,687; 16,072], [6,159; 6,225]	[15,588; 16,019], [6,304; 6,37]
Значение функционала (3.90)	–15,76102	–15,77986	–15,87158	–15,78959
Значения коэффициентов в разложении (3.93)	–60,33; 69,5	–43,46; –0,2; 0,5; 57,16	–91,21; –25,97; 30,59; 94,7	–179,6; –118,9; –0,1; 0,1; 0,1; 0,1; 0,15; 0,2; 35,98; 52,55; 57,11; 154,74

На основе анализа полученных численных результатов из таблиц 3.38 и 3.39 можно заключить, что описанный подход при решении примера 3.14 хотя и позволил получить приближенное решение задачи, но достигнутая при этом точность не достаточна. Наилучшее относительное значение критерия получается при управлении с неполной обратной связью, а наихудшее – при программном управлении. При применении гибридного мультиагентного алгоритма интерполяционного поиска получен

наилучший результат (см. табл. 3.38). Найденные наилучшие управление и соответствующий пучок траекторий для случая управления $\boldsymbol{u}(t, x_2)$ изображены на рис. 3.21.

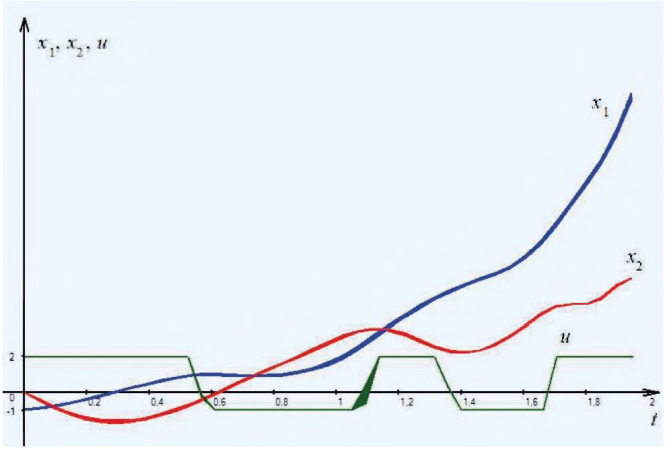


Рис. 3.21. Графики управления и траекторий, полученные в примере 3.14

Кроме того, задача 3.14 решена с помощью метаэвристического мультиагентного метода, имитирующего поведение стаи криля [64, 89, 159], входящего в качестве модуля в последовательно-параллельный гибридный метод, описанный в разд. 1.6, с параметрами $NP = 5$, $I_{\max} = 100$, $\mu = 0{,}5$, $N_{\max} = 0{,}01$, $V_f = 0{,}02$, $D_{\max} = 0{,}005$. Численные результаты приведены в табл. 3.40.

Таблица 3.40. Результаты решения примера 3.14 методом, имитирующим поведение стаи криля

Закон управления	$u(t)$	$\boldsymbol{u}(t, x_1)$	$\boldsymbol{u}(t, x_2)$	$\boldsymbol{u}(t, x_1, x_2)$
Масштабы усечения	$L_0 = 2, L_1 = 0,$ $L_2 = 0$	$L_0 = 2, L_1 = 2,$ $L_2 = 0$	$L_0 = 2, L_1 = 0,$ $L_2 = 2$	$L_0 = 2, L_1 = 2,$ $L_2 = 2$
Отрезки значений координат $x_1(2), x_2(2)$	[14,97; 16,858], [5,394; 7,073]	[17,658; 19,03], [7,178; 7,255]	[19,3; 21,244], [6,464; 7,087]	[20,147; 22,83], [7,362; 8,446]
Величина функционала (3.90)	–15,54046	–18,38359	–20,2914	–21,49857
Значения коэффициентов в разложении (3.93)	–11,32; 23,01;	–19,91; –5,88; 12,28; 13,76	–20,18; –2,81; 3,87; 33,12	–28,41; –10,6; –6,59; –5,75; –2,55; 9,26; 16,43; 28,06

Исходя из приведенных данных для модельных примеров 3.12, 3.13 и 3.14, можно сделать вывод о том, что предложенный подход нахождению управления с неполной обратной связью на основе параметризации закона управления и решения смешанной целочисленно-непрерывной задачи условной оптимизации с применением гибридных мультиагентных алгоритмов позволяет получать решения достаточно высокого качества. Продемонстрировано, что с ростом степени информированности о координатах вектора состояния значение критерия качества, как правило, улучшается.

3.8. ПОИСК ОПТИМАЛЬНОГО УПРАВЛЕНИЯ СТОХАСТИЧЕСКИМИ СИСТЕМАМИ С НЕПОЛНОЙ ОБРАТНОЙ СВЯЗЬЮ НА ОСНОВЕ РАЗЛОЖЕНИЙ ПО БАЗИСНЫМ СИСТЕМАМ

3.8.1. Постановка задачи

Рассматривается задача поиска оптимального в среднем управления траекториями непрерывных стохастических систем с неполной обратной связью. К данному классу задач относятся проблемы управления, в которых начальные состояния описываются заданным законом распределения, учитываются случайные воздействия на объект управления, а также предполагается, что в каждый момент времени доступна информация только о части координат вектора состояния системы. Описан способ параметризации закона управления с помощью разложений по различным системам базисных функций, используемых в спектральном методе. Предложено решать полученную в результате преобразований задачу параметрической оптимизации с использованием мультиагентного алгоритма на основе применения обобщенных ПИД-регуляторов для управления движением агентов (разд. 1.5).

Поведение модели объекта управления описывается стохастическим дифференциальным уравнением Ито:

$$dX = f(t, X(t), u(t))\,dt + \sigma(t, X(t), u(t))\,dW, \quad X(t_0) = X_0, \tag{3.98}$$

где X – вектор состояния системы, $X = (X^1, X^2)^T \in R^n$, $X^1 = (X_1, ..., X_m)^T$, $X^2 = (X_{m+1}, ..., X_n)^T$, $0 \le m \le n$; u – вектор управления, $u \in U \subseteq R^q$, U – некоторое заданное множество вида $U = [a_1, b_1] \times \cdots \times [a_q, b_q]$, $t \in T = [t_0, t_f]$, T – промежуток времени функционирования системы, моменты времени t_0 и t_f заданы; $W(t)$ – l-мерный стандартный винеровский случайный процесс; $f(t, x, u)$ – заданная вектор-функция размеров $(n \times 1)$, $\sigma(t, x, u)$ – заданная матричная функция размеров $(n \times l)$.

Предполагается, что о компонентах вектора $X^1 \in R^m$ текущая информация известна, а о компонентах вектора $X^2 \in R^{n-m}$ отсутствует.

Начальное состояние X_0 определяется плотностью вероятности

$$p(t_0, x) = p_0(x) \in P \qquad \forall x \in R^n, \tag{3.99}$$

где $P = \left\{ p(x) \mid p(x) \in C^2(R^n), \int\limits_{R^n} p(x)\,dx = 1,\ p(x) \ge 0 \quad \forall x \in R^n \right\}$, $C^k(R^n)$ – множество k-раз непрерывно дифференцируемых на множестве R^n функций.

Предполагается, что при управлении используется информация только о времени t и о компонентах вектора X^1, т.е. управление, применяемое в каждый момент времени $t \in T$, имеет вид управления с неполной обратной связью $u(t) = \boldsymbol{u}(t, X^1(t))$.

Число m, $0 \le m \le n$, определяется условиями информированности. При $m = n$ имеется информация обо всех компонентах вектора X, т.е. рассматривается система с полной обратной связью, а при $m = 0$ – система, разомкнутая по состоянию. В последнем случае применяется так называемое программное управление $u(t)$.

Множество допустимых управлений с неполной обратной связью $\mathcal{U}_m$ образуют функции $\boldsymbol{u}(t, x^1) : T \times R^m \to U$ такие, что для всех $i = 1, ..., n; j = 1, ..., l$ функции $f_i^{u(\cdot)}(t, x) = f_i(t, x, \boldsymbol{u}(t, x^1))$, $\sigma_{ij}{}^{u(\cdot)}(t, x) = \sigma_{ij}(t, x, \boldsymbol{u}(t, x^1))$ удовлетворяют условиям, при

которых решение уравнения (3.98) существует, единственно и является непрерывным марковским процессом [86, 87].

На отдельной траектории определим функционал

$$I = \int_{t_0}^{t_f} f^0(t, X(t), \boldsymbol{u}(t, X^1(t)))\, dt + F(X(t_f)), \tag{3.100}$$

где $f^0(t,x,u): T \times R^n \times U \to R$, $F(x): R^n \to R$ – заданные непрерывные функции.

Зададим функционал качества управления

$$J(\boldsymbol{u}(t,x^1)) = M\left[\int_{t_0}^{t_f} f^0(t, X(t), \boldsymbol{u}(t, X^1(t)))\, dt + F(X(t_f))\right], \tag{3.101}$$

где M – знак математического ожидания.

Требуется найти такое управление $\boldsymbol{u}^*(t,x^1) \in \mathcal{U}_m$, что

$$J(\boldsymbol{u}^*(t,x^1)) = \min_{\boldsymbol{u}(t,x^1) \in \mathcal{U}_m} J(\boldsymbol{u}(t,x^1)). \tag{3.102}$$

Искомое управление называется оптимальным в среднем, так как минимизируется среднее значение функционала (3.100) на множестве реализаций.

3.8.2. Алгоритм решения задачи

Рассмотрим сначала процедуру вычисления критерия оптимальности управления стохастической системой (3.98), а затем возможные варианты параметризации законов управления с неполной обратной связью.

ПРИБЛИЖЕННОЕ ВЫЧИСЛЕНИЕ ВЕЛИЧИНЫ ФУНКЦИОНАЛА

Генерируется N начальных состояний X_0^k, $k = 1,...,N$, в соответствии с заданной плотностью вероятности $p_0(x)$.

Для каждого начального состояния $X_0^k, k = 1,...,N$, следует проинтегрировать систему стохастических дифференциальных уравнений (3.98) с управлением $\boldsymbol{u}(t,x^1)$ одним из численных методов, например, методом Эйлера–Маруямы на промежутке времени $[t_0, t_f]$ с шагом $h = \dfrac{t_f - t_0}{S}$, где S – число подынтервалов на множестве T:

$$\begin{aligned} X^k(t_{s+1}) &= X^k(t_s) + h\, f(t_s, X^k(t_s), \boldsymbol{u}(t_s, X^1(t_s))) + \\ &+ \sqrt{h}\, \sigma(t_s, X^k(t_s), \boldsymbol{u}(t_s, X^1(t_s)))\xi, \quad s = 0,1,...,S-1, \\ X^k(t_0) &= X_0^k, k = 1,...,N, \end{aligned} \tag{3.103}$$

где $t_{s+1} = t_s + h$, $s = 0,1,...,S-1$; $t_S = t_f$; ξ – l-мерный случайный вектор, каждая координата которого является случайной величиной, имеющей стандартное нормальное распределение с нулевым математическим ожиданием и единичной дисперсией. Для ее моделирования можно воспользоваться методом Бокса–Мюллера:

$$\xi = \sqrt{-2\ln\alpha_1}\cos 2\pi\alpha_2 \text{ или } \xi = \sqrt{-2\ln\alpha_1}\sin 2\pi\alpha_2,$$

в котором α_1 и α_2 – независимые равномерно распределенные на интервале (0;1) случайные величины.

В результате получаются пары $d^k, k = 1,...,N$, образованные траекториями $X^k(t)$ и соответствующими управлениями $u^k(t) = \boldsymbol{u}(t, X^{1,k}(t))$, для которых вычисляются значения функционала (3.100): $I\left(X_0^k, d^k\right)$.

В результате можно найти приближенное значение критерия качества (3.101):

$$J\left[\boldsymbol{u}(t,x^1)\right] \cong \frac{1}{N}\sum_{k=1}^{N} I\left(X_0^k, d^k\right). \tag{3.104}$$

ПАРАМЕТРИЗАЦИЯ ЗАКОНА УПРАВЛЕНИЯ С НЕПОЛНОЙ ОБРАТНОЙ СВЯЗЬЮ

Реализуем переход от задачи (3.102) к задаче конечномерной оптимизации, т.е. проблеме поиска наилучших значений параметров, определяющих структуру управления. Управление $\boldsymbol{u}(t,x^1)$ ищется в параметрическом виде, определяемом числом коэффициентов в разложении управления по выбираемой системе базисных функций и их значениями.

В качестве способа параметризации закона управления могут применяться финитные базисные системы, радиально-базисные функции, многочлены Чебышева, Лежандра и др.

При использовании ортонормированных систем базисных функций, получивших широкое распространение в спектральном методе [50, 51], предлагается искать закон управления в виде функции насыщения sat, гарантирующей выполнение параллелепипедных ограничений на управление [112]:

$$\boldsymbol{u}_j(t,x^1) = \mathrm{sat}\left\{g_j\left(t,x_1,...,x_m\right)\right\},\ j = 1,...,q, \tag{3.105}$$

где

$$\mathrm{sat}\, v_j(t) = \begin{cases} v_j(t), & a_j \le v_j(t) \le b_j, \\ a_j, & v_j(t) < a_j, \\ b_j, & v_j(t) > b_j, \end{cases} \tag{3.106}$$

$v_j(t) = g_j\left(t, x_1(t),...,x_m(t)\right)$, а аргументы $g_j\left(t,x_1,...,x_m\right)$ функции насыщения искать в виде линейной комбинации базисных функций:

$$g_j\left(t,x_1,...,x_m\right) = \sum_{i_0=0}^{L_0^j-1}\sum_{i_1=0}^{L_1^j-1}\cdots\sum_{i_m=0}^{L_m^j-1} c^j_{i_0 i_1 \ldots i_m}\, q\left(i_0,t\right)p_1\left(i_1,x_1\right)\cdots p_m\left(i_m,x_m\right), \tag{3.107}$$

где $c^j_{i_0 i_1 \ldots i_m}$ – неизвестные коэффициенты; $L_0^j, L_1^j, ..., L_m^j$ – масштабы усечения по времени и координатам вектора состояния, используемым в управлении.

В качестве базисных функций $q\left(i_0,t\right)$, $p_k\left(i_k,x_k\right)$, $k = 1,...,m$, можно, например, взять ортонормированную на отрезке $\left[0;t_f\right]$ систему нестационарных косинусоид

$$q(i_0,t)=\begin{cases}\sqrt{\dfrac{1}{t_f}}, & i_0=0,\\ \sqrt{\dfrac{2}{t_f}}\cos\dfrac{i_0\pi t}{t_f}, & i_0=1,...,L_0-1,\end{cases}$$

$$p_k(i_k,x_k)=\begin{cases}\sqrt{\dfrac{1}{\overline{x}_k-\underline{x}_k}}, & i_k=0,\\ \sqrt{\dfrac{2}{\overline{x}_k-\underline{x}_k}}\cos\dfrac{i_k\pi(x_k-\underline{x}_k)}{\overline{x}_k-\underline{x}_k}, & i_k=1,...,L_k-1,\end{cases} \qquad (3.108)$$

многочленов Лежандра, функции Уолша и т.д. [50, 51].

Здесь предполагается, что $t_0=0$ и известна оценка множества возможных состояний, которая представляется прямым произведением $[\underline{x}_1,\overline{x}_1]\times\cdots\times[\underline{x}_n,\overline{x}_n]$, где $\underline{x}_i,\overline{x}_i$ – нижняя и верхняя границы по каждой координате соответственно, определяемые физическим смыслом решаемой задачи, граница множества возможных состояний является поглощающей.

Неизвестные параметры j-й координаты закона управления: матрица-столбец масштабов усечения $L^j=(L_0^j,L_1^j,...,L_m^j)^T$ и наилучших значений коэффициентов разложения $c^j_{i_0,i_1,...,i_m}$, удобно представить в виде блочной гиперстолбцовой матрицы:

$$C^j=\left[\begin{bmatrix}c^j_{00...0}\\ c^j_{00...1}\\ \vdots\\ c^j_{00...(L_m^j-1)}\end{bmatrix}\begin{bmatrix}c^j_{10...0}\\ c^j_{10...1}\\ \vdots\\ c^j_{10...(L_m^j-1)}\end{bmatrix}\cdots\begin{bmatrix}c^j_{(L_0^j-1)0...0}\\ c^j_{(L_0^j-1)0...1}\\ \vdots\\ c^j_{(L_0^j-1)0...(L_m^j-1)}\end{bmatrix}\cdots\begin{bmatrix}c^j_{(L_0^j-1)(L_1^j-1)...0}\\ c^j_{(L_0^j-1)(L_1^j-1)...1}\\ \vdots\\ c^j_{(L_0^j-1)(L_1^j-1)...(L_m^j-1)}\end{bmatrix}\right]^T,\quad j=1,...,q.$$

Тогда векторы неизвестных, подлежащих поиску, можно записать в форме объединенных матриц-столбцов

$$L=(L^1,...,L^q)^T,\quad C=(C^1,....,C^q)^T \qquad (3.109)$$

с ограничениями на компоненты:

$$0\le L_i^j\le L_{\max},\ \ L_i^j-\text{ целые};\ \ j=1,...,q,\ \ i=0,...,m, \qquad (3.110)$$

$$c_{\min}\le c^j_{i_0,i_1,...,i_n}\le c_{\max},\ i_0=0,1,...,L_0^j-1;...;\ i_m=0,1,...,L_m^j-1;$$

где значения $L_{\max}$, $c_{\min},c_{\max}$ задаются исходя из возможных требований к точности решения и лимиту вычислительных затрат.

Исходя из описанного способа параметризации управления, предлагается искать значения координат блочной матрицы (L,C), содержащей блок L целочисленных переменных и блок C действительных переменных, на координаты которой наложены интервальные ограничения (3.110).

Таким образом, предлагается перейти к решению смешанной целочисленно-непрерывной задачи условной оптимизации:

$$f(L^*, C^*) = \min_{L \in \boldsymbol{L}, C \in \boldsymbol{C}} f(L, C), \tag{3.111}$$

где

$$\boldsymbol{L} = \{L = (L^1, ..., L^q)^T \,|\, L_i^j \in \{0, ..., L_{\max}\}, j = 1, ..., q;\ i = 0, 1, ..., m\},$$

$$\boldsymbol{C} = \{C = (C^1, ..., C^q)^T \,|\, c^j_{i_0, i_1, ..., i_m} \in [c_{\min}, c_{\max}], i_0 = 0, 1, ..., L_0^j - 1; ...; i_m = 0, 1, ..., L_m^j - 1;\ j = 1, ..., q\},$$

значение целевой функции $f(L, C)$ вычисляется по формуле (3.104).

Для решения задачи (3.111) предлагается применить модифицированную процедуру целочисленного линейного поиска и гибридные мультиагентные алгоритмы, описанные в разд. 1.3 – 1.5.

Опишем пошаговый алгоритм поиска оптимального управления с неполной обратной связью, используя предложенный подход.

Шаг 1. *Задание параметров.* Выбрать мультиагентный алгоритм оптимизации и задать его параметры. Задать количество координат m, информация о которых используется в управлении, ограничение на максимальное значение масштаба усечения $L_{\max}$, ограничения $c_{\min}$ и $c_{\max}$ на значения коэффициентов в разложении (3.107), максимальное число проходов P. Указать нижнюю и верхнюю границы $\underline{x}_i, \overline{x}_i$ по каждой координате, а также множество начальных состояний $\Omega = [\alpha_1, \beta_1] \times ... \times [\alpha_n, \beta_n]$. Положить $it = 0$ (счетчик числа итераций), $p = 0$ (счетчик числа проходов).

Шаг 2. *Генерирование начального приближения.*

Для целочисленных переменных:

$$L_i^j = \mathrm{INT}\left[\mathrm{rand}[0;1]\ (L_{\max}^j + 1)\right],\ 0 \le L_i^j \le L_{\max},\ i = 0, ..., m;\ j = 1, ..., q,$$

где $\mathrm{rand}[0;1]$ – число, генерируемое согласно равномерному закону распределения, $\mathrm{INT}[.]$ – операция выделения целой части числа.

Для непрерывных переменных требуется сгенерировать начальную популяцию, состоящую из NP агентов, значения координат $c^j_{i_0, i_1, ..., i_m}$ которых определены на интервале $[c_{\min}, c_{\max}]$:

$$c^j_{i_0, i_1, ..., i_m} = c_{\min} + \mathrm{rand}[0;1](c_{\max} - c_{\min}),\ j = 1, ..., q;\ i_0 = 0, 1, ..., L_0^j - 1; ...;\ i_m = 0, 1, ..., L_m^j - 1.$$

Шаг 3. *Реализация процедуры поиска на множестве $\boldsymbol{L}$ по целочисленным переменным при известных значениях координат вектора C.* Значения координат вектора C в блочном векторе (L, C) считаются фиксированными.

Шаг 3.1. Упорядочить координаты вектора L случайным образом. Положить $New = false$ – индикатор наличия улучшения, $j = 1$.

Шаг 3.2. Пусть на j-м месте стоит r-я переменная. «Просканировать» множество решений вида $Z = L + ke_r, k = 0, \pm 1, \pm 2, ..., 0 \le Z \le L^{\max}$, где e_r – единичный орт размеров $(q(m+1) \times 1)$, составленный из всех нулей и одной единицы на r-м месте; знак неравенства понимается покоординатно, $L^{\max}$ – вектор размеров $(q(m+1) \times 1)$, все координаты которого равны $L_{\max}$.

Шаг 3.3. Если среди полученных в результате сканирования решений имеется наилучшее, удовлетворяющее условию $f(Z,C) < f(L,C)$, то заменить L на Z и положить $New = true$.

Шаг 3.4. Если $j < q(m+1)$, то положить $j = j+1$ и перейти к шагу 3.2. Если $j = q(m+1)$ и если $New = true$, то положить $New = false$, $j = 1$ и перейти к шагу 3.2. Иначе поиск завершить и положить $p = p+1$ (счетчик числа проходов).

Шаг 4. *Реализация процедуры поиска на множестве C по непрерывным переменным при фиксированных координатах вектора L.*

Шаг 4.1. Для каждого из NP агентов по сгенерированным коэффициентам, входящим в C, сформировать управление в виде функции насыщения sat, гарантирующей выполнение ограничений на управление:

$$\boldsymbol{u}_j(t,x^1) = \mathrm{sat}\left\{g_j\left(t,x_1,...,x_m\right)\right\},\ j = 1,...,q,$$

$$g_j\left(t,x_1,...,x_m\right) = \sum_{i_0=0}^{L_0^j-1}\sum_{i_1=0}^{L_1^j-1}\cdots\sum_{i_m=0}^{L_m^j-1} c^j_{i_0 i_1 \ldots i_m}\, q\left(i_0,t\right)p_1\left(i_1,x_1\right)\cdots p_m\left(i_m,x_m\right).$$

В качестве базисных функций $q\left(i_0,t\right)$, $p_k\left(i_k,x_k\right), k = 1,...,m$ возьмем ортонормированную на отрезке $\left[0;t_f\right]$ систему нестационарных косинусоид:

$$q(i_0,t) = \begin{cases} \sqrt{\dfrac{1}{t_f}}, & i_0 = 0, \\ \sqrt{\dfrac{2}{t_f}}\cos\dfrac{i_0\pi t}{t_f}, & i_0 = 1,...,L_0^j - 1, \end{cases}$$

$$p_k(i_k,x_k) = \begin{cases} \sqrt{\dfrac{1}{\overline{x}_k - \underline{x}_k}}, & i_k = 0, \\ \sqrt{\dfrac{2}{\overline{x}_k - \underline{x}_k}}\cos\dfrac{i_k\pi(x_k - \underline{x}_k)}{\overline{x}_k - \underline{x}_k}, & i_k = 1,...,L_k^j - 1. \end{cases}$$

Шаг 4.2. Проинтегрировать NP систем стохастических дифференциальных уравнений (3.98) с управлениями $\boldsymbol{u}^1(t,x^1),...,\boldsymbol{u}^{NP}(t,x^1)$, например, методом Эйлера–Маруямы для каждого начального состояния X_0^k, $k = 1,...,N$, полученного в соответствии с заданной плотностью вероятности $p_0(x)$. Для каждого агента получить семейство траекторий $(x_1^{1,X_0^k}(\cdot),...,x_n^{1,X_0^k}(\cdot)), k = 1,...,N;\ ...,(x_1^{NP,X_0^k}(\cdot),...,x_n^{NP,X_0^k}(\cdot)), k = 1,...,N$ и вычислить значения функционала $I^{1,X_0^k},...,I^{NP,X_0^k}$. $k = 1,...,N$. Найти значения критерия качества управления множеством траекторий по формуле (3.104): $J^1,...,J^{NP}$.

Шаг 4.3. Выполнить очередную итерацию выбранного мультиагентного метода минимизации функционала (3.104). Получить новые положения агентов $1',...,NP'$ (векторы значений коэффициентов разложения по системе базисных функций).

Шаг 4.4. Проверка критериев окончания поиска по непрерывным переменным.

Если $it < ITER$, то положить $it = it+1$ и перейти к шагу 4.1.

Если $it \geq ITER$, завершить процедуру поиска значений непрерывных переменных и перейти к шагу 5.

Шаг 5. *Завершение поиска.* Если $p < P$, перейти к шагу 3. Если $p = P$, поиск завершить. Выбрать в качестве приближенного решения наилучшего агента в популяции, т.е. вектор L с найденными значениями его координат и вектор C с координатами $c^J_{i_0,i_1,\ldots,i_m}$, $i_0 = 0,\ldots,L^J_0 - 1;\ldots;\ i_m = 0,\ldots,L^J_m - 1$, соответствующие ему управление и пучок траекторий, а также значение J функционала (3.104).

3.8.3. Модельные примеры

Пример 3.15. Постановка задачи, обобщающей рассмотренную в [61], приведена в табл. 3.41.

Таблица 3.41. Постановка задачи (пример 3.15)

Система стохастических дифференциальных уравнений	$\begin{cases} dX_1 = [X_2(t) + \sin X_1(t) + u(t)]\,dt + \sigma_1 dW_1(t), \\ dX_2 = [X_1 \cos X_2(t) u(t)]\,dt + \sigma_2 dW_2(t). \end{cases}$
Временной интервал	$t \in [0;1]$
Ограничение на управление	$-1 \le u \le 1$
Функционал (3.100)	$I = X_2(1)$

Начальные состояния описываются равномерным законом распределения на множестве $\Omega = [-0,05;0,05] \times [-0,05;0,05]$, $N_1 = N_2 = 5$. Параметры модели: $\sigma_1 = \sigma_2 = 0,01$. Использованы следующие параметры выбранного мультиагентного алгоритма оптимизации на основе применения обобщенных ПИД-регуляторов для управления движением агентов (разд. 1.5): $NP = 13$, $P_{\max} = 5$, $NMAX = 10$, $h = 0,001$, $K_P = 0,1$, $K_{D_1} = 10$, $K_{D_2} = 0,01$, $K_I = 10$. Наложенные ограничения на коэффициенты в разложении (3.107): $c_{\min} = -5$, $c_{\max} = 5$ и на масштабы усечения: $L_{\max} = 5$. Результаты решения примера приведены в табл. 3.42.

Таблица 3.42. Результаты решения примера 3.15

Закон управления	$u(t)$	$\boldsymbol{u}(t, x_1)$	$\boldsymbol{u}(t, x_2)$	$\boldsymbol{u}(t, x_1, x_2)$
Масштабы усечения	$L_0 = 2, L_1 = 0,$ $L_2 = 0$	$L_0 = 2, L_1 = 1,$ $L_2 = 0$	$L_0 = 2, L_1 = 0,$ $L_2 = 1$	$L_0 = 2, L_1 = 1,$ $L_2 = 1$
Интервалы значений координат $X_1(1), X_2(1)$	[–0,076; 0,523], [–0,25; –0,076]	[0,071; 0,658], [–0,34; –0,097]	[0,1; 0,61], [–0,26; –0,061]	[0,136; 0,516], [–0,22; –0,084]
Величина функционала (3.101)	– 0,14962	– 0,15981	– 0,15966	– 0,16169
Коэффициенты разложения (3.107)	– 0,03, 3,11	– 0,21; 2,63	– 0,28; 3,54	– 0,29; 2,72

В задаче минимизировалось среднее значение функционала (3.100). Решены задачи синтеза оптимального управления с полной и неполной обратной связью, включая случай $m = 0$. Для каждого случая определялось оптимальное количество коэф-

фициентов в разложении L_0, L_1, L_2 и значения коэффициентов. В качестве базисных функций использовалась система нестационарных косинусоид (3.108). Исходя из полученных численных результатов, лучшее значение критерия получается при управлении с полной обратной связью, наихудшее – при программном управлении. Множество реализаций траекторий и управления для случая $m = 2$ изображены на рис. 3.22. Затраченное время процессора на выполнение программы не превосходит 1 секунды.

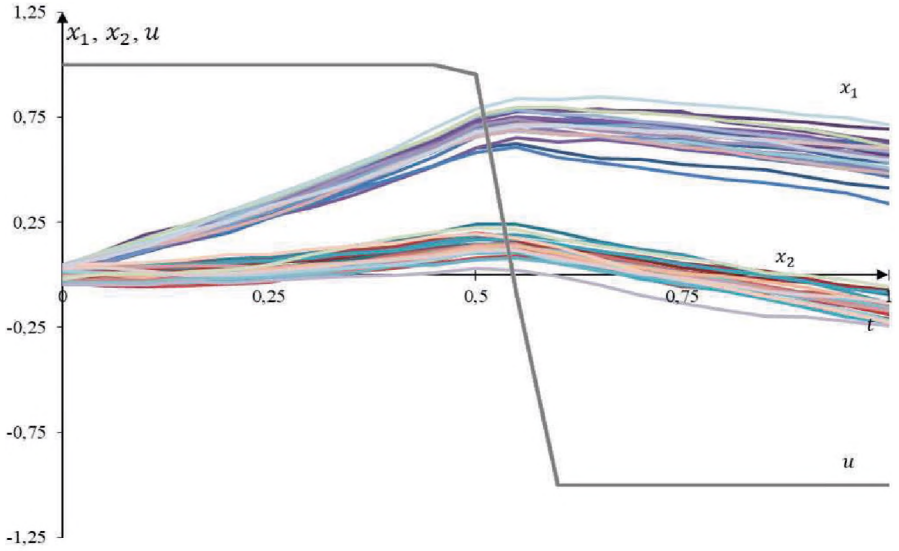


Рис. 3.22. Графики траекторий и управления в примере 3.15

Пример 3.16. Постановка задачи, обобщающей рассмотренную в [10], приведена в табл. 3.43.

Таблица 3.43. Постановка задачи (пример 3.16)

Система стохастических дифференциальных уравнений	$\begin{cases} dX_1 = [\dfrac{1}{\cos X_1(t) + 2} + 3\sin X_2(t) + u(t)]dt + \sigma_1 dW_1(t) \\ dX_2 = [X_1(t) + X_2(t) + u(t)]dt + \sigma_2 dW_2(t) \end{cases}$
Временной интервал	$t \in [0; 1,6]$
Ограничение на управление	$-2 \le u \le 1$
Функционал (3.100)	$I = -X_1(1,6) + \dfrac{1}{2} X_2(1,6)$

Начальные состояния описываются равномерным законом распределения на множестве $\Omega = [-0,05; 0,05] \times [-0,05; 0,05]$, $N_1 = N_2 = 5$, параметры модели $\sigma_1 = \sigma_2 = 0,03$. Использованы следующие параметры выбранного мультиагентного алгоритма оптимизации на основе применения обобщенных ПИД-регуляторов для управления движением агентов (разд. 1.5): $NP = 13$, $P_{\max} = 5$, $NMAX = 10$, $h = 0,001$, $K_P = 0,1$, $K_{D_1} = 10$, $K_{D_2} = 0,01$, $K_I = 10$. Наложенные ограничения на коэффициенты

в разложении (3.107): $c_{\min} = 0$, $c_{\max} = 350$ и на масштабы усечения: $L_{\max} = 5$. В задаче минимизировалось среднее значение функционала (3.100). Решены задачи синтеза оптимального управления с полной и неполной обратной связью, включая случай $m = 0$. Для каждого случая определялось наилучшее количество коэффициентов в разложении L_0, L_1, L_2 и значения коэффициентов. Результаты решения задачи приведены в табл. 3.44. Исходя из полученных численных результатов, лучшее значение критерия получается при управлении с полной обратной связью, наихудшее – при программном управлении. Множество реализаций траекторий и управления для случая $m = 2$ изображены на рис. 3.23. Затраченное время процессора на выполнение программы не превосходит 1 секунды.

Таблица 3.44. Результаты решения примера 3.16

Законы управления	$u(t)$	$\boldsymbol{u}(t, x_1)$	$\boldsymbol{u}(t, x_2)$	$\boldsymbol{u}(t, x_1, x_2)$
Масштабы усечения	$L_0 = 2, L_1 = 0,$ $L_2 = 0$	$L_0 = 2, L_1 = 1,$ $L_2 = 0$	$L_0 = 2, L_1 = 0,$ $L_2 = 1$	$L_0 = 2, L_1 = 1,$ $L_2 = 1$
Интервалы значений координат $X_1(1,6), X_2(1,6)$	[3,155; 3,784], [11,22; 13,131]	[3,249; 3,855], [10,722; 13,13]	[3,371; 4,096], [10,766;13,193]	[3,199; 3,895], [11,18; 13,492]
Величина функционала (3.101)	–2,49909	–2,52902	–2,55033	–2,59518
Коэффициенты разложения (3.107)	247,49; 247,49	107,25; 142,04	179,65; 238,44	160,03; 223

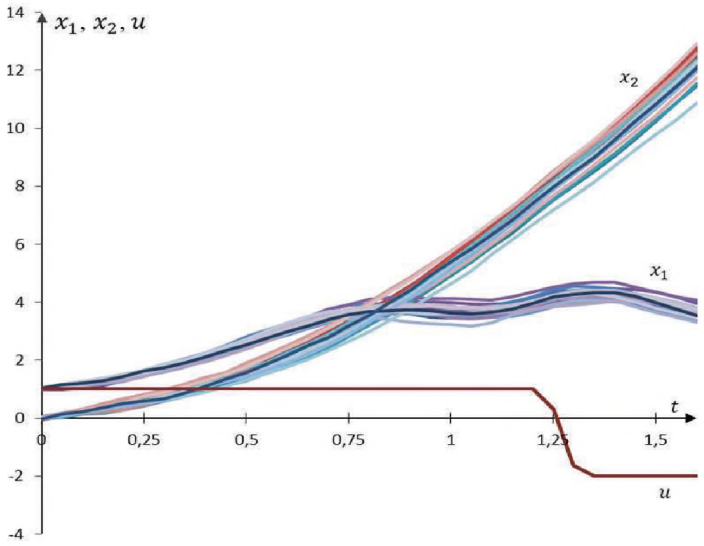


Рис. 3.23. Графики траекторий и управления в примере 3.16

Пример 3.17. Постановка задачи, обобщающей рассмотренную в [61], приведена в табл. 3.45.

Таблица 3.45. Постановка задачи (пример 3.17)

Система стохастических дифференциальных уравнений	$\begin{cases} dX_1 = [(X_2(t))^2 + u(t)]\,dt + \sigma_1 dW_1(t) \\ dX_2 = [8\sin X_1(t) + X_1(t) - X_2(t) + u(t)]\,dt + \sigma_2 dW_2(t) \end{cases}$
Временной интервал	$t \in [0;2]$
Ограничение на управление	$-2 \le u \le 1$
Функционал (3.100)	$I = -X_1(2)$

Начальные состояния описываются равномерным законом распределения на множестве $\Omega = [-1,05;-0,95] \times [-0,05;0,05]$, $N_1 = N_2 = 5$, параметры модели $\sigma_1 = \sigma_2 = 0,01$. Использованы следующие параметры алгоритма оптимизации: $NP = 13$, $P_{\max} = 5$, $NMAX = 10$, $h = 0,001$, $K_P = 0,1$, $K_{D_1} = 10$, $K_{D_2} = 0,01$, $K_I = 10$. Наложенные ограничения на коэффициенты в разложении (3.107): $c_{\min} = -50$, $c_{\max} = 50$. В задаче минимизировалось среднее значение функционала (3.100). Решены задачи синтеза оптимального управления с полной и неполной обратной связью, включая случай $m = 0$. Для каждого случая определялось наилучшее количество коэффициентов в разложении L_0, L_1, L_2 и значения коэффициентов. Результаты решения задачи приведены в табл. 3.46. Исходя из полученных численных результатов, лучшее значение критерия получается при управлении с полной обратной связью, наихудшее – при программном управлении. Множество реализаций траекторий и управления для случая $m = 2$ изображены на рис. 3.24. Затраченное время процессора на выполнение программы не превосходит 1 секунды.

Таблица 3.46. Результаты решения примера 3.17

Закон управления	$u(t)$	$\boldsymbol{u}(t, x_1)$	$\boldsymbol{u}(t, x_2)$	$\boldsymbol{u}(t, x_1, x_2)$
Масштабы усечения	$L_0 = 2, L_1 = 0,$ $L_2 = 0$	$L_0 = 2, L_1 = 2,$ $L_2 = 0$	$L_0 = 2, L_1 = 0,$ $L_2 = 1$	$L_0 = 2, L_1 = 2,$ $L_2 = 1$
Интервалы значений координат $X_1(2), X_2(2)$	[14,112; 18,65], [5,655; 6,967]	[16,65; 20,61], [6,772; 7,295]	[18,17; 22,48], [7,221; 8,363]	[18,797; 21,97], [6,909; 7,899]
Величина функционала (3.101)	–16,74973	–18,64712	–19,86348	–20,66234
Коэффициенты разложения (3.107)	–1,1; 10,48	0,01; 1,92; 8,67; 20,28	–25,8; –10,95; –5,96; –2,72	–3,55; –3,01; –2,39; 3,55

Оценка эффективности мультиагентного алгоритма на основе применения обобщенных ПИД-регуляторов для управления движением агентов основана на сравнительном анализе полученных значений функционала (3.101), траекторий движения системы и законов управления, затраченных вычислительных ресурсов. Приведенные при решении примеров 3.15–3.17 данные свидетельствуют о приемлемых затратах вычислительных ресурсов на выполнение алгоритма.

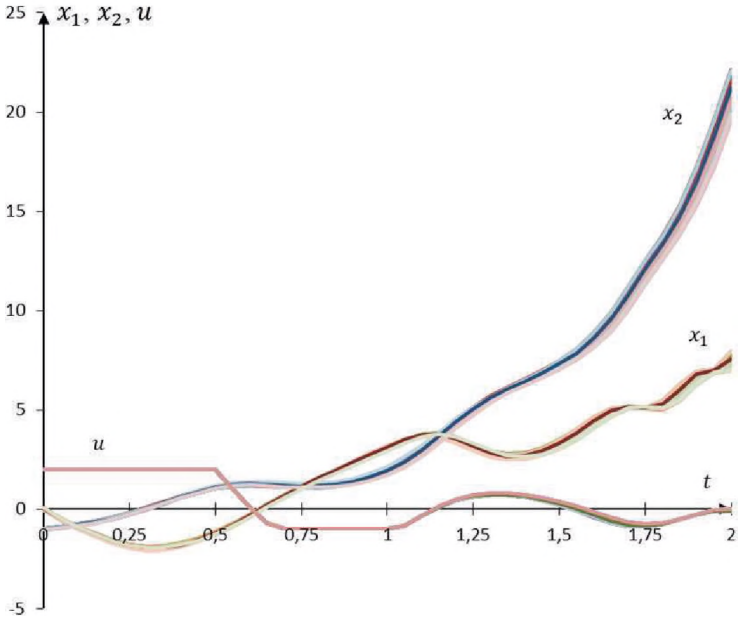


Рис. 3.24. Графики траекторий и управления в примере 3.17

Заметим, что постановки задач в примерах 3.12–3.14, решение которых изложено в разд. 3.7, отличаются от соответствующих формулировок в примерах 3.15–3.17 отсутствием случайных внешних воздействий. При этом имеется неопределенность задания начальных условий. Наличие случайных внешних воздействий и начальных состояний приводит к увеличению разброса значений координат вектора состояния, как в любой текущий, так и в заданный конечный момент времени, по сравнению со случаем управления пучками траекторий в условиях неопределенности начальных состояний (разд. 3.7). По мере увеличения числа координат вектора состояния, используемых в управлении, минимальное значение функционала последовательно уменьшается, что соответствует известным теоретическим результатам [87].

На основании результатов решения примеров 3.15–3.17 можно сделать вывод о том, что предложенный подход нахождению управления с неполной обратной связью на основе параметризации закона управления и решения смешанной целочисленно-непрерывной задачи условной оптимизации с применением гибридных мультиагентных алгоритмов позволяет получать решения достаточно высокого качества.

3.9. ПРИЛОЖЕНИЕ МУЛЬТИАГЕНТНЫХ МЕТОДОВ ОПТИМИЗАЦИИ В ЗАДАЧЕ СТАБИЛИЗАЦИИ СПУТНИКА

3.9.1. Постановка задачи

Рассматривается задача гашения вращательного движения спутника с помощью установленных на нем двигателей [23]. Описаны две постановки этой задачи: в первой начальное состояние системы фиксировано, а во второй задано множество возможных начальных состояний, что соответствует случаю априорной неопределенности. В первой задаче реализуется управление отдельной траекторией, а во второй – пучком траекторий, исходящим из заданного множества возможных начальных состояний. В обоих случаях найдено программное управление с использованием параметризации закона управления в виде разложения по выбранной системе базисных функций и дальнейшего применения мультиагентных алгоритмов оптимизации (разд. 1.3, 1.4). В качестве систем базисных функций использованы: финитные функции (кусочно-постоянные, кусочно-линейные, квадратичные, кубические сплайны (разд. 3.4) и многочлены Чебышева (разд. 3.6).

Задача 1 (оптимальное управление отдельной траекторией).

Движение твердого тела относительно центра инерции после перехода к безразмерным переменным, имеет вид:

$$\begin{aligned} \dot{p}(t) &= u_1(t)/6\ , \\ \dot{q}(t) &= u_2(t) - 0,2r(t)p(t), \\ \dot{r}(t) &= 0,2[u_3(t) + p(t)q(t)], \end{aligned} \tag{3.112}$$

где p, q, r – проекции угловой скорости на главные центральные оси инерции, а u_1, u_2, u_3 – управления, которые характеризуют тяги двигателей, расположенных на спутнике; $t \in T = [t_0, t_f]$ – промежуток времени функционирования системы. Множество допустимых значений управления $U = [-200; 200] \times [-200; 200] \times [-200; 200]$.

Начальное состояние системы задано в момент $t_0 = 0$:

$$p(t_0) = 24,\ q(t_0) = 16,\ r(t_0) = 16\,. \tag{3.113}$$

В момент окончания функционирования системы при $t_f = 1$ должны выполняться условия: $p(t_f) = q(t_f) = r(t_f) = 0$.

Функционал качества управления характеризует затраты топлива при работе реактивных двигателей (величина $I(x_0, d)$) и степень выполнения конечных условий:

$$I_1(x_0, d) = \underbrace{\int_{t_0}^{t_f} \left[|u_1(t)| + |u_2(t)| + |u_3(t)| \right] dt}_{I(x_0, d)} + R_1 p(t_f)^2 + R_2 q(t_f)^2 + R_3 r(t_f)^2, \tag{3.114}$$

где $R_i, i = 1, 2, 3$ – параметры штрафа.

Преобразуем задачу Больца (3.114) к задаче Майера

$$I_1(x_0, d) = s(t_f) + R_1 p(t_f)^2 + R_2 q(t_f)^2 + R_3 r(t_f)^2. \tag{3.115}$$

с помощью введения дополнительного дифференциального уравнения:

$$\dot{s}(t) = |u_1(t)| + |u_2(t)| + |u_3(t)|,\ s(t_0) = 0.$$

Требуется найти оптимальное управление и соответствующую ему траекторию, минимизирующие значение функционала (3.115) при заданных начальных условиях (3.113).

Задача 2 (оптимальное управление пучком траекторий).

Математическая модель объекта управления задана системой (3.112). Функционал качества управления отдельной траекторией имеет вид (3.115).

В отличии от задачи 1 вместо фиксированного начального условия (3.113) задано множество возможных начальных состояний:

$$\Omega = [23,85;24,15] \times [15,85;16,15] \times [15,85;16,15]. \qquad (3.116)$$

Качество управления пучком траекторий предлагается оценивать величиной функционала:

$$J[u(\cdot)] = \int_{\Omega} I_1(x_0, d)dx_0 \,/\, \text{mes}\,\Omega. \qquad (3.117)$$

Требуется найти оптимальное в среднем программное управление и соответствующий ему пучок траекторий, минимизирующие среднее значение функционала (3.115) на множестве начальных условий Ω, т.е. значение функционала (3.117).

3.9.2. Решение задачи оптимального управления отдельной траекторией

Рассмотрим решение задачи 1, поставленной в разд. 3.9.1, т.е. оптимального управления отдельной траекторией при фиксированном начальном условии (3.113). На основе описанного в разд. 3.4 алгоритма с использованием финитных базисных функций разработано программное обеспечение, позволяющее находить оптимальное программное управление для описанного класса задач. Результатом работы программы является значение критерия (3.115) и значения координат вектора состояния в конечный момент времени функционирования системы. Помимо этого вычисляются наилучшие значения коэффициентов в разложении по базисным функциям, найденные в результате поиска при помощи гибридного мультиагентного алгоритма интерполяционного поиска (разд. 1.3). По окончании поиска оптимального управления программное обеспечение позволяет изобразить графики траекторий и управления.

Задача решается при использовании разных типов сплайнов: кусочно-постоянного, кусочно-линейного, квадратичного и кубического. Далее проводится сравнительный анализ результатов, полученных в этих четырех случаях. Кроме того, решение сравнивается с найденным в [23] при помощи применения соотношений принципа максимума и численных методов их удовлетворения.

Как следует из постановки задачи, требуется задать значения коэффициентов штрафной функции, чтобы обеспечить выполнение конечных условий $p(t_1) = q(t_1) = r(t_1) = 0$ с достаточной точностью. В процессе решения задачи были подобраны следующие коэффициенты штрафа: $R_1 = R_2 = R_3 = 10000$. На коэффициенты в разложении по системе базисных функций наложены ограничения: для u_1 – $c_{\min} = -150$, $c_{\max} = -50$, для u_2 – $c_{\min} = -50$, $c_{\max} = 50$, для u_3 – $c_{\min} = -1$, $c_{\max} = 1$.

Рассмотрим четыре случая поиска оптимального программного управления, соответствующие выбору одной из финитных функций в качестве базисной.

Первый случай. В качестве системы базисных функций взят кусочно-постоянный сплайн ($p = 0$). Найдено наилучшее количество коэффициентов в разложении: $L_1 = 8$, $L_2 = 8$, $L_3 = 2$ для u_1, u_2 и u_3 соответственно. Определены наилучшие параметры гиб-

ридного мультиагентного алгоритма интерполяционного поиска: $NP = 40$, $I_{\max} = 400$, $M_1 = 2$, $M_2 = 5$, $PRT = 0,01$, $nstep = 5$, $b_2 = 8$. Результаты численного эксперимента приведены в табл. 3.47, а на рис. 3.25 изображены графики изменения координат p, q, r вектора состояния и координат u_1, u_2, u_3 вектора управления.

Таблица 3.47. Результаты работы алгоритма для кусочно-постоянного сплайна

Координаты точек $(p(t_f), q(t_f), r(t_f))$	(0,0004; – 0,0001; 0,0002)
Коэффициенты разложения для u_1	(–145,67; –134,49; –145,8; –146,74; –146,67; –147,08; –147,75; –133,25)
Коэффициенты разложения для u_2	(– 41,2; – 25,03; 1,07; 38,01; 24,78; 43,53; 32,16; 41,87)
Коэффициенты разложения для u_3	(0,03; 0,05)
Значение функционала $I(x_0, d)$	169,42002

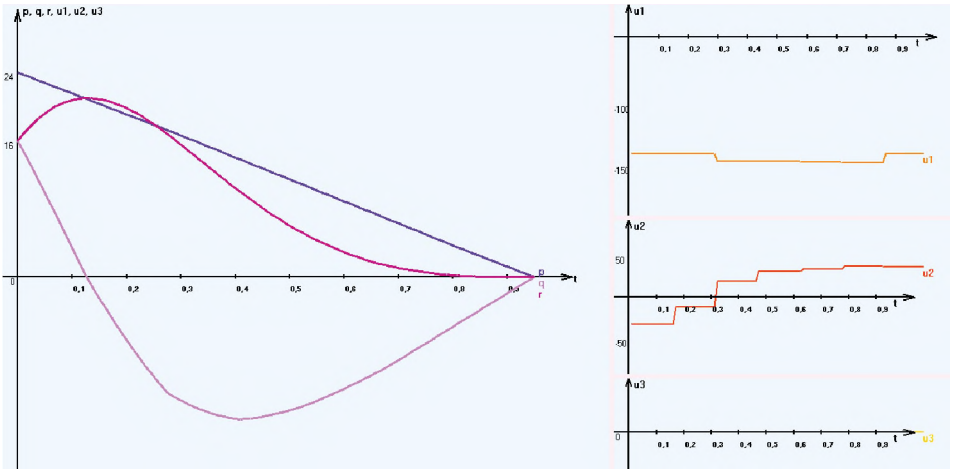


Рис. 3.25. Графики траектории и управления в случае кусочно-постоянного сплайна

Второй случай. В качестве системы базисных функций взят кусочно-линейный сплайн ($p = 1$). Найдено наилучшее количество коэффициентов в разложении: $L_1 = 8$, $L_2 = 8$, $L_3 = 2$ для u_1, u_2 и u_3 соответственно. Определены наилучшие параметры гибридного мультиагентного алгоритма интерполяционного поиска: $NP = 40$, $I_{\max} = 400$, $M_1 = 2$, $M_2 = 5$, $PRT = 0,01$, $nstep = 5$, $b_2 = 8$. Результаты численного эксперимента приведены в табл. 3.48, а на рис. 3.26 изображены графики изменения координат p, q, r вектора состояния и координат u_1, u_2, u_3 вектора управления.

Таблица 3.48. Результаты работы алгоритма для кусочно-линейного сплайна

Координаты точек $(p(t_f), q(t_f), r(t_f))$	(0,0052; 0,0002; – 0,0023)
Коэффициенты разложения для u_1	(–145,78; –127,67; –148,05; –144,31; –148,19; –147,76; –148,31; –141,14)
Коэффициенты разложения для u_2	(–28,8; –38,16; 10,5; 42,61; 20,38; 40,27; 35,26; 32,15)
Коэффициенты разложения для u_3	(0,09; 0,02)
Значение функционала $I(x_0, d)$	169,42325

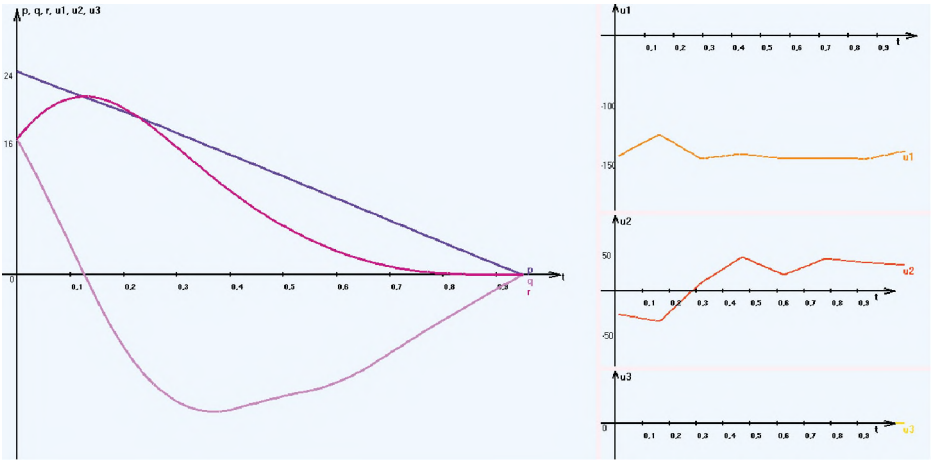


Рис. 3.26. Графики траектории и управления в случае кусочно-линейного сплайна

Третий случай. В качестве системы базисных функций взят квадратичный сплайн ($p = 2$). Найдено наилучшее количество коэффициентов в разложении: $L_1 = 4$, $L_2 = 3$, $L_3 = 2$ для u_1, u_2 и u_3 соответственно. Определены наилучшие параметры гибридного мультиагентного алгоритма интерполяционного поиска: $NP = 40$, $I_{\max} = 400$, $M_1 = 2$, $M_2 = 5$, $PRT = 0,01$, $nstep = 5$, $b_2 = 8$. Результаты численного эксперимента приведены в табл. 3.49, а на рис. 3.27 изображены графики изменения координат p, q, r вектора состояния и координат u_1, u_2, u_3 вектора управления.

Таблица 3.49. Результаты работы алгоритма для квадратичного сплайна

Координаты точек $(p(t_f), q(t_f), r(t_f))$	(0,014; – 0,0109; 0,0001)
Коэффициенты разложения для u_1	(–134,12; –143,29; –146,64; –149,48)
Коэффициенты разложения для u_2	(–36,252; 32,6207; 36,4115)
Коэффициенты разложения для u_3	(0,0372; 0,0096)
Значение функционала $I(x_0, d)$	171,81301

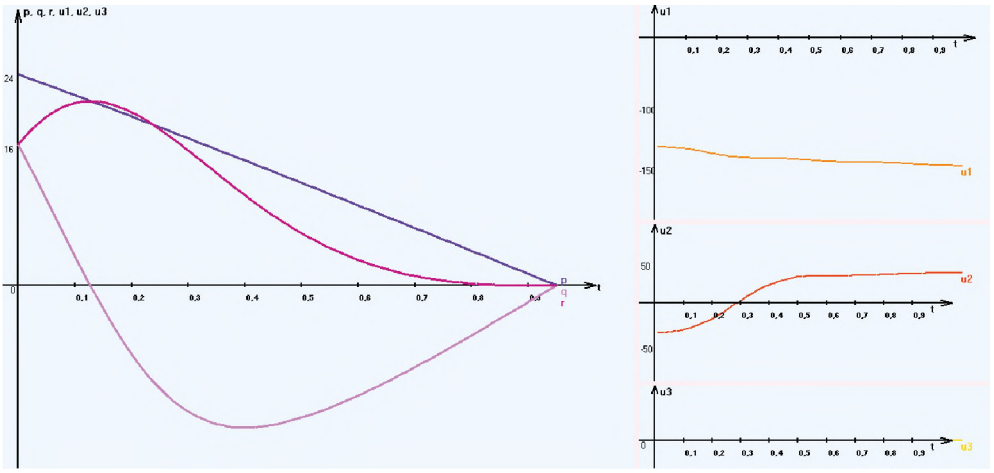


Рис. 3.27. Графики траектории и управления в случае квадратичного сплайна

Четвертый случай. В качестве системы базисных функций взят кубический сплайн ($p=3$). Найдено наилучшее количество коэффициентов в разложении: $L_1=4$, $L_2=4$, $L_3=2$ для u_1, u_2 и u_3 соответственно. Определены наилучшие параметры гибридного мультиагентного алгоритма интерполяционного поиска: $NP=40$, $I_{\max}=400$, $M_1=2$, $M_2=5$, $PRT=0{,}01$, $nstep=5$, $b_2=8$. Результаты численного эксперимента приведены в табл. 3.50, а на рис. 3.28 изображены графики изменения координат p,q,r вектора состояния и координат u_1,u_2,u_3 вектора управления.

Таблица 3.50. Результаты работы алгоритма для кубического сплайна

Координаты точек $(p(t_f),q(t_f),r(t_f))$	(0,0015; – 0,0009; 0,0007)
Коэффициенты разложения для u_1	(–143; –140,25; –146,93; –146,57)
Коэффициенты разложения для u_2	(– 47,58; 17,68; 33,48; 38,28)
Коэффициенты разложения для u_3	(0,06; 0,03)
Значение функционала $I(x_0,d)$	169,42035

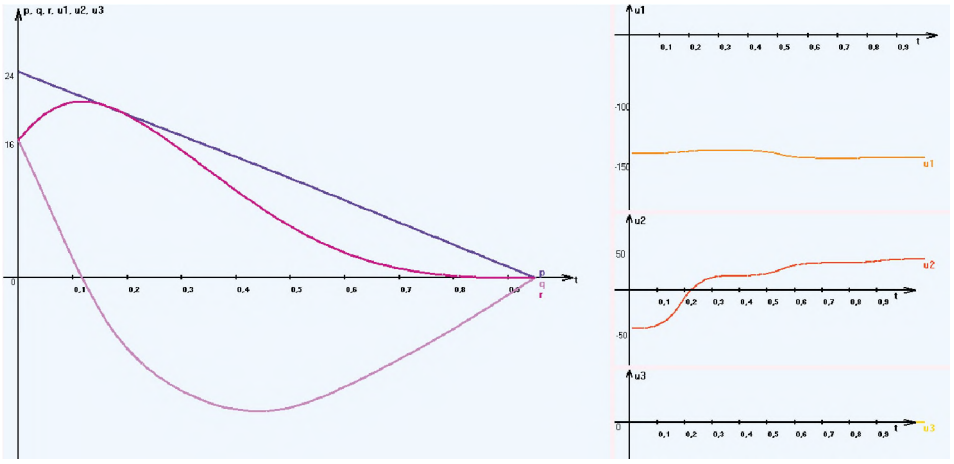


Рис. 3.28. Графики траектории и управления в случае кубического сплайна

В табл. 3.51 приведены вычислительные затраты и итоговые численные результаты, полученные в четырех случаях, в сравнении с известным решением, полученным в [23]. Результат из [23] представлен на рис. 3.29, а значение функционала $I=169{,}42$.

Таблица 3.51. Сравнение полученных результатов

Тип базисных функций	Значение функционала $I(x_0,d)$	Затраченное время
Кусочно-постоянные сплайны	169,42	2 мин. 33 с.
Кусочно-линейные сплайны	169,42	2 мин. 45 с.
Квадратичные сплайны	171,81	2 мин. 30 с.
Кубические сплайны	169,42	2 мин. 26 с.
Известное решение из [23]	169,42	–

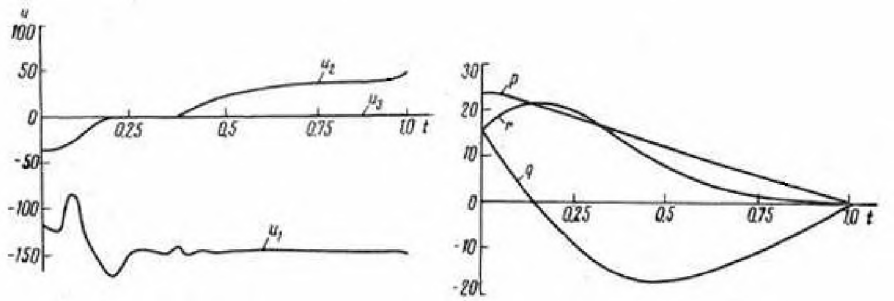


Рис. 3.29. Программное управление и соответствующая ему траектория из [23]

Во всех четырех случаях при использовании сплайнов алгоритм успешно справился с поиском оптимального программного управления для гашения вращательного движения спутника, что можно наблюдать как из вычисленных характеристик, приведенных в табл. 3.47 – 3.51, так и из графиков траекторий и управлений, изображенных на рис. 3.25 – 3.28. Подобранные базисные функции подошли для решения поставленной задачи, но можно заметить, что при использовании квадратичного сплайна значение минимизируемого функционала получилось хуже известного из [23]. В остальных же случаях вычисленное значение критерия близко к оптимальному при относительно небольших вычислительных затратах. Следовательно, описанный в разд. 3.4 алгоритм поиска оптимального программного управления на основе гибридного мультиагентного алгоритма интерполяционного поиска и финитных базисных функций может быть использован при решении аналогичных прикладных задач.

3.9.3. Решение задачи оптимального управления пучком траекторий

ПОИСК ОПТИМАЛЬНОГО В СРЕДНЕМ ПРОГРАММНОГО УПРАВЛЕНИЯ С ИСПОЛЬЗОВАНИЕМ ПОЛИНОМОВ ЧЕБЫШЕВА

Задача 2, поставленная в разд. 3.9.1, решается описанным в разд. 3.6 методом поиска оптимального программного управления пучками траекторий на основе псевдоспектрального метода. В связи с этим, сначала требуется привести исходную постановку задачи к типовому виду. Формулы преобразования времени ($t_0 = 0, t_f = 1$) имеют вид:

$$t = \frac{t_f - t_0}{2}\tau + \frac{t_f + t_0}{2} = \frac{1}{2}\tau + \frac{1}{2},\ dt = \frac{t_f - t_0}{2}d\tau = \frac{1}{2}d\tau,\ \tau \in [-1;1].$$

Тогда система (3.112) перепишется следующим образом (вместо новой переменной τ для упрощения обозначений будем по-прежнему использовать старую переменную текущего времени t, рассматриваемую на новом промежутке времени $[-1;1]$):

$$\dot{p}(t) = u_1(t)/12,$$
$$\dot{q}(t) = 0,5[u_2(t) - 0,2r(t)p(t)],$$
$$\dot{r}(t) = 0,1[u_3(t) + p(t)q(t)],$$

а начальный и конечный момент функционирования системы: $t_0 = -1,\ t_f = 1$.

Множество начальных состояний

$$\Omega = [23,85;24,15] \times [15,85;16,15] \times [15,85;16,15],$$

а параметры его разбиения на элементарные подмножества: $S_1 = S_2 = S_3 = 2$.

Зададим параметры гибридного мультиагентного алгоритма интерполяционного поиска: $NP = 40$, $I_{\max} = 100$, $M_1 = 8$, $M_2 = 1$, $PRTVector = 10$, $nstep = 5$, $b_2 = 20$, и метода, основанного на применении линейных регуляторов для управления движением агентов: $NP = 101$, $NMAX = 50$, $P_{\max} = 10$, $k_s = 0{,}1$, $k_l = 5$, $h = 0{,}0001$.

В табл. 3.52 приведены численные результаты, полученные двумя выбранными мультиагентными алгоритмами. Найдены векторы коэффициентов в разложении (3.81) для управлений $u_1(t)$, $u_2(t)$, $u_3(t)$. На коэффициенты разложения наложены ограничения: для u_1 – $c_{\min} = -160$, $c_{\max} = -50$, для u_2 – $c_{\min} = -60$, $c_{\max} = 60$, для u_3 – $c_{\min} = -1$, $c_{\max} = 1$. Наилучшие результаты (минимальное значение функционала) получены при использовании гибридного мультиагентного алгоритма интерполяционного поиска. Для этого случая изображены графики поведения пучка траекторий на рис. 3.30, а графики управления на рис. 3.31.

Таблица 3.52. Решение задачи о стабилизации спутника

Результаты	Гибридный мультиагентный алгоритм интерполяционного поиска	Мультиагентный алгоритм, основанный на применении линейных регуляторов для управления движением агентов
Отрезки значений координат $(p(t_f), q(t_f), r(t_f))$	[–0,0723; 0,0777], [–0,156; 0,1659], [–0,4662; 0,1472]	[–0,1378; 0,0122], [–0,3415; –0,0472], [–0,5527; 0,1354]
Параметры разложения (3.81) для $u_1(t)$, $u_2(t)$, $u_3(t)$	$L_0^{u_1} = 2, L_0^{u_2} = 3, L_0^{u_3} = 2$	$L_0^{u_1} = 2, L_0^{u_2} = 4, L_0^{u_3} = 2$
Векторы коэффициентов разложения для $u_1(t)$, $u_2(t)$, $u_3(t)$	(–159,59; –128,38), (17,69; 33,17; –54,73), (0,007; 0,0005)	(–149,4638; –139,2899), (18,3174; 50,594; –10,4979; –11,2656), (– 0,01; – 0,01)
Значение функционала J	181,300499	183,712969

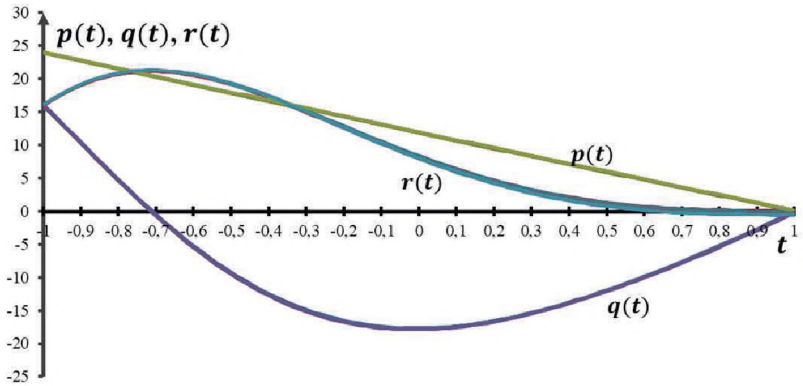


Рис. 3.30. Графики покоординатного поведения пучка траекторий

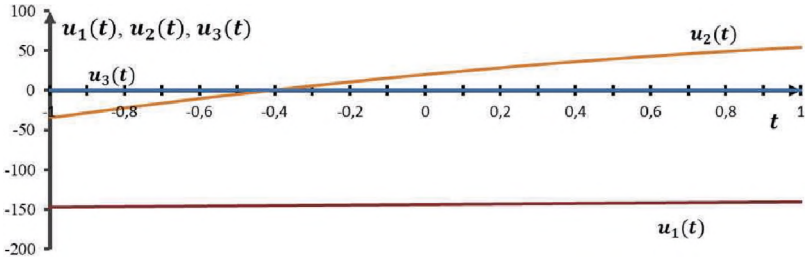


Рис. 3.31. График оптимального в среднем программного управления

В ходе решения задачи получены данные о вычислительных затратах (времени решения задачи). При решении гибридным мультиагентным алгоритмом интерполяционного поиска на выполнение задачи было затрачено 686 секунд, а алгоритмом, основанном на применении линейных регуляторов для управления движением агентов, 895 секунд.

Численный результат сравнивался с решением, полученным с помощью мультиагентного метода, имитирующего поведение стаи криля [21], входящего в качестве модуля в последовательно-параллельный гибридный метод, описанный в разд. 1.6, с параметрами: $NP = 40$, $I_{\max} = 300$, $\mu = 0{,}05$, $N_{\max} = 0{,}005$, $V_f = 0{,}001$, $D_{\max} = 0{,}0005$. Результат решения задачи этим алгоритмом при заданных параметрах приведен в табл. 3.53. Сравнивая этот результат с результатами из табл. 3.52, можно сделать вывод о том, что изложенные в разд. 3.6 алгоритмы успешно справились с задачей стабилизации спутника, а полученное значение функционала близко к наилучшему из известных.

Таблица 3.53. Решение задачи стабилизации спутника (метод стаи криля)

Результаты	Метод, имитирующий поведение стаи криля
Отрезки значений координат $(p(t_f), q(t_f), r(t_f))$	[0,025; 0,175], [–0,2545; 0,0499], [–0,5998; 0,0779]
Количество коэффициентов в разложении для $u_1(t)$, $u_2(t)$, $u_3(t)$	$L_0^{u_1} = 2, L_0^{u_2} = 3, L_0^{u_3} = 2$
Векторы коэффициентов разложения для $u_1(t)$, $u_2(t)$, $u_3(t)$	(–157,4442; –129,3557), (48,7378; 23,3138; –30,414), (0,0006; 0,0028)
Значение функционала J	180,346631

ПОИСК ОПТИМАЛЬНОГО В СРЕДНЕМ ПРОГРАММНОГО УПРАВЛЕНИЯ С ИСПОЛЬЗОВАНИЕМ ФИНИТНЫХ ФУНКЦИЙ

Задача 2, поставленная в разд. 3.9.1, была решена описанным в разд. 3.4 методом с использованием двух рассмотренных в разд. 1.3 и 1.4 мультиагентных алгоритмов оптимизации. Закон управления ищется в виде разложения по системе базисных функций, в качестве которых применялись кусочно-постоянный, кусочно-линейный, квадратичный и кубический сплайны. В каждом случае определялось наилучшее количество коэффициентов в разложении $L_0^{u_1}, L_0^{u_2}, L_0^{u_3}$ для каждой из координат u_1, u_2, u_3 вектора управления, а также значения коэффициентов разложения.

Для гибридного мультиагентного алгоритма, основанного на применении линейных регуляторов для управления движением агентов, были подобраны следующие параметры: $NP = 101$, $NMAX = 8$, $P_{\max} = 12$, $k_s = 1$, $k_l = 5$, $h = 0{,}0001$. Коэффициенты штрафной функции в (3.115): $R_1 = R_2 = R_3 = 1000$. На коэффициенты разложения наложены ограничения: для u_1 – $c_{\min} = -160$, $c_{\max} = -50$, для u_2 – $c_{\min} = -60$, $c_{\max} = 60$, для u_3 – $c_{\min} = -1$, $c_{\max} = 1$.

Результаты, полученные этим методом, приведены в табл. 3.54. На рис. 3.32 и рис. 3.33 представлены графики изменения значений координат вектора состояния и программного управления для случая кубического сплайна. Для остальных трех рассмотренных систем базисных функций графики выглядят аналогично.

Таблица 3.54. Результаты решения задачи о стабилизации спутника (мультиагентный алгоритм, основанный на использовании линейных регуляторов для управления движением агентов)

Тип сплайна	Отрезки значений координат $(p(t_f), q(t_f), r(t_f))$	Параметры разложения (3.49)	Коэффициенты разложения для $u_1(t)$, $u_2(t)$, $u_3(t)$	Значение функционала J
Кусочно-постоянный сплайн	[0,0782; 0,2282], [–0,1996; 0,1048], [–0,5772; 0,1065]	$L_0^{u_1} = 8$ $L_0^{u_2} = 8$ $L_0^{u_3} = 2$	(–132; –132; –148; –148; –148; –141,0195; –148; –141,242), (–15,9096; –1,8506; –50; 50; 40,9225; 49,8753; 25,0656; 50), (0,0077; 0,0949)	175,75168
Кусочно-линейный сплайн	[–0,0219; 0,1281], [–0,2079; 0,0823], [–0,6238; 0,0785]	$L_0^{u_1} = 4$ $L_0^{u_2} = 6$ $L_0^{u_3} = 2$	(–147,5634; –136,6439; –146,6189; –148), (–19,6645; –12,5656; 10,7914; 30,8425; 48,6036; 50), (0,1; 0)	170,38687
Квадратичный сплайн	[–0,0597; 0,0903], [–0,1974; 0,1157], [–0,5481; 0,0786]	$L_0^{u_1} = 3$ $L_0^{u_2} = 3$ $L_0^{u_3} = 2$	(– 144,3531; –141,6404; – 148), (– 42,588; 35,2651; 31,7664), (0,1; 0,0821)	173,72941
Кубический сплайн	[0,0079; 0,1579], [–0,2102; 0,097], [–0,5339; 0,0997]	$L_0^{u_1} = 2$ $L_0^{u_2} = 4$ $L_0^{u_3} = 2$	(– 140,7678; –146,2378), (– 49,9956; 19,9524; 28,9641; 43,4511), (0,0846; 0,1)	169,48431

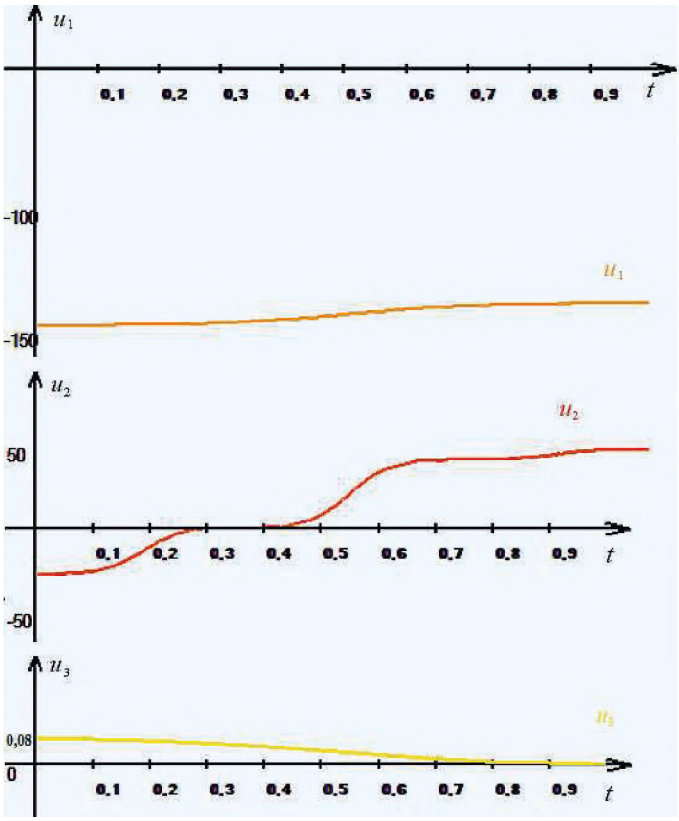


Рис. 3.32. Графики изменения координат $u_1(t)$, $u_2(t)$, $u_3(t)$ (кубический сплайн)

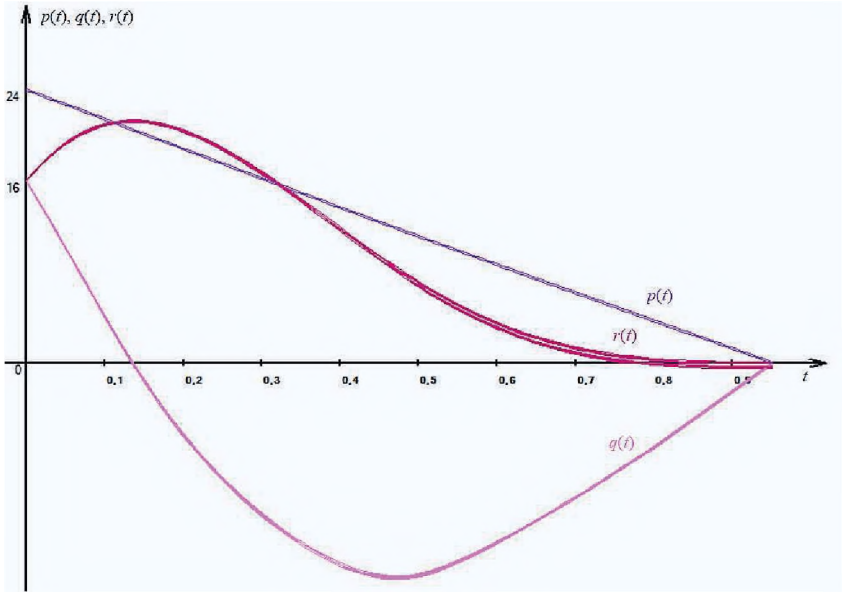


Рис. 3.33. Графики поведения пучка траекторий (кубический сплайн)

Для гибридного мультиагентного алгоритма интерполяционного поиска подобраны следующие параметры: $NP = 40$, $I_{\max} = 500$, $M_1 = 5$, $M_2 = 10$, $PRTVector = 1$, $nstep = 5$, $b_2 = 20$. Коэффициенты штрафной функции в (3.115): $R_1 = R_2 = R_3 = 1000$. Результаты, полученные этим методом, представлены в табл. 3.55.

Таблица 3.55. Результаты решения задачи о стабилизации спутника (гибридный мультиагентный алгоритм интерполяционного поиска)

Тип сплайна	Отрезки значений координат $(p(t_f), q(t_f), r(t_f))$	Параметры разложения (3.49)	Коэффициенты разложения для $u_1(t)$, $u_2(t)$, $u_3(t)$	Значение функционала J
Кусочно-постоянный сплайн	[0,1303; 0,2803], [–0,2345; 0,0645], [–0,5413; 0,1252]	$L_0^{u_1} = 6$ $L_0^{u_2} = 5$ $L_0^{u_3} = 2$	(–145,3921; –145,9129; –148; –137,2779; –138,4453; –135,8569), (–13,6532; –35,8449; 50; 50; 16,2456), (0,027; 0,0785)	187,32285
Кусочно-линейный сплайн	[0,0696; 0,2196], [–0,1709; 0,1221], [–0,5854; 0,0829]	$L_0^{u_1} = 5$ $L_0^{u_2} = 4$ $L_0^{u_3} = 2$	(–147,5928; –146,1051; –147,9969; –134,2685; –140,7262), (–44,5223; 11,5358; 34,9848; 47,5463), (0,0439; 0,098)	176,46869
Квадратичный сплайн	[–0,0062; 0,1438], [–0,2246; 0,0948], [–0,5577; 0,0438]	$L_0^{u_1} = 4$ $L_0^{u_2} = 3$ $L_0^{u_3} = 2$	(–141,2904; –146,5011; –145,2951; –136,6422), (–49,8131; 41,0978; 21,9337), (0,057; 0,0691)	173,18994
Кубический сплайн	[0,0192; 0,1692], [–0,2157; 0,1149], [–0,4574; 0,1184]	$L_0^{u_1} = 3$ $L_0^{u_2} = 3$ $L_0^{u_3} = 2$	(–137,7344; –146,6208; –142,7623), (–50; 43,5601; 15,245), (0,0002; 0)	170,13441

Данные о времени решения задачи приведены в табл. 3.56. Анализ результатов, представленных в табл. 3.54–3.55, позволяет сделать вывод о том, что применение гибридного мультиагентного алгоритма, основанного на использовании линейных регуляторов для управления движением агентов, приводит к лучшим результатам. При этом с ростом степени сплайна минимальное значение функционала уменьшается, за исключением квадратичного сплайна в случае гибридного мультиагентного алгоритма, основанного на применении линейных регуляторов для управления движением агентов. Полученные результаты сравнимы по точности и вычислительным затратам с решением методом, имитирующим поведение стаи криля (разд. 1.6) (табл. 3.57). При подсчете гибридным алгоритмом интерполяционного поиска результат достаточно высокого качества получается только при значительном числе вычислений целевой функции, что, однако приводит к росту затраченного времени процессора примерно на порядок по сравнению с другими рассмотренными мультиагентными методами оптимизации.

Таблица 3.56. Вычислительные затраты при решении задачи о стабилизации спутника

Метод	Тип сплайна	Значение функционала J	Затраченное время t, с
Гибридный алгоритм интерполяционного поиска	Кусочно-постоянный сплайн	187,32285	2819
	Кусочно-линейный сплайн	176,46869	1150
	Квадратичный сплайн	173,18994	1945
	Кубический сплайн	170,13441	1412
Мультиагентный алгоритм, основанный на использовании линейных регуляторов для управления движением агентов	Кусочно-постоянный сплайн	175,75168	130
	Кусочно-линейный сплайн	170,38687	99
	Квадратичный сплайн	173,72941	317
	Кубический сплайн	169,48431	306

Таблица 3.57. Результаты решения задачи о стабилизации спутника (метод, имитирующий поведение стаи криля)

Тип сплайна	Отрезки значений координат $(p(t_f), q(t_f), r(t_f))$	Параметры разложения (3.49)	Коэффициенты разложения для $u_1(t)$, $u_2(t)$, $u_3(t)$	Значение функционала J	Затраченное время t, с
Кусочно-постоянный сплайн	[–0,0007; 0,1493], [–0,2057; 0,0862], [–0,7656; -0,0485]	$L_0^{u_1} = 6$ $L_0^{u_2} = 6$ $L_0^{u_3} = 2$	(–142,63; –133,95; –145,94; –144,13; –145,14; –147,36), (15,64; –33,95; 7,26; 43,89; 47,95; 48,65), (0,07; 0,09)	191,26713	262
Кусочно-линейный сплайн	[0,0586; 0,2086], [–0,1992; 0,1074], [–0,5253; 0,1166]	$L_0^{u_1} = 6$ $L_0^{u_2} = 4$ $L_0^{u_3} = 2$	(–137,77; –144,66; –144,06; –142,31; –147,34; –137,43), (–46,52; 12,53; 43,75; 28,82), (0,07; 0,07)	173,2539	200
Квадратичный сплайн	[–0,0205; 0,1295], [–0,1988;0,1165], [–0,5284; 0,096]	$L_0^{u_1} = 4$ $L_0^{u_2} = 3$ $L_0^{u_3} = 2$	(–136,03; –144,01; –145,50; –146,97), (–41,23; 35,57; 31,16), (0,08; 0,02)	170,01691	205
Кубический сплайн	[–0,0018; 0,1482], [–0,2114; 0,0875], [–0,5573; 0,0986]	$L_0^{u_1} = 2$ $L_0^{u_2} = 4$ $L_0^{u_3} = 2$	(–142,5836; –144,5378), (–42,9706; 15,1389; 29,766; 49,0521), (0,0166; 0,0322)	169,4311	340

ГЛАВА 4
ПРИМЕНЕНИЕ МУЛЬТИАГЕНТНЫХ И БИОИНСПИРИРОВАННЫХ МЕТОДОВ ОПТИМИЗАЦИИ В ЗАДАЧАХ ПОИСКА ОПТИМАЛЬНОГО УПРАВЛЕНИЯ ДИСКРЕТНЫМИ ДИНАМИЧЕСКИМИ СИСТЕМАМИ

4.1. КЛАССИФИКАЦИЯ ПОСТАНОВОК ЗАДАЧ ОПТИМАЛЬНОГО УПРАВЛЕНИЯ ДИСКРЕТНЫМИ ДИНАМИЧЕСКИМИ СИСТЕМАМИ

Математические модели дискретных динамических систем управления описываются разностными уравнениями различных типов. Задачи оптимального управления сводятся к проблеме минимизации значений функционалов, определенных на их траекториях. Приведем типовые формулировки постановок задач в порядке возрастания сложности.

Задача 1 (оптимальное управление отдельной траекторией).

Случай А (оптимальное программное управление). Пусть поведение нелинейной детерминированной дискретной модели объекта управления описывается разностным уравнением

$$x(t+1) = f(t, x(t), u(t)),\ t = 0, 1, ..., N-1, \tag{4.1}$$

где t – дискретное время, $t \in T = \{0, 1, ..., N\}$; x – вектор состояния системы, $x \in R^n$; u – вектор управления, $u \in U \subseteq R^q$; U – множество допустимых значений управления, представляющее собой прямое произведение отрезков $[a_i, b_i]$, $i = 1, 2, ..., q$; $f(t, x, u) = (f_1(t, x, u), ..., f_n(t, x, u))^T$ – непрерывная вектор-функция; число шагов дискретного времени N задано. Правый конец траектории $x(N)$ свободен.

Начальное состояние задано:

$$x(0) = x_0. \tag{4.2}$$

Предполагается, что при управлении используется информация только о дискретном времени t, т.е. применяется программное управление, а, следовательно, система управления является разомкнутой по вектору состояния (рис. 4.1).

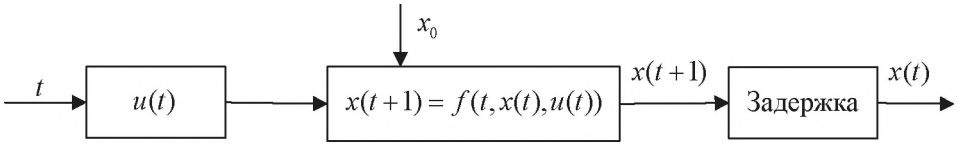


Рис. 4.1

Последовательность векторов $u(\cdot) = \{u(0), u(1), \ldots, u(N-1)\}$ называется управлением, а последовательность $x(\cdot) = \{x_0, x(1), \ldots, x(N)\}$, определяемая решением уравнения (4.1) с начальным условием (4.2) и управлением $u(\cdot)$, – траекторией.

Множество допустимых управлений $\mathcal{U}_0$ образуют управления вида $u(\cdot) = \{u(0), u(1), ..., u(N-1)\}$, где $u(t) \in U$, $t = 0, 1, \ldots, N-1$.

Множество допустимых процессов $\mathcal{D}(0, x_0)$ – это множество пар $d = (x(\cdot), u(\cdot))$, включающих траекторию $x(\cdot) = \{x_0, x(1), ..., x(N)\}$ и допустимое управление $u(\cdot) = \{u(0), u(1), ..., u(N-1)\} \in \mathcal{U}_0$, удовлетворяющих уравнению состояния (4.1) и начальному условию (4.2).

На множестве $\mathcal{D}(0, x_0)$ определен функционал качества управления

$$I(d) = \sum_{t=0}^{N-1} f^0\left(t, x(t), u(t)\right) + F\left(x(N)\right) \tag{4.3}$$

или

$$I(d) = F(x(0), ..., x(N); u(0), ..., u(N-1)), \tag{4.4}$$

где $f^0\left(t, x, u\right)$, $F\left(x\right)$, $F(x(\cdot), u(\cdot))$ – заданные непрерывные функции.

Требуется найти такую пару $d^* = \left(x^*(\cdot), u^*(\cdot)\right) \in \mathcal{D}(0, x_0)$, что

$$I(d^*) = \min_{d \in \mathcal{D}(0, x_0)} I(d) \tag{4.5}$$

Искомые в задаче (4.5) траектория $x^*(\cdot) = \{x_0, x^*(1), ..., x^*(N)\}$ и управление $u^*(\cdot) = \{u^*(0), u^*(1), ..., u^*(N-1)\}$ (элементы пары d^*) называются соответственно оптимальной траекторией и оптимальным управлением.

Если любому допустимому управлению $u(\cdot)$ соответствует единственная пара d, то задача (4.5) может быть переписана в эквивалентной форме как задача поиска минимума функционала на множестве допустимых управлений.

Задача поиска максимума функционала может быть сведена к задаче поиска минимума путем замены знака перед функционалом на противоположный. Функционал (4.4) является более общим по сравнению с (4.3) и его применение характерно для так называемых несепарабельных задач.

Случай Б (оптимальное управление с полной обратной связью). Пусть поведение модели объекта управления описывается разностным уравнением (4.1), Начальное состояние системы $x(0) = x_0 \in R^n$ заранее не задано и может быть любым. На множестве допустимых процессов $\mathcal{D}(0, x_0)$ определен функционал качества управления (4.3) или (4.4).

Предполагается, что при управлении используется информация о времени t и обо всех координатах вектора состояния x.

Множество $\mathcal{U}_n$ допустимых управлений с полной обратной связью образуют функции $\boldsymbol{u}(t, x) : T \times R^n \to U$, которые на траекториях системы (4.1) для различных начальных состояний (4.2) порождают соответствующие допустимые программные управления $u(\cdot) \in \mathcal{U}_0$, где $\forall\ t \in T \quad u(t) = \boldsymbol{u}\left(t, x(t)\right) \in U$.

Применяемое в каждый дискретный момент времени $t \in T$ управление имеет вид управления с полной обратной связью по вектору состояния (рис. 4.2).

Требуется найти такую функцию $\boldsymbol{u}^*\left(t, x\right) \in \mathcal{U}_n$, что

$$I\left(d^*\right) = \min_{d \in \mathcal{D}(0, x_0)} I\left(d\right) \qquad \forall x_0 \in R^n, \tag{4.6}$$

где $d^* = \left(x^*(\cdot),\ u^*(\cdot) = \boldsymbol{u}^*(\cdot, x^*(\cdot))\right)$.

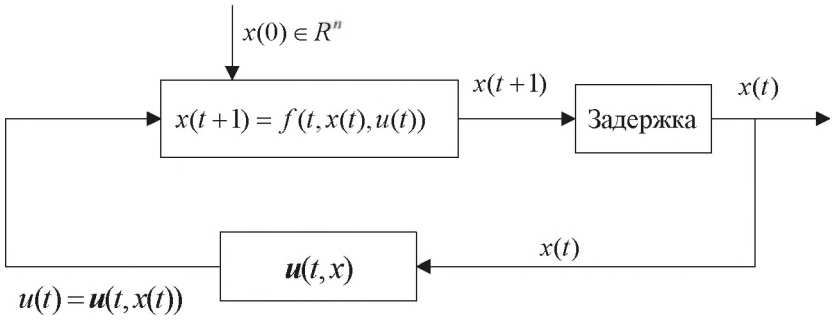


Рис. 4.2

Функция $\boldsymbol{u}^*(t,x) \in \mathcal{U}_n$ называется оптимальным управлением с полной обратной связью. Для каждого начального состояния x_0 из множества R^n она порождает соответствующую оптимальную пару, т.е. оптимальную траекторию $x^*(\cdot)$ и оптимальное программное управление $u^*(\cdot) \in \mathcal{U}_0$.

Случай B (оптимальное управление с неполной обратной связью). Пусть поведение модели объекта управления описывается уравнением (4.1). Вектор состояния системы управления представляется в виде $x = (x^1, x^2)^T$, $x^1 = (x_1, ..., x_m)^T \in R^m$; $x^2 = (x_{m+1}, ..., x_n)^T \in R^{n-m}$, $0 \le m \le n$. Предположим, что при управлении используется информация о дискретном времени и первых m координатах вектора состояния, т.е. о векторе x^1. Вектор x^1 содержит все используемые в управлении координаты, а вектор x^2 – неиспользуемые.

На множестве допустимых процессов $\mathcal{D}(0, x_0)$ определен функционал качества управления (4.3) или (4.4).

Начальное состояние x_0 принадлежит m-мерному многообразию Ω:

$$x_0 \in \Omega = \left\{ x \,\middle|\, x^2 = y_0(x^1),\ x^1 \in R^m \right\}, \tag{4.7}$$

где $y_0(x^1) = \left\{ y_{m+1}(x^1), ..., y_n(x^1) \right\}^T$ – заданная непрерывная вектор-функция.

Рассмотрим класс $\mathcal{U}_m$ функций $\boldsymbol{u}(t, x^1) : T \times R^m \to U$, которые на траекториях системы (4.1) порождают последовательности из множества $\mathcal{U}_0$ допустимых программных управлений: $u(\cdot) = \left\{ u(t) = \boldsymbol{u}(t, x^1(t)) \right\}_{t=0}^{N-1}$. Применяемое в каждый дискретный момент времени $t \in T$ управление имеет вид управления с обратной связью по неполному вектору состояния, т.е. с неполной обратной связью (рис. 4.3).

Требуется найти такую функцию $\boldsymbol{u}^*(t, x^1) \in \mathcal{U}_m$, что

$$I(d^*) = \min_{d \in \mathcal{D}(0, x_0)} I(d) \quad \forall x_0 \in \Omega, \tag{4.8}$$

где $d^* = \left\{ x^*(\cdot) = \left\{ x^*(t) \right\}_{t=0}^{N},\ u^*(\cdot) = \left\{ \boldsymbol{u}^*(t, x^{1*}(t)) \right\}_{t=0}^{N-1} \right\}$.

Подчеркнем, что число используемых в управлении координат вектора состояния совпадает с размерностью множества начальных условий Ω. При $m = 0$ множество Ω

является точкой x_0, для которой ищется оптимальное программное управление $u^*(\cdot)$, а при $m = n$ множество Ω совпадает с n-мерным пространством состояний и ищется оптимальное управление $\boldsymbol{u}^*(t, x)$ с полной обратной связью по вектору состояния.

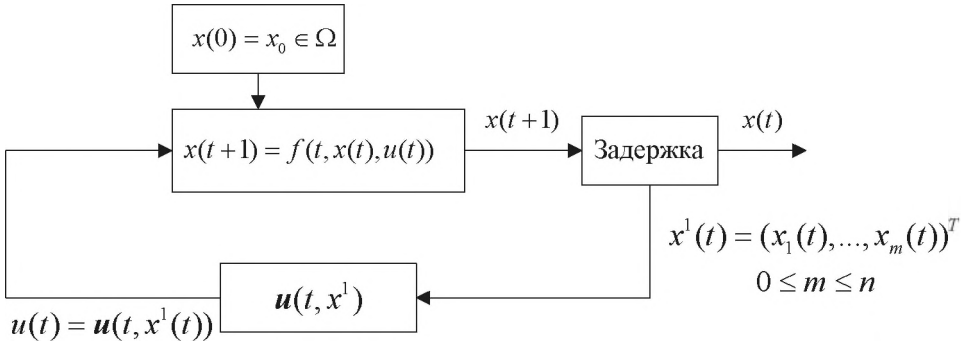


Рис. 4.3.

Задача 2 (оптимальное управление пучками траекторий)**.**

Пусть поведение нелинейной детерминированной дискретной модели объекта управления описывается разностным уравнением (4.1). Вектор состояния системы представляется в виде $x = (x^1, x^2)^T, x^1 = (x_1, \ldots, x_m)^T \in R^m$; $x^2 = (x_{m+1}, \ldots, x_n)^T \in R^{n-m}$, $0 \le m \le n$. Предположим, что при управлении используется информация о дискретном времени и первых m координатах вектора состояния, т.е. о векторе x^1. Вектор x^1 содержит все используемые в управлении координаты, а вектор x^2 – неиспользуемые.

На множестве допустимых процессов $\mathcal{D}(0, x_0)$ определен функционал качества управления (4.3) или (4.4).

Начальное состояние задано компактным множеством положительной меры:

$$x(0) = x_0 \in \Omega \subset R^n. \tag{4.9}$$

Предполагается, что при управлении используется информация только о времени t и о координатах вектора x^1, т.е. управление $u(t)$, применяемое в каждый момент времени $t \in T$, имеет вид управления $u(t) = \boldsymbol{u}(t, x^1(t))$ с неполной обратной связью по вектору состояния (см. рис. 4.3). Если $m = 0$, система управления будет разомкнутой по состоянию (см. рис. 4.1), а соответствующее управление $u(t)$ – программным, а если $m = n$, то система управления будет замкнутой с полной обратной связью, определяемой управлением $\boldsymbol{u}(t, x)$ (см. рис. 4.2).

Множество допустимых управлений $\mathcal{U}_m$ образуют такие функции $\boldsymbol{u}(t, x^1)$, что порождаемое управление удовлетворяет условию $u(t) = \boldsymbol{u}(t, x^1(t)) \in U \ \forall t \in T$.

Каждому допустимому управлению $\boldsymbol{u}(t, x^1) \in \mathcal{U}_m$ и множеству Ω поставим в соответствие пучок траекторий системы уравнений (4.1):

$$X(t, u(\cdot)) = \cup \{ x(t, u(\cdot), x_0) \mid x_0 \in \Omega \}, \tag{4.10}$$

т.е. объединение решений $x(t, u(\cdot), x_0)$ системы уравнений (4.1) по всем возможным начальным состояниям (4.9).

Качество управления пучком траекторий предлагается оценивать величиной функционала

$$J[\boldsymbol{u}(t,x^1)] = \int_{\Omega} I(x_0,d)\,dx_0 / \operatorname{mes}\Omega\ , \tag{4.11}$$

где $\operatorname{mes}\Omega$ – мера множества Ω.

Требуется найти такое управление $\boldsymbol{u}^*(t,x^1) \in \mathcal{U}_m$ что

$$J\left[\boldsymbol{u}^*(t,x^1)\right] = \min_{\boldsymbol{u}(t,x^1)\in\mathcal{U}_m} J\left[\boldsymbol{u}(t,x^1)\right]. \tag{4.12}$$

Искомое управление называется оптимальным в среднем, поскольку минимизируется среднее значение функционала (4.3) или (4.4) на множестве начальных состояний Ω.

Задача 3 (оптимальное управление стохастическими системами).

Пусть поведение модели объекта управления описывается уравнением

$$X(t+1) = f(t,X(t),u(t),W(t)),\ \ t = 0,1,\ldots,N-1, \tag{4.13}$$

где X – вектор состояния системы, $X \in R^n$, R^n – n-мерное евклидово пространство; u – вектор управления, $u \in U \subseteq R^q$, U – множество допустимых значений управления, представляющее собой прямое произведение отрезков $[a_i,b_i]$, $i = 1, 2,\ldots,q$; t – дискретное время, $t \in T = \{0,1,\ldots,N-1\}$, число шагов N задано; $f(t,x,u,w)\colon\ T\times R^n\times U\times R^m \to R^n$ – непрерывная вектор-функция; $W(t)$, $t = 0,1,\ldots,N-1$, – m-мерный случайный вектор с известным законом распределения, описывающий случайные внешние воздействия на объект управления.

Начальное состояние системы (4.13)

$$X(0) = X_0 \tag{4.14}$$

является случайным вектором с заданным законом распределения.

Вектор состояния системы представляется в виде $X = (X^1, X^2)^T$, $X^1 = (X_1,\ldots,X_m)^T \in R^m$; $X^2 = (X_{m+1},\ldots,X_n)^T \in R^{n-m}$, $0 \le m \le n$. Предположим, что при управлении используется информация о дискретном времени и первых m координатах вектора состояния, т.е. управление, применяемое в каждый момент времени $t \in T$, имеет вид управления с неполной обратной связью $u(t) = \boldsymbol{u}(t,X^1(t))$ (рис 4.4).

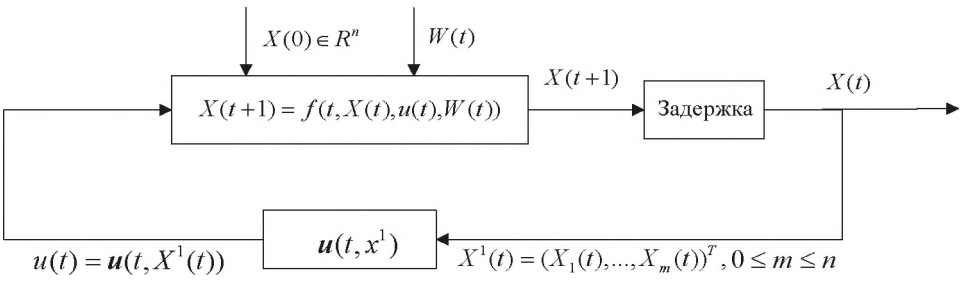


Рис. 4.4

Множество допустимых управлений с неполной обратной связью $\mathcal{U}_m$ образуют функции $\boldsymbol{u}(t,x^1): T \times R^m \to U$ такие, что для всех $t \in T$ справедливо $u(t) = \boldsymbol{u}(t, X^1(t)) \in U$.

Число m, $0 \le m \le n$, определяется условиями информированности. При $m = n$ имеется информация обо всех координатах вектора X, т.е. система (см. рис. 4.4) будет системой с полной обратной связью (рис. 4.5, *а*), а при $m = 0$ – системой, разомкнутой по состоянию (рис. 4.5, *б*). В последнем случае рассматривается так называемое программное управление $u(t)$.

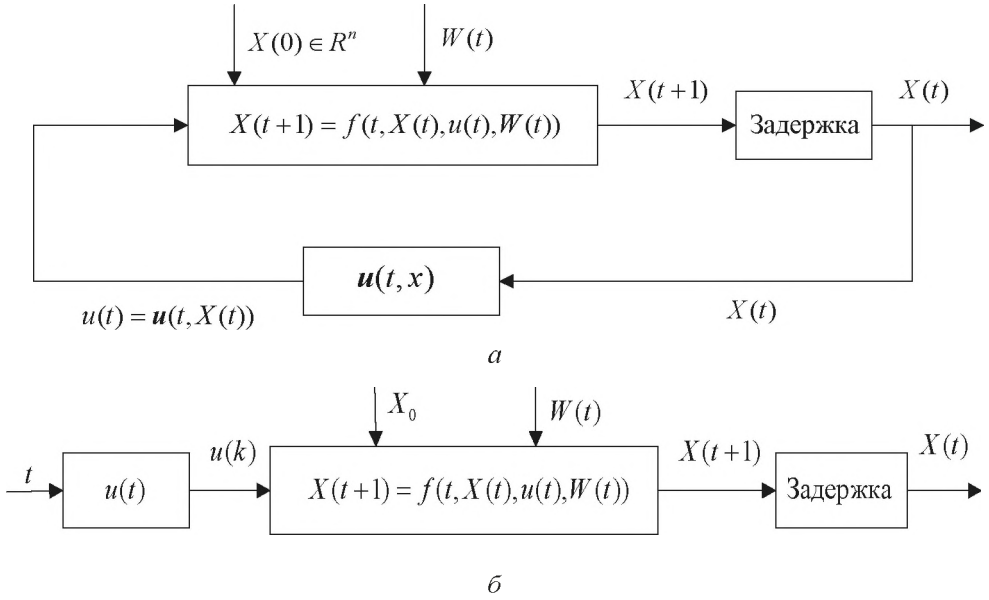


Рис. 4.5

Определим функционал качества управления

$$J[\boldsymbol{u}(t,x^1)] = M\left\{\sum_{t=0}^{N-1} f^0(t, X(t), \boldsymbol{u}(t, X^1(t))) + F(X(N))\right\}, \tag{4.15}$$

где $f^0(t,x,u)$, $F(x)$ – заданные непрерывные функции, M – знак математического ожидания, причем осреднение производится по множеству реализаций случайного процесса $X(t)$, порожденного случайными векторами $X(0)$ и $W(t)$, $t = 0,1,\ldots,N-1$ при допустимом управлении $\boldsymbol{u}(t,x^1)$.

Требуется найти управление $\boldsymbol{u}^*(t,x^1) \in \mathcal{U}_m$, минимизирующее величину функционала (4.15):

$$J[\boldsymbol{u}^*(t,x^1)] = \min_{\boldsymbol{u}(t,x^1) \in \mathcal{U}_m} J[\boldsymbol{u}(t,x^1)]. \tag{4.16}$$

Искомое управление называется оптимальным в среднем, поскольку минимизируется среднее значение функционала, определенного на траекториях динамической системы. При $m = n$, т.е. если имеется информация обо всех координатах вектора X, формула (4.16) соответствует формулировке задачи поиска оптимального управления с полной обратной связью, а при $m = 0$ – формулировке поиска оптимального программного управления.

4.2. СПОСОБЫ ПАРАМЕТРИЗАЦИИ ЗАКОНОВ УПРАВЛЕНИЯ

Рассмотренные в параграфе 4.1 постановки задач оптимального управления связаны с нахождением закона управления, зависящего от времени и от заданного набора координат вектора состояния, доступных измерению.

Для численного решения задачи требуется задать структуру искомого закона управления, зависящую от неизвестных параметров. Таким образом, реализуется переход от поставленных вариационных задач к проблеме конечномерной оптимизации, т.е. проблеме поиска наилучших значений параметров, задающих структуру управления. Для того, чтобы задать структуру управления, рекомендуется предварительно воспользоваться необходимыми или достаточными условиями оптимальности и связанными с ними соотношениями.

При решении задачи 1 используются: дискретный принцип максимума для частного случая динамических систем или обобщающие его необходимые условия оптимальности [57] для поиска программного управления, уравнение Беллмана [4, 11, 22] для поиска управления с полной обратной связью, а при решении задачи 2 – необходимые условия оптимальности [31, 32]. Для решения задачи 3 применяются: стохастический принцип максимума [40] для поиска программного управления, уравнение Беллмана для стохастических систем [4, 40, 116] для нахождения управления с полной обратной связью.

З а м е ч а н и я.

1. В задачах с параллелепипедными ограничениями на управление структура оптимального управления, как следует из необходимых или достаточных условий оптимальности, часто является релейной. Поэтому желательно задавать ее так, чтобы ограничения на управление удовлетворялись автоматически.

2. Для численного решения проблемы поиска оптимального программного управления $u(\cdot) = \{u(0), u(1), ..., u(N-1)\} \in \mathcal{U}_0$ можно использовать различные процедуры улучшения начального приближения в пространстве управлений с применением метаэвристических методов глобальной оптимизации.

В общем случае закон управления $\boldsymbol{u}(t, x^1)$ предлагается искать в параметрическом виде, определяемом числом коэффициентов в разложении управления по выбираемой системе базисных функций и их значениями (см. параграф 3.2).

Могут применяться следующие системы базисных функций.

1. Ортонормированные системы базисных функций, получившие широкое распространение в спектральном методе [50, 51].

2. Финитные базисные системы функций.

3. Радиально-базисные функции [73, 146].

4. Многочлены Чебышёва, используемые в различных вариантах псевдоспектрального метода [70, 90, 91].

При всех описанных в параграфе 3.2 способах параметризации закона управления предлагается искать значения координат блочных расширенных векторов вида (3.21), (3.26), (3.34), содержащих блок целочисленных переменных и блок действительных переменных, на которые наложены интервальные ограничения. Поэтому применяемый метод оптимизации должен реализовывать поиск как в множестве допустимых значений дискретных переменных, так и в соответствующем множестве допустимых значений непрерывных переменных. Эти два процесса могут быть реализованы как параллельно, так и последовательно в итерационном режиме.

4.3. ПОИСК ОПТИМАЛЬНОГО ПРОГРАММНОГО УПРАВЛЕНИЯ ДИСКРЕТНЫМИ ДЕТЕРМИНИРОВАННЫМИ ДИНАМИЧЕСКИМИ СИСТЕМАМИ

4.3.1. Постановка задачи

Поведение нелинейной детерминированной дискретной модели объекта управления описывается разностным уравнением

$$x(t+1) = f(t, x(t), u(t)),\ t = 0,1,...,N-1, \tag{4.17}$$

где t – дискретное время, $t \in \{0,1,...,N\}$; x – вектор состояния системы, $x \in R^n$; u – вектор управления, $u \in U \subseteq R^q$; U – множество допустимых значений управления, представляющее собой прямое произведение отрезков $[a_i, b_i]$, $i = 1, 2, ..., q$; $f(t,x,u) = (f_1(t,x,u),..., f_n(t,x,u))^T$ – непрерывная вектор-функция; число шагов дискретного времени N задано. Правый конец траектории $x(N)$ свободен.

Начальное состояние задано:

$$x(0) = x_0. \tag{4.18}$$

Предполагается, что при управлении используется информация только о дискретном времени t, т.е. применяется программное управление.

Множество допустимых процессов $\mathcal{D}(0, x_0)$ – это множество пар $d = (x(\cdot), u(\cdot))$, включающих траекторию $x(\cdot) = \{x_0, x(1),..., x(N)\}$ и допустимое управление $u(\cdot) = \{u(0), u(1),..., u(N-1)\}$, где $u(t) \in U$, удовлетворяющих уравнению состояния (4.17) и начальному условию (4.18).

На множестве $\mathcal{D}(0, x_0)$ определен функционал качества управления

$$I(d) = \sum_{k=0}^{N-1} f^0\left(k, x(k), u(k)\right) + F\left(x(N)\right) \tag{4.19}$$

или

$$I(d) = F(x(0),..., x(N); u(0),..., u(N-1)), \tag{4.20}$$

где $f^0\left(t,x,u\right)$, $F(x)$, $F(x(\cdot), u(\cdot))$ – заданные непрерывные функции.

Требуется найти такую пару $d^* = \left(x^*(\cdot), u^*(\cdot)\right) \in \mathcal{D}(0, x_0)$, что

$$I(d^*) = \min_{d \in \mathcal{D}(0, x_0)} I(d). \tag{4.21}$$

Искомые в задаче (4.21) траектория $x^*(\cdot) = \{x_0, x^*(1),..., x^*(N)\}$ и управление $u^*(\cdot) = \{u^*(0), u^*(1),..., u^*(N-1)\}$ (элементы пары d^*) называются соответственно оптимальной траекторией и оптимальным управлением.

Если любому допустимому управлению $u(\cdot)$ соответствует единственная пара d, то задача (4.21) может быть переписана в эквивалентной форме как задача поиска минимума функционала на множестве допустимых управлений.

Задача поиска максимума функционала может быть сведена к задаче поиска минимума путем замены знака перед функционалом на противоположный. Функционал (4.20) является более общим по сравнению с (4.19) и его применение характерно для так называемых несепарабельных задач.

4.3.2. Алгоритм решения задачи

Сравним поставленную задачу оптимального управления с формулировкой общей задачи минимизации (1.1) с целевой функцией $f(x) = f(x_1, x_2, ..., x_n)$, определенной на множестве допустимых решений $D \subseteq R^n$ параллелепипедного типа, где $D = \{x \mid x_i \in [a_i, b_i], i = 1, 2, ..., n\}$.

В качестве вектора независимых переменных $x = (x_1, ..., x_n)^T$ используется вектор $(u(0), u(1), ..., u(N-1))^T$, координаты которого удовлетворяют ограничениям вида $u(t) \in U \subseteq R^q$, где $U = \{u \mid u_i \in [a_i, b_i], i = 1, ..., q\}$, а в качестве целевой функции – функционал (4.19) или (4.20).

Алгоритм решения задачи выглядит следующим образом.

1. Выбрать мультиагентный или биоинспирированный метод оптимизации и задать его параметры.
2. Реализовать заданное число итераций выбранным методом. Для вычисления значения функционала (4.19) или (4.20) решать систему уравнений (4.17) с заданным начальным условием (4.18) с соответствующим управлением $u(\cdot) = \{u(0), u(1), ..., u(N-1)\}$.
3. В результате найти управление $u^*(\cdot) = \{u^*(0), u^*(1), ..., u^*(N-1)\}$ и соответствующую ему траекторию $x^*(\cdot) = \{x_0, x^*(1), ..., x^*(N)\}$, а также подсчитать величину функционала.

4.3.3. Модельные примеры

Для подбора параметров и оценки эффективности работы биоинспирированных методов оптимизации, имитирующих поведение стаи окуней (PSS) и имитирующих поведение стаи синиц (TFO), описанных в разд. 2.3 и 2.4, а также их применения в задачах поиска оптимального управления создан программный комплекс [133, 134]. Он реализован и протестирован на компьютере с процессором Intel Core i7-2860QM с частотой 3,3 GHz и оперативной памятью DDR3 объемом 16 GB. В качестве среды разработки использовалась Microsoft Visual Studio 2019 и язык программирования C# 7.0.

Мультиагентные методы оптимизации: самоорганизующийся миграционный алгоритм (SOMA) и модифицированный самоорганизующийся миграционный алгоритм (MSOMA), описанные в разд. 1.7, мультиагентный миграционный алгоритм с прогнозирующими динамическими моделями движения агентов (MPCMA), описанный в разд. 1.8, реализованы с помощью программного обеспечения Jupiter notebook на языке Python. Использовался компьютер MacBook Pro с 16 GB оперативной памяти и процессором M1 Pro с частотой 3,2 GHz [49, 142, 143].

Пример 4.1 (одномерная задача оптимального управления линейной системой с квадратичным критерием).

Поведение модели объекта управления описывается разностным уравнением:

$$x(t+1) = x(t) + u(t),$$

где $x \in R$, $t = 0, 1, ..., N-1$, $u \in R$.

Начальное состояние определяется из условия: $x(0)=3$.

На множестве допустимых процессов определен функционал качества:

$$I=\sum_{t=0}^{N-1}\left[u^2(t)+x^2(t)\right].$$

Требуется найти минимальное значение функционала и оптимальный процесс $\left(x^*(\cdot),u^*(\cdot)\right)$, на котором это значение достигается.

В данном примере введем искусственное ограничение на управление: $-100\le u\le 100$. Граничные значения подбираются итерационно, чтобы убедиться, что они заведомо выполняются (в исходной задаче ограничений на управление нет). Число шагов дискретного времени (разбиения): $N=10$.

В табл. 4.1 и 4.2 приведены параметры алгоритмов PSS и TFO, при которых находится решение рассматриваемой задачи.

Таблица 4.1. Параметры метода PSS

NStep	*M*	*s*	$Iter_{\max}$	λ	α	$PR_{\max}$	Δ_{pr}
50	4	190	80	3	0,6	15	5

Таблица 4.2. Параметры метода TFO

NP	γ	η	α	λ	*P*	*K*	*L*	μ	*h*	c_1	c_2	c_3	ρ
80	0,9	0,7	0,01	1,3	60	40	8	2	0,1	3	3	3	20

Табл. 4.3 и 4.4 содержат в себе полученные численные решения с помощью методов, имитирующих поведение стаи окуней и стаи синиц соответственно.

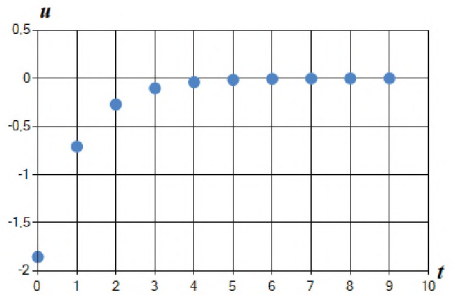
Таблица 4.3. Оптимальные управление и траектория (PSS)

t	0	1	2	3	4	5	6
u^*	–1,85403	– 0,70826	– 0,27049	– 0,10337	– 0,03953	– 0,01489	– 0,00576
x^*	3,00000	1,14597	0,43770	0,16722	0,06384	0,02432	0,00942
t	7	8	9	10			
u^*	– 0,00219	– 0,00072	0,00034	—			
x^*	0,00366	0,00147	0,00076	0,00109			

Таблица 4.4. Оптимальные управление и траектория (TFO)

t	0	1	2	3	4	5	6
u^*	– 1,85411	– 0,70819	– 0,27052	– 0,10333	– 0,03947	– 0,01507	– 0,00573
x^*	3,00000	1,14589	0,43771	0,16719	0,06386	0,02439	0,00932
t	7	8	9	10			
u^*	– 0,00216	– 0,00071	0,00000	—			
x^*	0,00359	0,00143	0,00071	0,00071			

Время расчета составляет $t^*_{PSS}=1,65$ сек., $t^*_{TFO}=0,39$ сек. На рис. 4.1, 4.2 показаны полученные оптимальные управление и траектория для обоих методов.



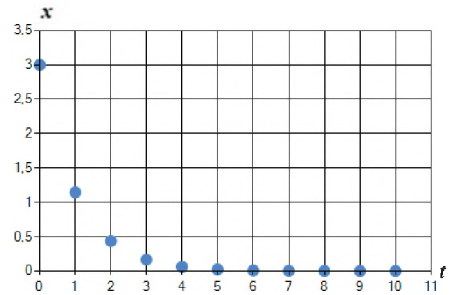
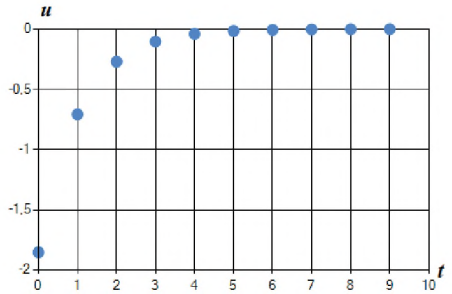


Рис. 4.1. Оптимальные управление и траектория в примере 4.1 (PSS)



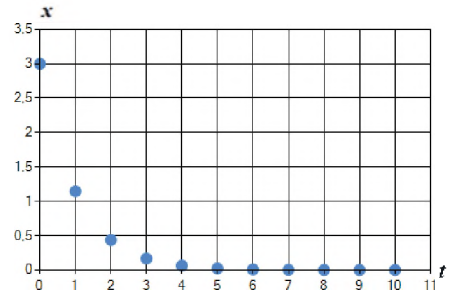


Рис. 4.2. Оптимальные управление и траектория в примере 4.1 (TFO)

Полученные значения функционала качества: $I^*_{PSS} = 14,5623058616287$, $I^*_{TFO} = 14,5623056693663$. Точное значение: $I_{\min} = 14,56230566850035$.

Пример 4.2. (одномерная задача оптимального управления с известным точным решением).

Поведение модели объекта управления описывается разностным уравнением:

$$x(t+1) = x(t) + u(t), \quad t = 0,1,\ldots,N-1,$$

где $x \in R$, ограничение на управление: $-2 \cdot 10^4 \le u \le 0$.

Начальное состояние: $x(0) = 0$.

На множестве допустимых процессов определен функционал качества

$$I = \frac{1}{2}\sum_{t=0}^{N-1} \gamma^{-t} u^2(t) + x(N), \quad \gamma > 1.$$

Требуется найти минимальное значение функционала и оптимальный процесс $(x^*(\cdot), u^*(\cdot))$, на котором это значение достигается.

Рассматриваемая задача имеет аналитическое решение, представленное в виде:

$$u^*(t) = -\gamma^t, \quad x^*(t) = x_0 + \frac{1-\gamma^t}{\gamma - 1}, \quad \min I = x_0 + \frac{1-\gamma^N}{2(\gamma-1)}.$$

Зададим значение параметра $\gamma = 1,1$ и число шагов дискретного времени $N = 50$, тогда можно вычислить точное минимальное значение функционала качества: $I_{\min} = -581,95426439877$. На рис. 4.3 изображены графики точного решения примера 4.2.

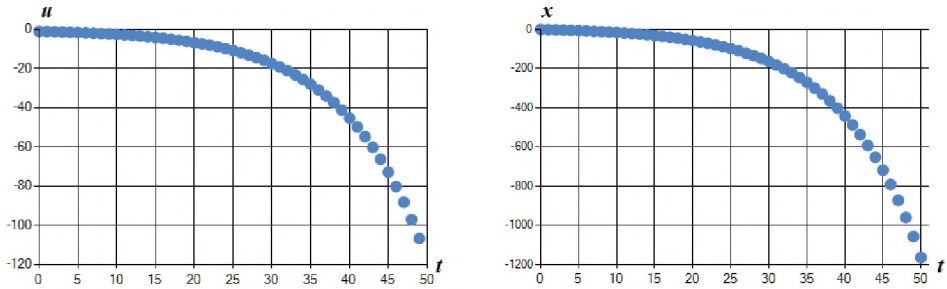


Рис. 4.3. Оптимальные управление и траектория в примере 4.2 (точное решение)

Результаты применения алгоритма SOMA (параметры приведены в табл. 4.5): полученное значение функционала качества $\min I = -581,32586179$, время расчетов 2,2 мин. На рис. 4.4 показаны найденные оптимальные траектория и управление.

Таблица 4.5. Параметры алгоритма SOMA

Параметры				
NP	*Nstep*	*PRT*	*Migrations*	*MinDist*
1000	100	0,3	70	0,001

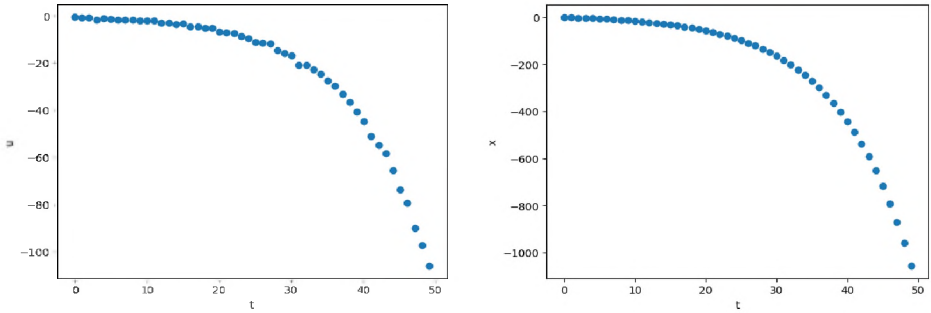


Рис. 4.4. Оптимальные управление и траектория в примере 4.2, найденные при помощи SOMA

Результаты применения алгоритма MSOMA (параметры приведены в табл. 4.6): полученное значение функционала качества $\min I = -581,8043456$, время расчетов 10,7 мин. На рис. 4.5 показаны найденные оптимальные траектория и управление.

Таблица 4.6. Параметры алгоритма MSOMA

Параметры				
NP	*Nstep*	*PRT*	*Migrations*	*MinDiv*
1000	100	0,3	150	0,001

Результаты применения алгоритма MPCMA, описанного в разд. 1.8 (параметры приведены в табл. 4.7): значение функционала качества $\min I = -580,99923$, время расчетов 44 мин. На рис. 4.6 показаны полученные оптимальные траектория и управление.

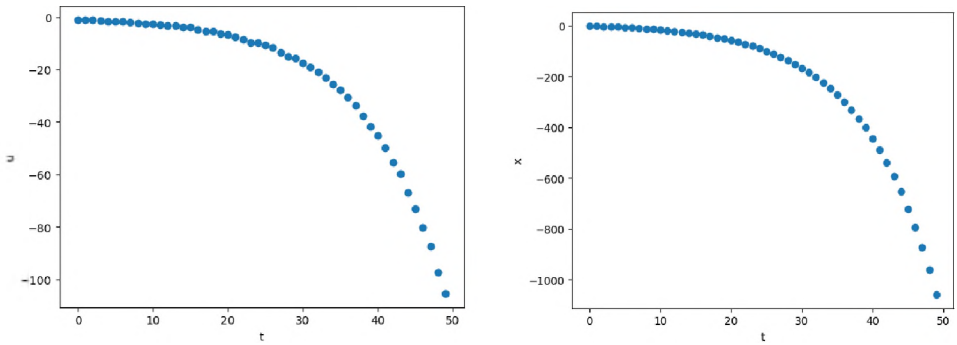


Рис. 4.5. Оптимальные управление и траектория в примере 4.2, найденные при помощи MSOMA

Таблица 4.7. Параметры алгоритма MPCMA

Параметры								
NP	$ITER_{\max}$	h	$N_{\max}$	$dist$	N_p	N_c	σ%	q
300	5	0,01	20	0,001	25	20	30	2

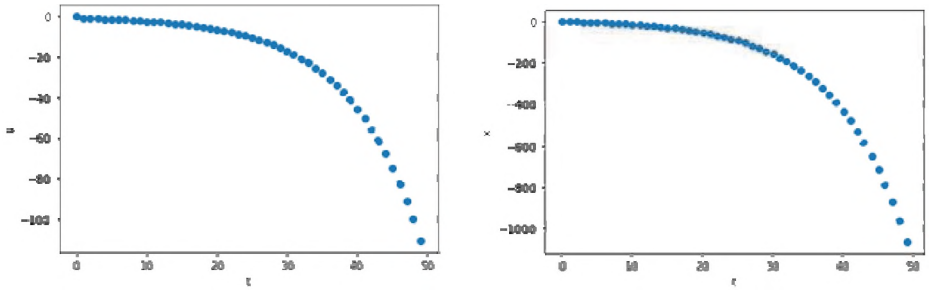


Рис. 4.6. Оптимальные управление и траектория в примере 4.2, найденные при помощи MPCMA

Метод MPCMA эффективно справляется с решением примера 4.2, но требуется достаточно большой объем вычислительных ресурсов. Методы SOMA и MSOMA превосходят его в точности и скорости.

ПРИМЕНЕНИЕ БИОИНСПИРИРОВАННЫХ МЕТОДОВ

Приведем результаты исследования влияния параметров методов PSS и TFO на время расчетов и точность результата. Кроме того, исследуем влияние увеличения размерности задачи.

Вариант 1 (сравнительный анализ алгоритмов PSS и TFO).

В таблицах 4.8 и 4.9 приведены параметры алгоритмов, при которых находится решение рассматриваемой задачи.

Таблица 4.8. Параметры метода PSS, вариант 1

$NStep$	M	s	$Iter_{\max}$	λ	α	$PR_{\max}$	Δ_{pr}
50	6	50	4000	1,4	0,2	20	9

Таблица 4.9. Параметры метода TFO, вариант 1

NP	γ	η	α	λ	P	K	L	μ	h	c_1	c_2	c_3	ρ
100	0,89	0,8	0,001	1,6	85	40	15	2	0,1	5	5	5	20

Время расчета составляет $t^*_{PSS} = 3$ мин. 56 сек., $t^*_{TFO} = 6,94$ сек. Очевидно, метод TFO оказался более быстрым. Величины полученных значений функционала качества $I^*_{PSS} = -581,954264289767$, $I^*_{TFO} = -581,954248600788$. Точное минимальное значение функционала качества: $I_{\min} = -581,95426439877$.

На рис. 4.7, 4.8 изображены полученные оптимальные управление и траектория для обоих методов.

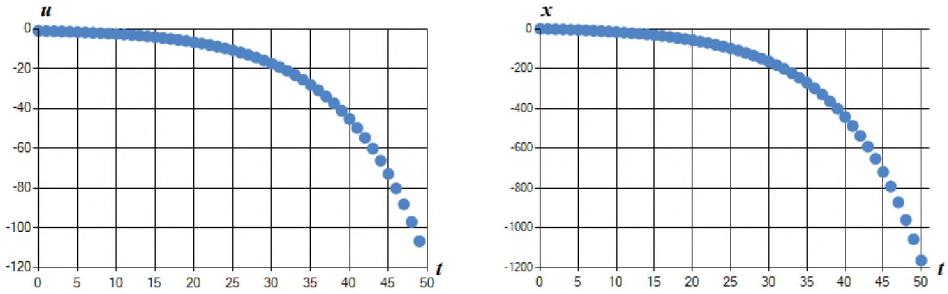


Рис. 4.7. Оптимальные управление и траектория в примере 4.2 (PSS, вариант 1)

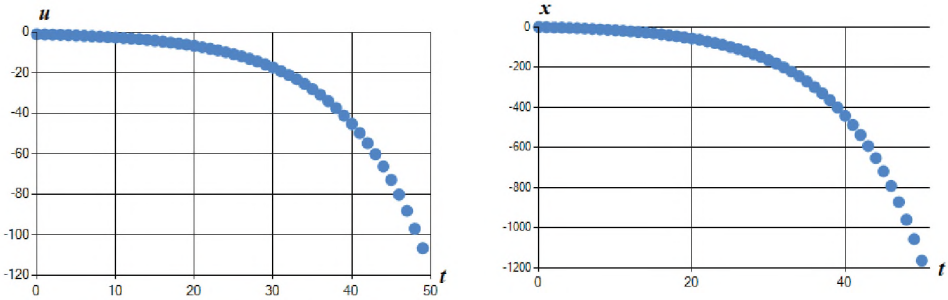


Рис. 4.8. Оптимальные управление и траектория в примере 4.2 (TFO, вариант 1)

Вариант 2 (анализ влияния параметров алгоритма TFO на точность и время расчетов).

Случай А. Изменим параметры метода TFO по сравнению с представленными в табл. 4.9 и зададим $\varepsilon = 10^{-9}$. С параметрами метода TFO из табл. 4.10 величина функционала $I^*_{TFO} = -581,953556374572$, а время расчетов 1 мин. 7 сек. Полученные оптимальные управление и траектория аналогичны изображенным на рис. 4.8.

Таблица 4.10. Параметры метода TFO, вариант 2,A

NP	γ	η	α	λ	P	K	L	μ	h	c_1	c_2	c_3	ρ
1000	0,1	0,6	0,1	1,3	160	50	4	30	0,01	30	30	30	1000

Случай Б. Для оценки влияния отклонений от параметров метода, приведенных в табл. 4.6, 4.7, зададим другую совокупность параметров в табл. 4.11. Полученные оптимальные управление и траектория изображены на рис. 4.9. Величина функционала $I^*_{TFO} = -579{,}311903151533$ при этом ухудшилась, а время расчетов составило 14,92 сек., что в 2 раза больше соответствующего варианту 1.

Таблица 4.11. Параметры метода TFO, вариант 2,Б

NP	γ	η	α	λ	P	K	L	μ	h	c_1	c_2	c_3	ρ
1000	0,3	0,6	0,1	1,3	20	10	2	30	0,01	30	30	30	1000

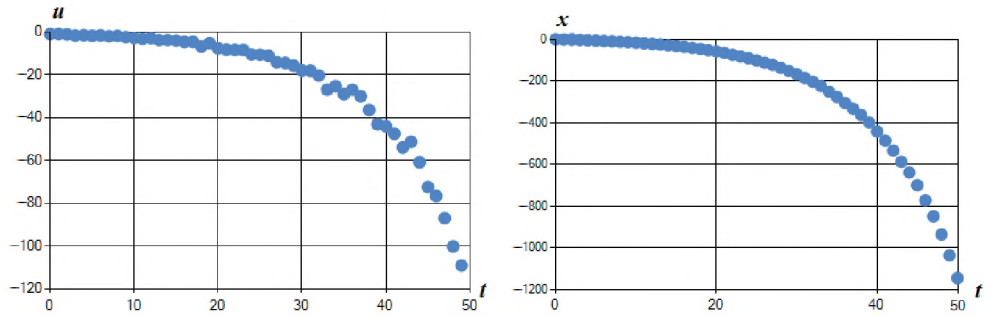


Рис. 4.9. Оптимальные управление и траектория в примере 4.2 (TFO, вариант 2,Б)

Случай В. Увеличим число шагов дискретного времени: $N = 80$. При этом точное минимальное значение функционала $I_{\min} = -10237{,}001072927329$. Полученное значение с параметрами из табл. 4.10: $I^*_{TFO} = -10174{,}6684936038$, а время расчетов 1 мин. 58 сек. Оптимальные управление и траектория изображены на рис. 4.10.

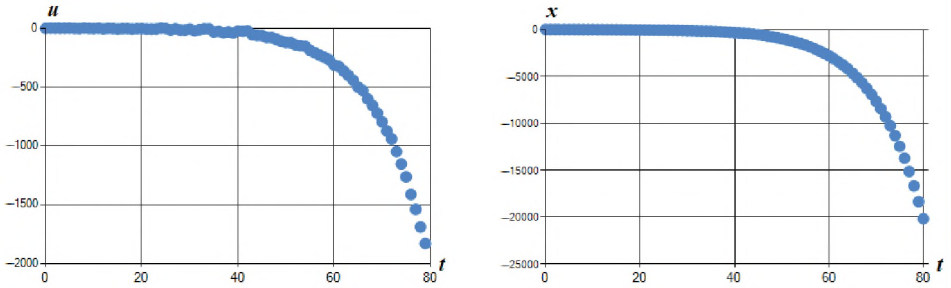


Рис. 4.10. Оптимальные управление и траектория в примере 4.2 (TFO, вариант 2,В)

Вариант 3 (анализ влияния параметров алгоритма PSS на точность и время расчетов).

Случай А. Изменим параметры метода PSS по сравнению с представленными в табл. 4.8. Полученное значение функционала с параметрами из табл. 4.12 хуже соответствующего варианту 1: $I^*_{PSS} = -569{,}064908725205$, а время вычислений 31 сек. Оптимальные управление и траектория изображены на рис. 4.11.

Таблица 4.12. Параметры метода PSS, вариант 3,А

$NStep$	M	s	$Iter_{\max}$	λ	α	$PR_{\max}$	Δ_{pr}
50	6	25	1000	1,4	0,2	20	9

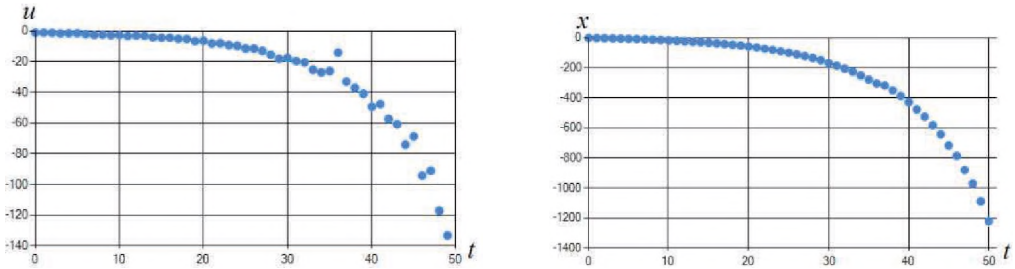


Рис. 4.11. Оптимальные управление и траектория в примере 4.2 (PSS, вариант 3,А)

Случай Б. Увеличим число шагов дискретного времени: $N = 80$. При этом точное минимальное значение функционала $I_{\min} = -10237{,}001072927329$. Полученное значение с параметрами из табл. 4.8: $I^{*}_{PSS} = -10237{,}0010729268$, а время расчетов 9 мин.31 сек. Заметим, что время расчетов значительно возросло. Оптимальные управление и траектория изображены на рис. 4.12.

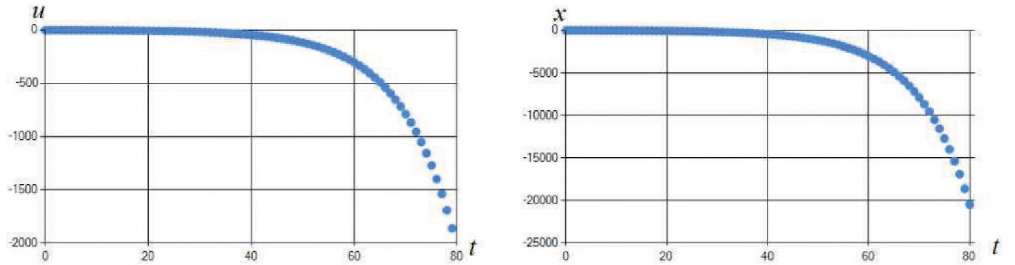


Рис. 4.12. Оптимальные управление и траектория в примере 4.2 (PSS, вариант 3,Б)

Пример 4.3 (задача Лууса–Тассона (Luus–Tassone) с несепарабельным критерием).

Поведение модели объекта управления описывается системой разностных уравнений дискретной системы:

$$x_1(t+1) = \frac{x_1(t)}{1+0{,}01u_1(t)(3+u_2(t))};$$

$$x_2(t+1) = \frac{x_2(t)+u_1(t)x_1(t+1)}{1+u_1(t)(1+u_2(t))};$$

$$x_3(t+1) = \frac{x_3(t)}{1+0{,}01u_2(t)(1+u_3(t))},$$

где $x \in R^3$, $t = 0,1,...,N-1$, $u \in R^3$. Ограничения на управление: $0 \le u_1(t) \le 4$, $0 \le u_2(t) \le 4$, $0 \le u_3(t) \le 0{,}5$.

Начальное состояние определяется из условия: $x(0) = (2;\ 5;\ 7)^T$.

На множестве допустимых процессов определен функционал качества

$$I = x_1^{\,2}(N) + x_2^2(N) + x_3^2(N) +$$

$$+\left[\left(\sum_{k=1}^{N} x_1^2(k-1) + x_2^2(k-1) + 2u_3^2(k-1)\right)\left(\sum_{k=1}^{N} x_3^2(k-1) + 2u_1^2(k-1) + 2u_2^2(k-1)\right)\right]^{\frac{1}{2}}.$$

Требуется найти минимальное значение функционала и оптимальный процесс $\left(x^*(\cdot), u^*(\cdot)\right)$, на котором это значение достигается.

Зададим число шагов дискретного времени (разбиения): $N = 20$. Тогда наилучшее известное значение функционала [120]: $I_{\min} = 209,26937$.

Вариант 1 (сравнительный анализ алгоритмов PSS и TFO).

В табл. 4.13 и 4.14 приведены параметры алгоритмов, при которых находится решение рассматриваемой задачи.

Таблица 4.13. Параметры метода PSS, вариант 1

NStep	*M*	*s*	$Iter_{\max}$	λ	α	$PR_{\max}$	Δ_{pr}
100	3	50	900	1,4	0,3	15	7

Таблица 4.14. Параметры метода TFO, вариант 1

NP	γ	η	α	λ	*P*	*K*	*L*	μ	*h*	c_1	c_2	c_3	ρ
100	0,89	0,8	0,01	1,5	20	20	15	3	0,1	10	10	10	0,5

Время расчета составляет $t^*_{PSS} = 45,01$ сек., $t^*_{TFO} = 2,79$ сек. На рис. 4.13 – 4.16 показаны полученные оптимальные управление и траектория для обоих методов.

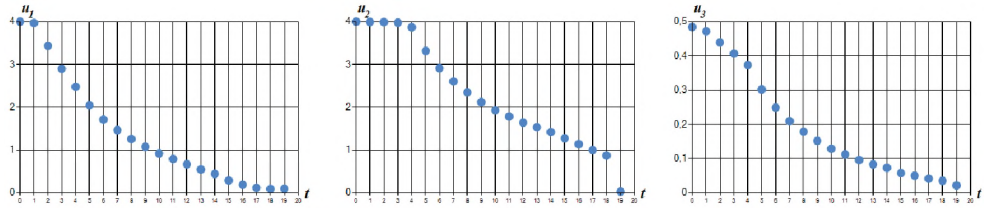


Рис. 4.13. Оптимальное управление в примере 4.3 (PSS, вариант 1)

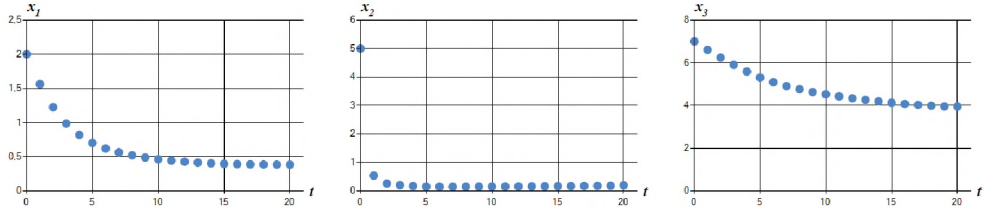


Рис. 4.14. Оптимальная траектория в примере 4.3 (PSS, вариант 1)

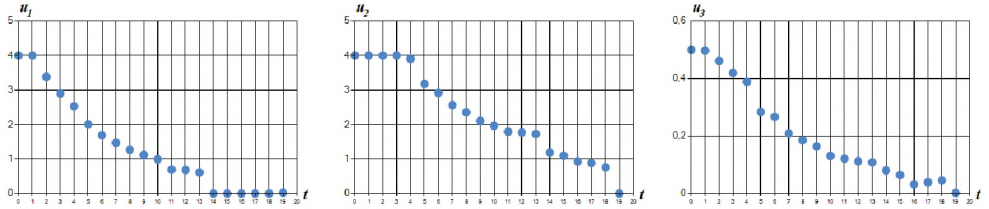


Рис. 4.15. Оптимальное управление в примере 4.3 (TFO, вариант 1)

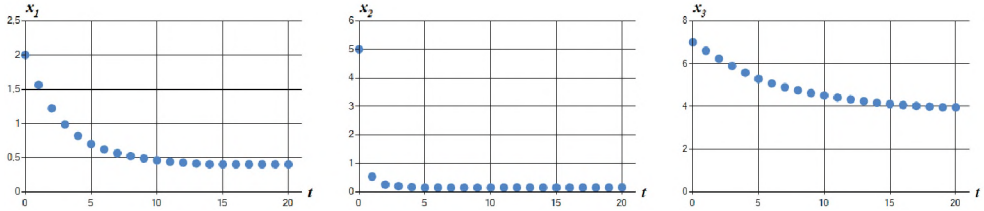


Рис. 4.16. Оптимальная траектория в примере 4.3 (TFO, вариант 1)

Величины полученных значений функционала качества: $I^*_{PSS} = 209,432827203523$, $I^*_{TFO} = 209,417488109186$.

Вариант 2 (анализ влияния параметров метода TFO на точность и время расчетов). Изменим параметры метода TFO по сравнению с представленными в табл. 4.14 и зададим $\varepsilon = 10^{-9}$. С параметрами метода TFO из табл. 4.15 величина функционала $I^*_{TFO} = 209,389060601957$, а время расчетов $t^*_{TFO} = 16,47$ сек. Полученные оптимальные управление и траектория изображены на рис. 4.17, 4.18.

Таблица 4.15. Параметры метода TFO, вариант 2

NP	γ	η	α	λ	*P*	*K*	*L*	μ	*h*	c_1	c_2	c_3	ρ
300	0,3	0,9	0,0001	1,1	100	10	10	1	0,05	2	0,5	3	2

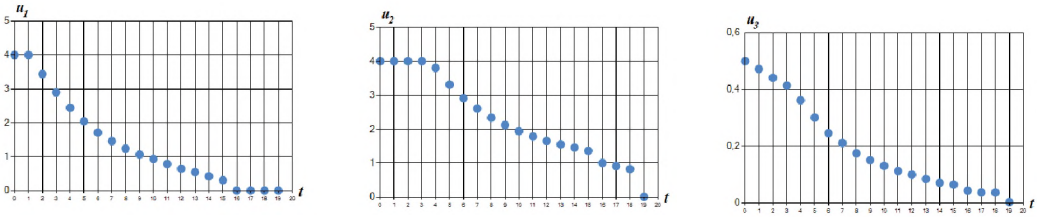


Рис. 4.17. Оптимальное управление в примере 4.3 (TFO, вариант 2)

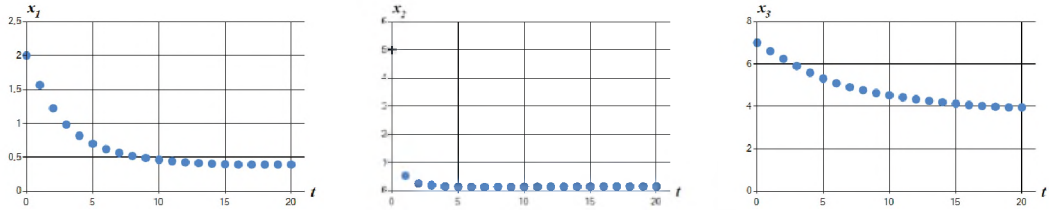


Рис. 4.18. Оптимальная траектория в примере 4.3 (TFO, вариант 2)

Вариант 3 (анализ случайных факторов в методе PSS на точность и время расчетов). Применим метод PSS с параметрами из табл. 4.13 и сделаем несколько запусков, выбирая наилучший вариант. Полученное в результате значение функционала $I^*_{PSS} = 209,429533683522$ лучше соответствующего варианту 1, а время вычислений 49,92 сек. Оптимальные управление и траектория аналогичны изображенным на рис. 4.13, 4.14.

Пример 4.4 (задача Ли–Хаймса (Li–Haimes) с несепарабельным критерием).
Поведение модели объекта управления описывается разностным уравнением:

$$x(1) = x(0)^{u(0)}, \quad x(2) = (1+u(1))\cdot x(1), \quad x(3) = x(2)+u(2),$$

где $x \in R$, $t = 0,1,2$, $u \in R$. Ограничение на управление: $-1 \le u(t) \le 1$. Число шагов дискретного времени (разбиения): $N = 3$.

Начальное состояние определяется из условия: $x(0) = 15$.

На множестве допустимых процессов определен функционал качества

$$I = [x^2(0) + x^2(1) + (2x^2(2) + x^2(3))\exp(x^2(1))] \times [50 + u^2(0) + (u^2(1) + u^2(2))\exp(u^2(0))]^{1/2}.$$

Требуется найти минимальное значение функционала и оптимальный процесс $\left(x^*(\cdot), u^*(\cdot)\right)$, на котором это значение достигается.

Лучшее известное значение функционала качества для данной несепарабельной задачи [120]: $I_{\min} = 1596,4796778$.

Вариант 1 (сравнительный анализ алгоритмов PSS и TFO). В табл. 4.16 и 4.17 приведены параметры алгоритмов, при которых находится решение рассматриваемой задачи.

Таблица 4.16. Параметры метода PSS, вариант 1

NStep	*M*	*s*	$Iter_{\max}$	λ	α	$PR_{\max}$	Δ_{pr}
40	4	16	40	3	0,6	15	3

Таблица 4.17. Параметры метода TFO, вариант 1

NP	γ	η	α	λ	*P*	*K*	*L*	μ	*h*	c_1	c_2	c_3	ρ
100	0,89	0,8	0,0001	1,2	10	20	7	2	0,1	7	7	7	0,5

Табл. 4.18 и 4.19 содержат в себе полученные численные решения с помощью метода, имитирующего поведение стаи окуней, и метода стаи синиц соответственно.

Таблица 4.18. Оптимальные управление и траектория (PSS, вариант 1)

t	*u*	*x*
0	– 0,42716	15,00000
1	– 0,09897	0,31450
2	– 0,08238	0,28337
3	—	0,20099

Таблица 4.19. Оптимальные управление и траектория (TFO, вариант 1)

t	*u*	*x*
0	– 0,42719	15,00000
1	– 0,09892	0,31448
2	– 0,08236	0,28337
3	—	0,20101

Время расчета составляет $t^*_{PSS} = 0,04$ сек., $t^*_{TFO} = 0,13$ сек. На рис. 4.19, 4.20 показаны полученные оптимальные управление и траектория для обоих методов.

Величины полученных обоими методами значений функционала качества: $I^*_{PSS} = 1596,4796778342$, $I^*_{TFO} = 1596,47967789742$.

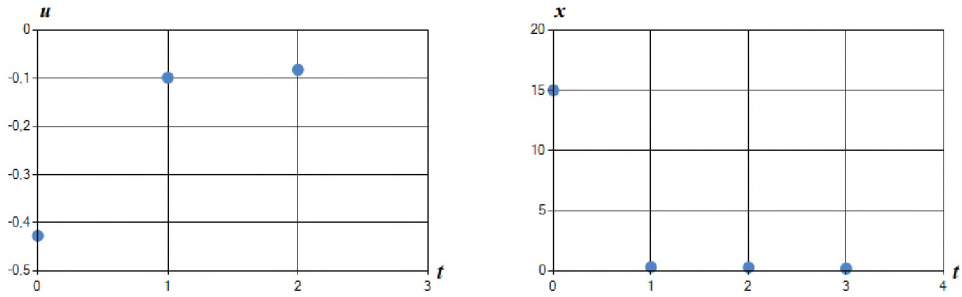


Рис. 4.19. Оптимальные управление и траектория в примере 4.4 (PSS, вариант 1)

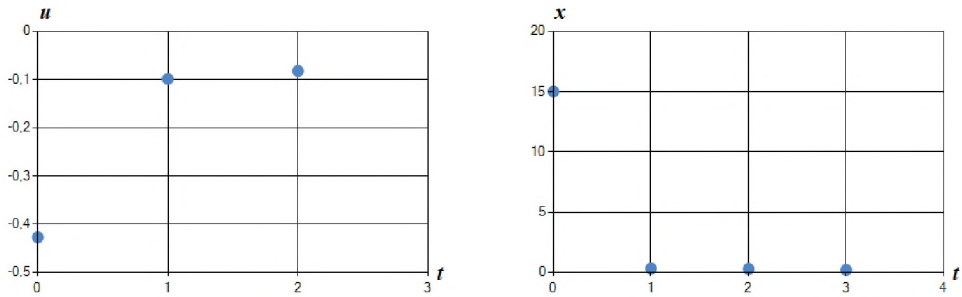


Рис. 4.20. Оптимальные управление и траектория в примере 4.4 (TFO, вариант 1)

Вариант 2 (анализ влияния параметров алгоритма TFO на точность и время расчетов). Изменим параметры метода TFO по сравнению с представленными в табл. 4.17 и зададим $\varepsilon = 10^{-9}$. С параметрами метода TFO из табл. 4.20 величина функционала $I^*_{TFO} = 1596,47967783381$, а время расчетов $t^*_{TFO} = 0,13$ сек. Полученные оптимальные управление и траектория совпадают с представленными в табл. 4.18.

Таблица 4.20. Параметры метода TFO, вариант 2

NP	γ	η	α	λ	P	K	L	μ	h	c_1	c_2	c_3	ρ
100	0,1	0,1	0,001	1,5	100	10	5	2	0,1	3	3	3	5

Вариант 3 (анализ влияния параметров алгоритма PSS на точность и время расчетов). Изменим параметры метода PSS по сравнению с представленными в табл. 4.17. С параметрами из табл. 4.21 полученное значение функционала: $I^*_{PSS} = 1596,47967783389$, а время вычислений 0,19 сек. Оптимальные управление и траектории совпадают с представленными в табл. 4.18.

Таблица 4.21. Параметры метода PSS, вариант 3

$NStep$	M	s	$Iter_{\max}$	λ	α	$PR_{\max}$	Δ_{pr}
100	4	20	65	3	0,6	15	5

Пример 4.5 (двумерная задача Лагранжа с известным точным решением).

Поведение модели объекта управления описывается системой разностных уравнений:

$$x_1(t+1) = x_1(t) + u(t),$$

$$x_2(t+1) = 2x_1(t) + x_2(t),$$

где $x \in R^2$, $t = 0,1$, $u \in R$. Число шагов дискретного времени (разбиения): $N = 2$.

Начальное состояние определяется из условия: $x(0) = (2;1)^T$. Для соответствия задаче (4.21) введем искусственное ограничение на управление: $-10^5 \le u \le 10^5$.

На множестве допустимых процессов определен функционал качества

$$I = \sum_{t=0}^{1} \left[x_1^2(t) + x_2^2(t) + u^2(t) \right].$$

Требуется найти минимальное значение функционала и оптимальный процесс $\left(x^*(\cdot), u^*(\cdot)\right)$, на котором это значение достигается.

Пример 4.5 имеет точное решение: $u^* = \{-1;0\}$; $x_1^*(\cdot) = \{2;1;1\}$; $x_2^*(\cdot) = \{1;5;7\}$.

Вариант 1 (сравнительный анализ алгоритмов PSS и TFO). В табл. 4.22 и 4.23 приведены параметры алгоритмов. при которых находится решение рассматриваемой задачи.

Таблица 4.22. Параметры метода PSS, вариант 1

NStep	*M*	*s*	$Iter_{\max}$	λ	α	$PR_{\max}$	Δ_{pr}
30	4	50	25	3	0,6	15	5

Таблица 4.23. Параметры метода TFO, вариант 1

NP	γ	η	α	λ	*P*	*K*	*L*	μ	*h*	c_1	c_2	c_3	ρ
20	0,85	0,91	0,001	1,6	50	10	4	2	0,1	4	4	4	10

Время расчета составляет $t^*_{PSS} = 0,05$ сек., $t^*_{TFO} = 0,04$ сек. На рис. 4.21. 4.22 показаны полученные оптимальные траектория и управления для обоих методов.

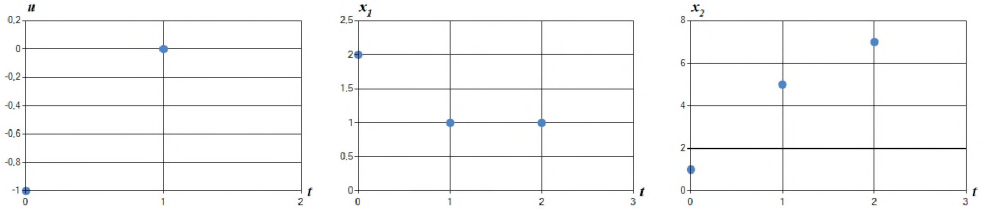


Рис. 4.21. Оптимальные управление и траектория в примере 4.5 (PSS, вариант 1)

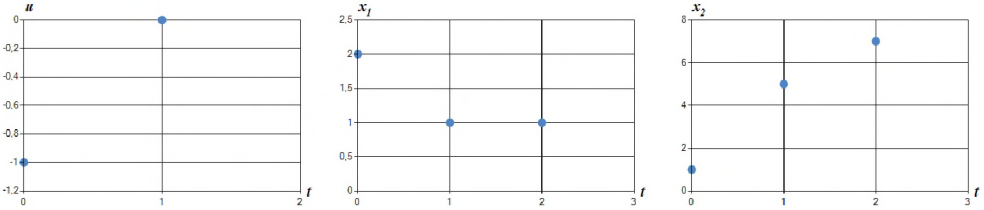


Рис. 4.22. Оптимальные управление и траектория в примере 4.5 (TFO, вариант 1)

Величины полученных значений функционала качества: $I^*_{PSS} = 32$, $I^*_{TFO} = 32,0000000003427$. Точное минимальное значение функционала качества: $I_{\min} = 32$. Значение функционала качества I^*_{PSS} совпадает с точным решением.

Вариант 2 (анализ влияния параметров алгоритма TFO на точность и время расчетов), Изменим параметры метода TFO по сравнению с представленными в табл. 4.23 и зададим $\varepsilon = 10^{-9}$. С параметрами метода TFO из табл. 4.24 величина функционала $I^*_{TFO} = 32,0000000000001$, а время расчетов $t^*_{TFO} = 0,01$сек. Полученные оптимальные управление и траектория практически совпадают с представленными на рис. 4.22.

Таблица 4.24. Параметры метода TFO, вариант 2

NP	γ	η	α	λ	*P*	*K*	*L*	μ	*h*	c_1	c_2	c_3	ρ
70	0,1	0,1	0,001	1,5	10	5	6	2	0,1	2	3	2	100

Вариант 3 (анализ влияния параметров алгоритма PSS на точность и время расчетов). Изменим параметры метода PSS по сравнению с представленными в табл. 4.22. С параметрами из табл. 4.25 полученное значение функционала: $I^*_{PSS} = 32$, а время вычислений 1,58 сек. Оптимальные управление и траектория совпадают с представленными на рис. 4.21.

Таблица 4.25. Параметры метода PSS, вариант 3

NStep	*M*	*s*	$Iter_{\max}$	λ	α	$PR_{\max}$	Δ_{pr}
100	4	100	110	3	0,6	15	5

Пример 4.6 (двумерная задача Майера).

Поведение модели объекта управления описывается системой разностных уравнений:

$$x_1(t+1) = x_2(t),$$

$$x_2(t+1) = x_1(t) - u(t),$$

где $x \in R^2$, $t = 0,1$, $u \in R$. Ограничение на управление: $-1 \le u \le 1$. Число шагов дискретного времени (разбиения): $N = 2$.

Начальное состояние определяется из условия: $x(0) = (2;-3)^T$.

На множестве допустимых процессов определен функционал качества

$$I = x_1^2(2) + x_2^2(2).$$

Требуется найти минимальное значение функционала и оптимальный процесс $\left(x^*(\cdot), u^*(\cdot)\right)$, на котором это значение достигается,

Вариант 1 (сравнительный анализ алгоритмов PSS и TFO). В табл. 4.26 и 4.27 приведены параметры алгоритмов, при которых находится решение рассматриваемой задачи.

Таблица 4.26. Параметры метода PSS, вариант 1

NStep	*M*	*s*	$Iter_{\max}$	λ	α	$PR_{\max}$	Δ_{pr}
100	4	15	120	1,5	0,4	10	5

Таблица 4.27. Параметры метода TFO, вариант 1

NP	γ	η	α	λ	P	K	L	μ	h	c_1	c_2	c_3	ρ
10	0,85	0,91	0,01	1,1	20	5	4	2	0,1	1	1	1	2

Табл. 4.28 и 4.29 содержат в себе полученные численные решения с помощью метода, имитирующего поведение стаи окуней, и метода стаи синиц соответственно.

Таблица 4.28. Оптимальные управление и траектория (PSS, вариант 1)

t	u	x_1	x_2
0	1	2	−3
1	−1	−3	1
2	—	1	−2

Таблица 4.29. Оптимальные управление и траектория (TFO, вариант 1)

t	u	x_1	x_2
0	1	2	−3
1	−1	−3	1
2	—	1	−2

Время расчета составляет $t^*_{PSS} = 0{,}19$ сек., $t^*_{TFO} = 0{,}02$ сек. На рис. 4.23, 4.24 показаны полученные оптимальные управление и траектория для обоих методов.

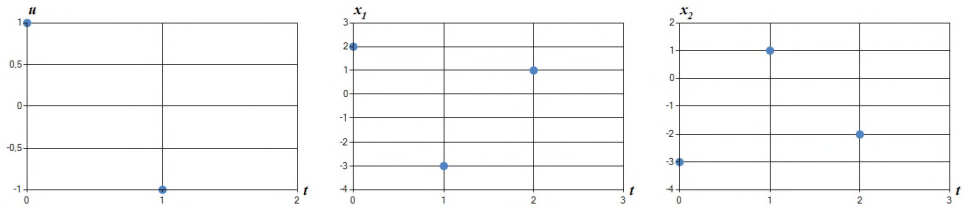


Рис. 4.23. Оптимальные управление и траектория в примере 4.6 (PSS, вариант 1)

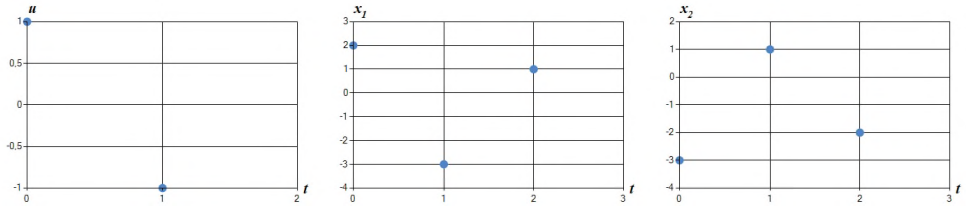


Рис. 4.24. Оптимальные управление и траектория в примере 4.6 (TFO, вариант 1)

Величины полученных обоими методами значений функционала качества: $I^*_{PSS} = 5{,}00000000000002$, $I^*_{TFO} = 5$. Точное минимальное значение функционала качества: $I_{\min} = 5$. Значение функционала качества I^*_{TFO} совпадает с точным.

Вариант 2 (анализ влияния параметров алгоритма TFO на точность и время расчетов). Изменим параметры метода TFO по сравнению с представленными в табл. 4.27 и зададим $\varepsilon = 10^{-9}$. С параметрами метода TFO из табл. 4.30 величина функцио-

нала $I^*_{TFO} = 5$, а время расчетов $t^*_{TFO} = 0,01$сек. Полученные оптимальные управление и траектория совпадают с представленными в табл. 4.29.

Таблица 4.30. Параметры метода TFO, вариант 2

NP	γ	η	α	λ	*P*	*K*	*L*	μ	*h*	c_1	c_2	c_3	ρ
30	0,81	0,6	0,2	1,2	10	5	6	2	0,1	1	1	1	2

Пример 4.7 (двумерная задача Больца с известным точным решением).

Поведение модели объекта управления описывается системой разностных уравнений:

$$x_1(t+1) = x_2(t),$$

$$x_2(t+1) = 2x_2(t) - x_1(t) + \frac{1}{N^2}u(t),$$

где $x \in R^2$, $u \in R$, $t = 0,1,...,N-1$. Задано ограничение на управление: $0 \le u \le 100$.

Начальное состояние: $x(0) = (0;0)^T$.

На множестве допустимых процессов задан функционал

$$I = -x_1(N) + \frac{1}{2N}\sum_{t=0}^{N-1} u^2(t).$$

Требуется найти минимальное значение функционал и оптимальный процесс, на котором это значение достигается.

Сформулированная задача имеет аналитическое решение: $u^*(t) = \frac{N-t-1}{N}$, $\min I = -\frac{1}{3} + \frac{3N-1}{6N^2} + \frac{1}{2N^3}\sum_{t=0}^{N-1} t^2$. Положим $N = 10$, тогда $\min I = -0,1425$.

ПРИМЕНЕНИЕ БИОИНСПИРИРОВАННЫХ МЕТОДОВ

Для решения примера используем два биоинспирированных метода оптимизации: метод, имитирующий поведение стаи окуней (PSS), и метод, имитирующий поведение стаи синиц (TFO).

Вариант 1 (сравнительный анализ алгоритмов PSS и TFO). В табл. 4.31 и 4.32 приведены параметры алгоритмов, при которых находится решение рассматриваемой задачи.

Таблица 4.31. Параметры метода PSS, вариант 1

NStep	*M*	*s*	$Iter_{\max}$	λ	α	$PR_{\max}$	Δ_{pr}
70	3	30	750	1,7	0,6	10	5

Таблица 4.32. Параметры метода TFO, вариант 1

NP	γ	η	α	λ	*P*	*K*	*L*	μ	*h*	c_1	c_2	c_3	ρ
30	0,73	0,91	0,01	1,1	200	30	8	2	0,1	2	2	2	7

Время расчета составляет $t^*_{PSS} = 3,69$ сек., $t^*_{TFO} = 0,49$ сек. На рис. 4.25, 4.26 показаны полученные оптимальные управление и траектория для обоих методов.

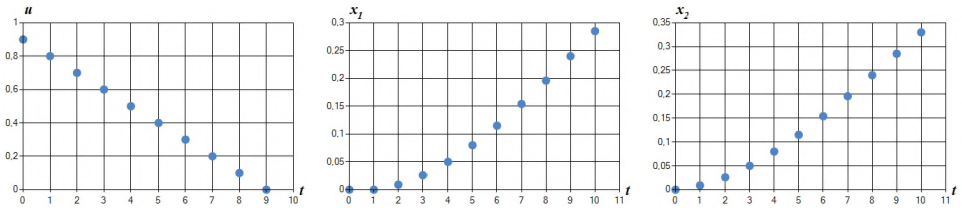


Рис. 4.25. Оптимальные управление и траектория в примере 4.7 (PSS, вариант 1)

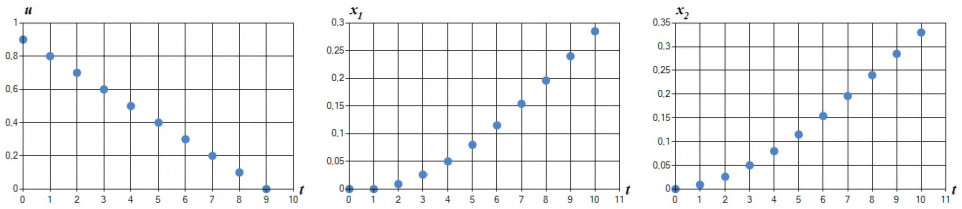


Рис. 4.26. Оптимальные управление и траектория в примере 4.7 (TFO, вариант 1)

Величины полученных обоими методами значений функционала качества: $I^*_{PSS} = -0,142499999999983$, $I^*_{TFO} = -0,14249999999985$. Точное минимальное значение функционала качества: $I_{\min} = -0,1425$.

Вариант 2 (анализ влияния параметров алгоритма TFO на точность и время расчетов). Изменим параметры метода TFO по сравнению с представленными в табл. 4.32 и зададим $\varepsilon = 10^{-9}$. С параметрами метода TFO из табл. 4.33 величина функционала $I^*_{TFO} = -0,142499964879453$, а время расчетов $t^*_{TFO} = 0,13$ сек. Полученные оптимальные управление и траектория практически совпадают с представленными на рис. 4.26.

Таблица 4.33. Параметры метода TFO, вариант 2

NP	γ	η	α	λ	*P*	*K*	*L*	μ	*h*	c_1	c_2	c_3	ρ
120	0,9	0,2	0,1	1,7	20	20	4	3	0,1	10	10	10	40

Вариант 3 (анализ влияния параметров алгоритма PSS на точность и время расчетов). Изменим параметры метода PSS по сравнению с представленными в табл. 4.31. С параметрами из табл. 4.34 полученное значение функционала: $I^*_{PSS} = -0,142499999999796$, а время вычислений 7,39 сек. Оптимальные управление и траектория совпадают с представленными на рис. 4.25.

Таблица 4.34. Параметры метода PSS, вариант 3

NStep	*M*	*s*	$Iter_{\max}$	λ	α	$PR_{\max}$	Δ_{pr}
100	3	30	750	1,7	0,6	10	5

Для решения примера используем самоорганизующийся миграционный алгоритм (SOMA), модифицированный самоорганизующийся миграционный алгоритм (MSOMA), миграционный алгоритм с прогнозирующими динамическими моделями движения агентов (MPCMA).

Результаты применения алгоритма SOMA (параметры приведены в табл. 4.35): полученное значение функционала качества $\min I = -0{,}14256$, время расчетов 26 сек. На рис. 4.27, 4.28 показаны найденные оптимальные управление и траектория.

Таблица 4.35. Параметры алгоритма SOMA

Параметры				
NP	*Nstep*	*PRT*	*Migrations*	*MinDist*
500	50	0,4	100	10^{-10}

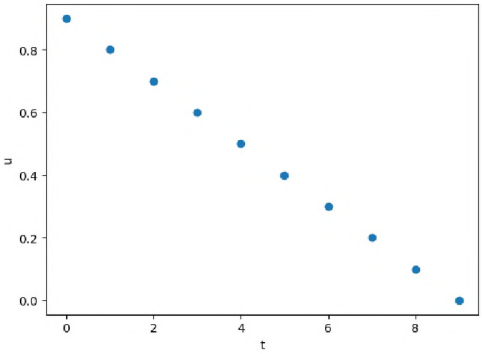


Рис. 4.27. Оптимальное управление в примере 4.7, найденное при помощи SOMA

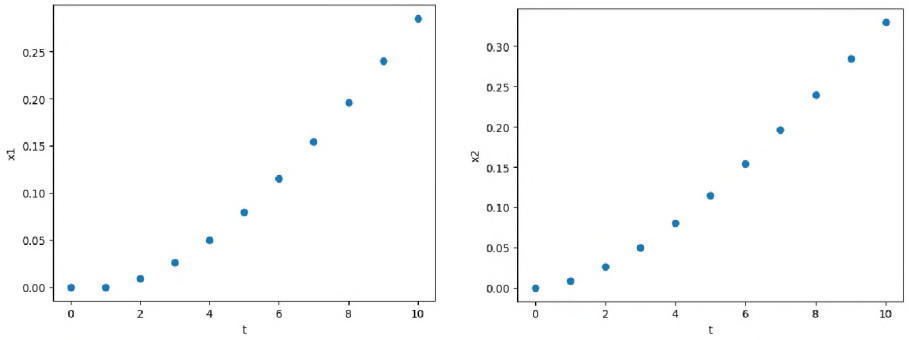


Рис. 4.28. Оптимальная траектория в примере 4.7, найденная при помощи SOMA

Результаты применения алгоритма MSOMA (параметры приведены в табл. 4.36): значение функционала качества $\min I = -0{,}1423658$, время расчетов 12,7 сек. На рис. 4.29, 4.30 показаны полученные оптимальные управление и траектория.

Таблица 4.36. Параметры алгоритма MSOMA

Параметры				
NP	*Nstep*	*PRT*	*Migrations*	*MinDiv*
100	50	0,3	100	0,0001

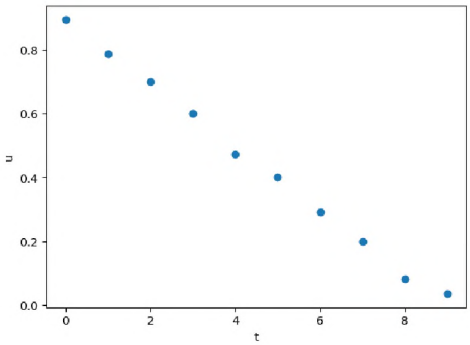


Рис. 4.29. Оптимальное управление в примере 4.7, найденное при помощи MSOMA

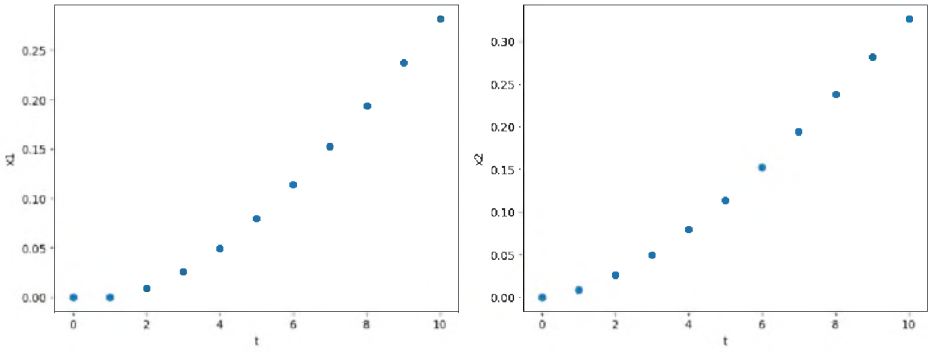


Рис. 4.30. Оптимальная траектория в примере 4.7, найденная при помощи MSOMA

Результаты работы алгоритма MPCMA (параметры приведены в табл. 4.37): полученное значение функционала качества $\min I = -0{,}14162110308$, время расчетов 159 сек. На рис. 4.31, 4.32 показаны найденные оптимальные управление и траектория.

Таблица 4.37. Параметры метода MPCMA

Параметры								
NP	$ITER_{\max}$	h	$N_{\max}$	$dist$	N_p	N_c	σ%	q
200	15	0,05	15	0,0001	15	10	30	2

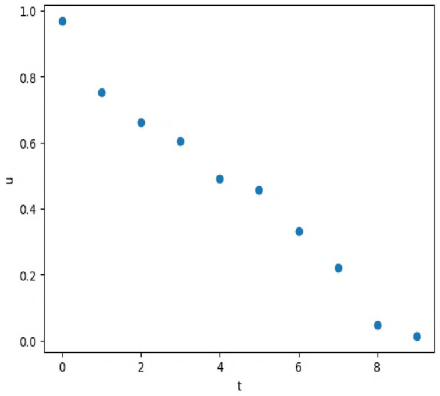


Рис. 4.31. Оптимальное управление в примере 4.7, найденное при помощи MPCMA

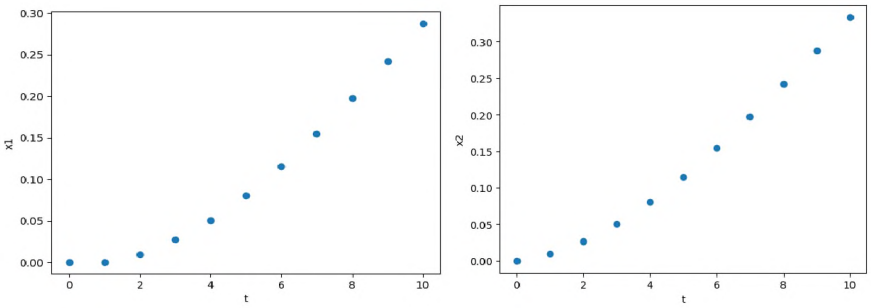


Рис. 4.32. Оптимальная траектория в примере 4.7, найденная при помощи MPCMA

В результате применения всех использованных мультиагентных алгоритмов получены решения достаточно хорошего качества. Достигнутая точность алгоритма MPCMA оказалось чуть меньше по сравнению с SOMA и MSOMA.

Пример 4.8 (двумерная задача Майера с известным точным решением).

Поведение модели объекта управления описывается системой разностных уравнений:

$$x_1(t+1) = x_1(t) + 2u(t),$$

$$x_2(t+1) = -x_1^2(t) + x_1(t) + u^2(t),$$

где $x \in R^2$, $t = 0,1$, $u \in R$. Ограничение на управление: $-5 \le u \le 5$. Число шагов дискретного времени (разбиения): $N = 2$.

Начальное состояние определяется из условия: $x(0) = (3;0)^T$.

На множестве допустимых процессов определен функционал качества

$$I = -x_2(2).$$

Требуется найти минимальное значение функционала и оптимальный процесс $\left(x^*(\cdot), u^*(\cdot)\right)$, на котором это значение достигается.

Нужно заметить, что в данной задаче дискретный принцип максимума не выполняется.

Рассматриваемая задача имеет два точных решения:

$$x_1^{1*}(\cdot) = \{3;-1;9\},\ x_2^{1*}(\cdot) = \{0;-5;19\},\ u^{1*}(\cdot) = \{-2;5\};$$

$$x_1^{2*}(\cdot) = \{3;-1;-11\},\ x_2^{2*}(\cdot) = \{0;-5;19\},\ u^{2*}(\cdot) = \{-2;-5\},$$

каждое из которых методы PSS и TFO успешно находят.

Вариант 1 (сравнительный анализ алгоритмов PSS и TFO). В табл. 4.38 и 4.39 приведены параметры алгоритмов, при которых находится решение рассматриваемой задачи.

Таблица 4.38. Параметры метода PSS, вариант 1

NStep	*M*	*s*	$Iter_{\max}$	λ	α	$PR_{\max}$	Δ_{pr}
100	6	20	75	1,3	0,2	10	5

Таблица 4.39. Параметры метода TFO, вариант 1

NP	γ	η	α	λ	P	K	L	μ	h	c_1	c_2	c_3	ρ
30	0,73	0,91	0,01	1,1	20	30	8	2	0,1	3	3	3	7

Рассмотрим задачу поиска первого решения задачи. Табл. 4.40 и 4.41 содержат в себе полученные численные решения с помощью метода, имитирующего поведение стаи окуней, и метода стаи синиц соответственно.

Таблица 4.40. Оптимальные управление и траектория (PSS), решение 1

t	u	x_1	x_2
0	–2	3	0
1	5	–1,00001	–4,99999
2	—	8,99999	19

Таблица 4.41. Оптимальные управление и траектория (TFO), решение 1

t	u	x_1	x_2
0	–1,99998	3,00000	0,00000
1	5,00000	– 0,99996	– 5,00008
2	—	9,00004	19,00000

Время расчета составляет $t^*_{PSS} = 0,22$ сек., $t^*_{TFO} = 0,05$ сек. На рис. 4.33. 4.34 показаны полученные оптимальные траектория и управления для обоих методов.

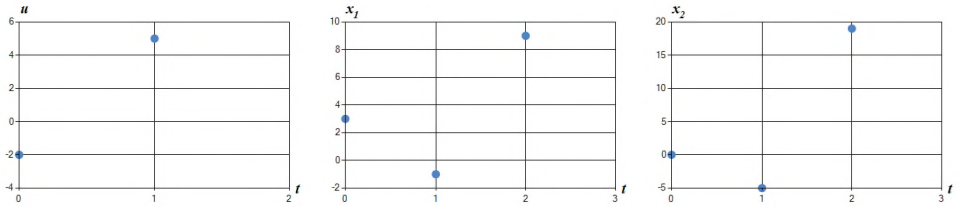


Рис. 4.33. Оптимальные управление и траектория в примере 4.8 (PSS, решение 1)

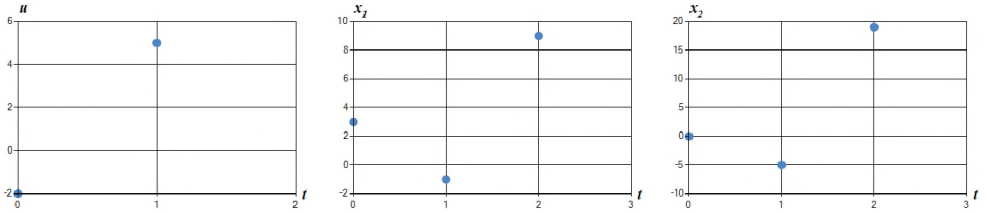


Рис. 4.34. Оптимальные управление и траектория в примере 4.8 (TFO, решение 1)

Величины полученных значений функционала качества: $I^{1*}_{PSS} = -18,999999999978$, $I^{1*}_{TFO} = -18,9999999988225$. Точное значение функционала качества: $I_{\min} = -19$.

Рассмотрим задачу поиска второго решения задачи. Табл. 4.42 и 4.43 содержат в себе полученные численные решения с помощью метода, имитирующего поведение стаи окуней, и метода, имитирующего поведение стаи синиц, соответственно.

Таблица 4.42. Оптимальные управление и траектория (PSS), решение 2

t	u	x_1	x_2
0	−2	3	0
1	−5	−1,00001	−4,99998
2	—	−11,00001	19

Таблица 4.43. Оптимальные управление и траектория (TFO), решение 2

t	u	x_1	x_2
0	−2	3	0
1	−5	−1,00003	−4,99994
2	—	−11,00003	19,00000

Время расчета составляет $t^*_{PSS} = 0,22$ сек., $t^*_{TFO} = 0,04$ сек. На рис. 4.35. 4.36 показаны полученные оптимальные управление и траектория для обоих методов.

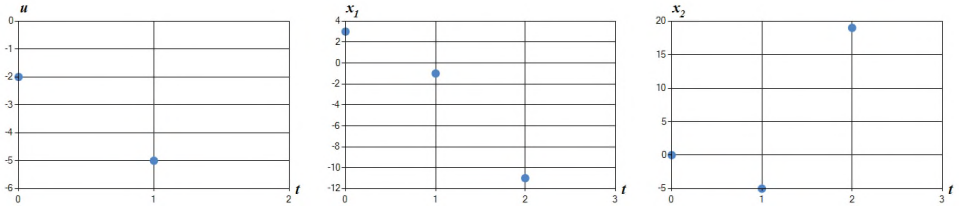


Рис. 4.35. Оптимальные управление и траектория в примере 4.8 (PSS, решение 2)

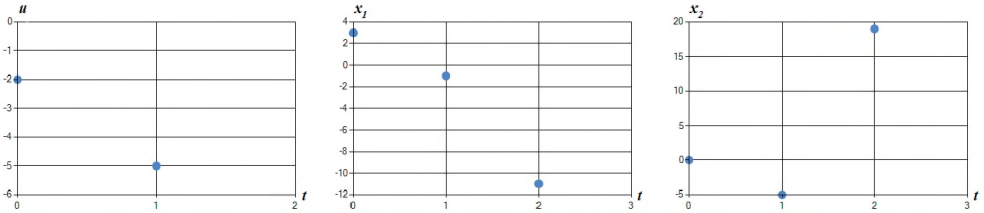


Рис. 4.36. Оптимальные управление и траектория в примере 4.8 (TFO, решение 2)

Величины полученных обоими методами значений функционала качества: $I^{2*}_{PSS} = -18,9999999999563$, $I^{2*}_{TFO} = -18,9999999993527$. Точное минимальное значение функционала: $I_{\min} = -19$.

Вариант 2 (анализ влияния параметров алгоритма TFO на точность и время расчетов). Изменим параметры метода TFO по сравнению с представленными в табл. 4.43 и зададим $\varepsilon = 10^{-9}$. С параметрами метода TFO из табл. 4.44 величины функционала $I^{1*}_{TFO} = -18,999999999918$, $I^{2*}_{TFO} = -18,9999999991551$, а время расчетов $t^*_{TFO} = 0,01$ сек. Полученные оптимальные управление и траектория представлены в табл. 4.45, 4.46.

Таблица 4.44. Параметры метода TFO, вариант 2

NP	γ	η	α	λ	P	K	L	μ	h	c_1	c_2	c_3	ρ
30	0,81	0,4	0,01	2	30	20	4	2,5	0,1	2,5	2,7	2,8	3

Таблица 4.45. Оптимальные управление и траектория (TFO), решение 1

t	u	x_1	x_2
0	−2,00001	3	0
1	5	−1,00001	−4,99998
2	—	8,99999	19

Таблица 4.46. Оптимальные управление и траектория (TFO), решение 2

t	u	x_1	x_2
0	−1,99998	3	0
1	−5	− 0,99997	−4,99994
2	—	− 10,99997	19,00000

В ходе проведения сравнительного анализа методов TFO и PSS при решении примеров 4.1 – 4.8 установлено, что оба метода позволяют получить решения приемлемого качества за допустимое время.

Метод PSS, имитирующий поведение стаи окуней, в большинстве задач дает решение медленнее метода стаи синиц, однако получаемые решения, как правило, оказываются более точными. Существенным отличием метода TFO является преобразование части популяции на каждой итерации с помощью закона распределения Леви, что помогает избегать нахождения только локальных экстремумов.

Метод стаи синиц позволяет получать решения за более короткий промежуток времени, но для приемлемого качества результата необходимо точно подбирать параметры, что является довольно сложной задачей.

Таким образом, установлено, что оба метода имеют свои преимущества и недостатки, следовательно, можно утверждать, что каждый из методов эффективен в достаточной мере при поиске условного глобального экстремума для рассмотренного класса задач оптимального управления.

Анализ решений примеров 4.1–4.8 показал, что в примерах, имеющих точное решение, и мультиагентные, и биоинспирированные методы оптимизации, предложенные авторами, позволяют найти ответ с высокой точностью. При быстром росте размерности задачи точность получаемого результата незначительно падает, но решение и в этих случаях может быть найдено, в то время, как применение необходимых условий оптимальности становится чрезвычайно затруднительным и требует также применения соответствующих численных методов.

Для несепарабельных задач оптимального управления, аналитическое решение которых с применением необходимых условий оптимальности найти не удается, рассмотренные метаэвристические методы оптимизации находят решение достаточно высокого качества по сравнению с наилучшим известным решением при очень небольших вычислительных затратах.

4.4. ПОИСК ОПТИМАЛЬНОГО В СРЕДНЕМ ПРОГРАММНОГО УПРАВЛЕНИЯ ПУЧКАМИ ТРАЕКТОРИЙ ДИСКРЕТНЫХ ДЕТЕРМИНИРОВАННЫХ ДИНАМИЧЕСКИХ СИСТЕМ

4.4.1. Постановка задачи

Поведение нелинейной детерминированной дискретной модели объекта управления описывается разностным уравнением

$$x(t+1) = f(t, x(t), u(t)), t = 0, 1, ..., N-1, \tag{4.22}$$

где t – дискретное время; x – вектор состояния системы, $x \in R^n$; u – вектор управления, $u \in U \subseteq R^q$; U – множество допустимых значений управления, для каждого значения t представляющее собой прямое произведение отрезков $[a_i, b_i]$, $i = 1, ..., q$; $f(t, x, u) = (f_1(t, x, u), ..., f_n(t, x, u))^{\mathrm{T}}$ – непрерывная вектор-функция; число шагов дискретного времени N задано. Правый конец траектории $x(N)$ свободен.

Начальное состояние задано компактным множеством положительной меры:

$$x(0) = x_0 \in \Omega \subset R^n. \tag{4.23}$$

Предполагается, что при управлении используется информация только о дискретном времени t, т.е. применяется программное управление.

Множество допустимых управлений $\mathcal{U}_0$ образуют последовательности $u(\cdot) = \{u(0), u(1), ..., u(N-1)\}$, где $u(t) \in U \ \ \forall t \in \{0, 1, ..., N-1\}$.

Множество допустимых процессов $\mathcal{D}(0, x_0)$ – это множество пар $d = (x(\cdot), u(\cdot))$, включающих траекторию $x(\cdot) = \{x_0, x(1), ..., x(N)\}$ и допустимое управление $u(\cdot) = \{u(0), u(1), ..., u(N-1)\} \in \mathcal{U}_0$, удовлетворяющих уравнению состояния (4.22) и начальному условию (4.23).

На множестве $\mathcal{D}(0, x_0)$ определен функционал качества управления отдельной траекторией

$$I(x_0, d) = \sum_{t=0}^{N-1} f^0\left(t, x(t), u(t)\right) + F\left(x(N)\right) \tag{4.24}$$

или

$$I(x_0, d) = F(x(0), ..., x(N); u(0), ..., u(N-1)), \tag{4.25}$$

где $f^0\left(t, x, u\right)$, $F\left(x\right)$, $F(x(\cdot), u(\cdot))$ – заданные непрерывные функции, траектория $x(\cdot) = \{x_0, x(1), ..., x(N)\}$ и управление $u(\cdot) = \{u(0), u(1), ..., u(N-1)\}$. Функционал (4.25) соответствует случаю несепарабельных критериев.

Каждому допустимому управлению $u(\cdot) \in \mathcal{U}_0$ и множеству Ω поставим в соответствие пучок траекторий системы уравнений (4.22):

$$X\left(t, u(\cdot)\right) = \cup\left\{x\left(t, u(\cdot), x_0\right) \mid x_0 \in \Omega\right\}, \tag{4.26}$$

т.е. объединение решений $x\left(t, u(\cdot), x_0\right)$ системы уравнения (4.22) по всем возможным начальным состояниям (4.23).

Качество управления пучком траекторий предлагается оценивать величиной функционала

$$J[u(\cdot)] = \int_{\Omega} I(x_0, u(\cdot))dx_0 / \operatorname{mes}\Omega \ , \tag{4.27}$$

где $\operatorname{mes}\Omega$ – мера множества Ω.

Требуется найти такое управление $u^*(\cdot) \in \mathcal{U}_0$ что

$$J\left[u^*(\cdot)\right] = \min_{u(\cdot)\in\mathcal{U}_0} J\left[u(\cdot)\right]. \tag{4.28}$$

Искомое управление называется оптимальным в среднем, поскольку минимизируется среднее значение функционала (4.24) или (4.25) на множестве начальных состояний.

4.4.2. Алгоритм решения задачи

Сравним поставленную задачу оптимального управления с формулировкой общей задачи минимизации (1.1) с целевой функцией $f(x) = f(x_1, x_2, ..., x_n)$, определенной на множестве допустимых решений $D \subseteq R^n$ параллелепипедного типа, где $D = \{x \mid x_i \in [a_i, b_i],\ i = 1, 2, ..., n\}$.

В качестве вектора независимых переменных $x = (x_1, ..., x_n)^T$ используется вектор $(u(0), u(1), ..., u(N-1))^T$, координаты которого удовлетворяют ограничениям вида $u(t) \in U \subseteq R^q$, где $U = \{u \mid u_i \in [a_i, b_i], i = 1, ..., q\}$, а в качестве целевой функции – функционал (4.27).

Предполагается, что множество начальных состояний Ω представляет собой параллелепипед, определенный прямым произведением отрезков $[\alpha_i, \beta_i], i = 1, ..., n$, т.е. $\Omega = [\alpha_1, \beta_1] \times \cdots \times [\alpha_n, \beta_n]$. Все отрезки с помощью шага Δx_i разбиваются на K_i отрезков, а параллелепипед Ω делится на $K = K_1 \cdots K_n$ элементарных непересекающихся подмножеств $\Omega_k, k = 1, ..., K$. В каждом элементарном подмножестве Ω_k задается начальное состояние $x_0^{\,k}$ (центр параллелепипеда Ω_k). Эти начальные состояния и применяемое управление порождают K траекторий, на которых вычисляется значение функционала (4.24) или (4.25).

Алгоритм решения задачи выглядит следующим образом.

1. Выбрать мультиагентный или биоинспирированный метод оптимизации и задать его параметры.

2. Реализовать заданное число итераций выбранным методом. Для вычисления значений $I(x_0^k, u(\cdot))$ функционала (4.24) или (4.25) следует K раз последовательно решить уравнение (4.22) с начальными условиями $x_0^{\,k}, k = 1, ..., K$. Затем усреднить полученные результаты, т.е. найти приближенное значение функционала (4.27) по формуле

$$J = \frac{\sum_{k=1}^{K} I(x_0^k, u(\cdot))}{K} \ . \tag{4.29}$$

3. В результате найти управление $u^*(\cdot) = \{u^*(0), u^*(1), ..., u^*(N-1)\}$ и соответствующий ему пучок траекторий (образованный K реализациями).

4.4.3. Модельные примеры

Приведенные далее решения примеров 4.9 и 4.10 получены методами SOMA и MSOMA, описанными в разд. 1.7, и методом MPCMA, описанным в разд. 1.8. Они реализованы с помощью программного обеспечения Jupiter notebook на языке Python. Использовался компьютер MacBook Pro с 16 GB оперативной памяти и процессором M1 Pro с частотой 3,2 GHz.

Пример 4.9. Поведение модели объекта управления описывается разностным уравнением:

$$x(t+1) = x(t) + u(t),$$

где $x \in R$, $t = 0,1,\ldots, N-1$, ограничение на управление: $-2 \cdot 10^4 \le u \le 0$.

Начальные состояния определяются из условия: $x(0) \in \Omega = [-200; 200]$.

Функционал (4.24) имеет вид

$$I(x_0, u(\cdot)) = \frac{1}{2} \sum_{t=0}^{N-1} \gamma^{-t} u^2(t) + x(N), \quad \gamma > 1.$$

Требуется найти управление, минимизирующее значение функционала (4.27):

$$J[u(\cdot)] = \int_{\Omega} I(x_0, u(\cdot))\, dx_0 \,/\, \mathrm{mes}\,\Omega \to \min.$$

Зададим значение параметра $\gamma = 1{,}1$, число шагов дискретного времени $N = 50$, число реализаций $K = 5$.

Результаты работы алгоритма SOMA (параметры приведены в табл. 4.47): полученное значение функционала качества $\min J = -581{,}8247$, время расчетов 520,5289 сек. На рис. 4.37 показаны полученные оптимальное управление и траектории пучка.

Таблица 4.47. Параметры алгоритма SOMA

Параметры				
NP	*Nstep*	*PRT*	*Migrations*	*MinDist*
1000	100	0,3	70	0,001

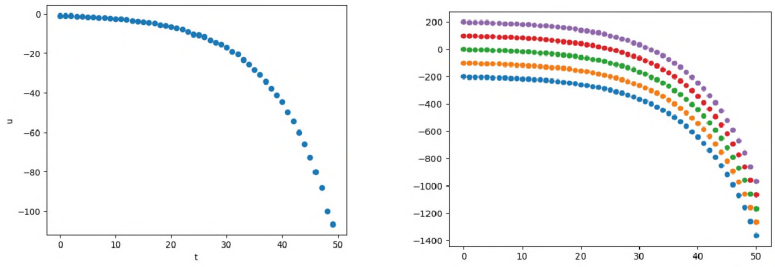


Рис. 4.37. Оптимальные управление и траектории пучка в примере 4.9, найденные при помощи SOMA

Результаты работы алгоритма MSOMA (параметры приведены в табл. 4.48): полученное значение функционала качества $\min J = -581{,}9022$, время расчетов 730,2414 сек. На рис. 4.38 показаны полученные оптимальное управление и траектории пучка.

Таблица 4.48. Параметры алгоритма MSOMA

Параметры				
NP	*Nstep*	*PRT*	*Migrations*	*MinDiv*
1000	100	0,3	150	0,001

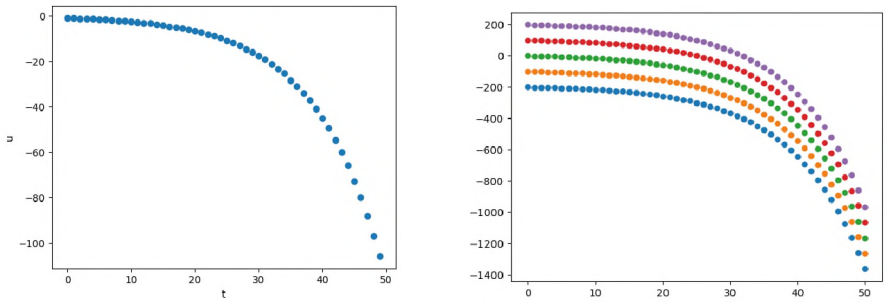


Рис. 4.38. Оптимальные управление и траектории пучка в примере 4.9, найденные при помощи MSOMA

Результаты работы алгоритма MPCMA (параметры приведены в табл. 4.49): полученное значение функционала качества $\min J = -553{,}3341$, время расчетов 2523,3303 сек. На рис. 4.39 показаны полученные оптимальное управление и траектории пучка.

Таблица 4.49. Параметры алгоритма MPCMA

Параметры								
NP	$ITER_{\max}$	*h*	$N_{\max}$	*dist*	N_p	N_c	σ%	*q*
300	5	0,01	20	0,001	25	20	30	2

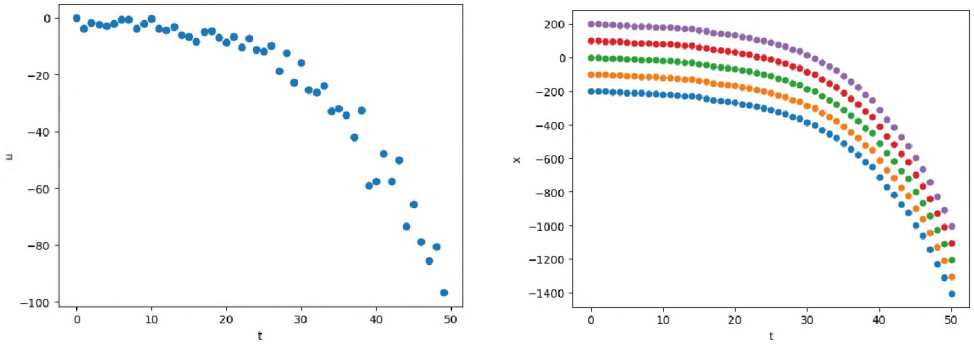


Рис. 4.39. Оптимальные управление и траектории пучка в примере 4.9, найденные при помощи MPCMA

Пример 4.10. Поведение модели объекта управления описывается системой разностных уравнений:

$$x_1(t+1) = x_2(t),$$

$$x_2(t+1) = 2x_2(t) - x_1(t) + \frac{1}{N^2}u(t),$$

где $x \in R^2$, $u \in R$. Задано ограничение на управление: $0 \le u \le 100$.

Начальные состояния определяются из условия: $x(0) \in \Omega = [-2;2] \times [-2;2]$.

Функционал (4.24) имеет вид

$$I(x_0, u(\cdot)) = -x_1(N) + \frac{1}{2N}\sum_{t=0}^{N-1} u^2(t).$$

Требуется найти оптимальное программное управление и минимальное значение функционала (4.27):

$$J[u(\cdot)] = \int_{\Omega} I(x_0, u(\cdot))\, dx_0 \,/\, \mathrm{mes}\,\Omega \to \min.$$

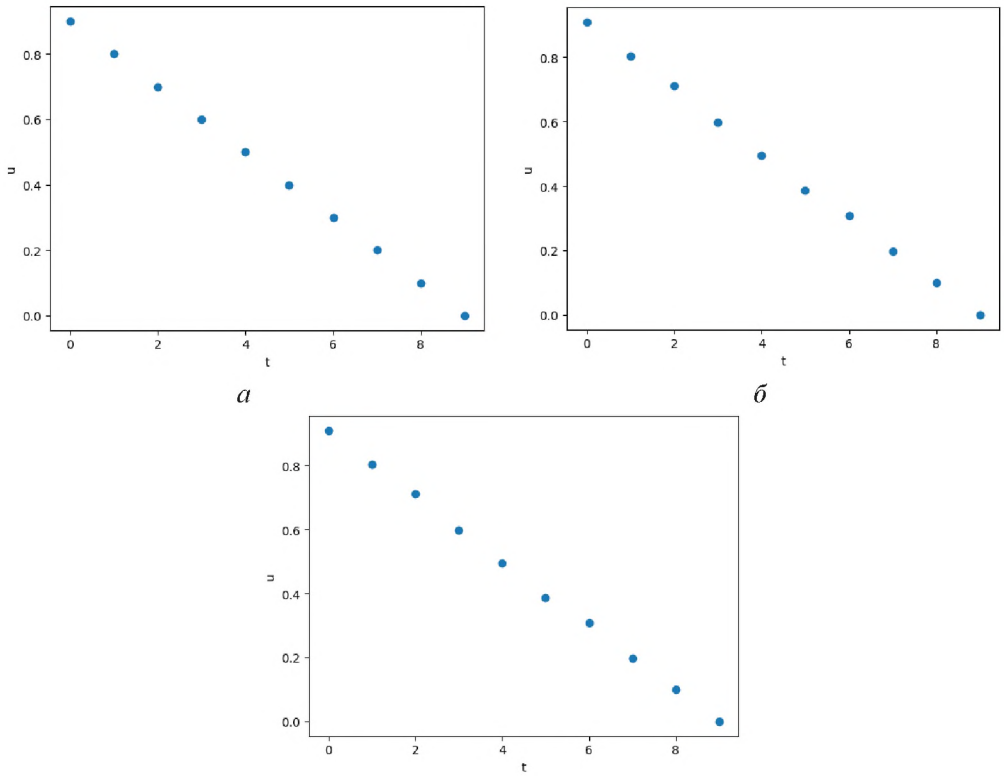
Положим $N = 10$, число реализаций $K = K_1 K_2 = 25$.

Результаты работы алгоритма SOMA (параметры приведены в табл. 4.47): полученное значение функционала качества: $\min J = -0,5424$, время расчетов 225,58 сек.

Результаты работы алгоритма MSOMA (используемые параметры указаны в табл. 4.48): полученное значение функционала качества $\min J = -0,5424$, время расчетов 868,34 сек.

Результаты работы алгоритма MPCMA (параметры приведены в табл. 4.49): полученное значение функционала качества $\min J = -0,5362$, время расчетов 1129,124 сек.

На рис. 4.40, *а–в* изображено полученное оптимальное управление тремя алгоритмами. На рис. 4.41 показаны полученные траектории пучка при решении алгоритмом SOMA, на рис. 4.42 – алгоритмом MSOMA, на рис. 4.43 – алгоритмом MPCMA.



а *б*

в

Рис. 4.40. Оптимальное управление в примере 4.10, найденное при помощи SOMA (*а*), MSOMA (*б*), MPCMA (*в*)

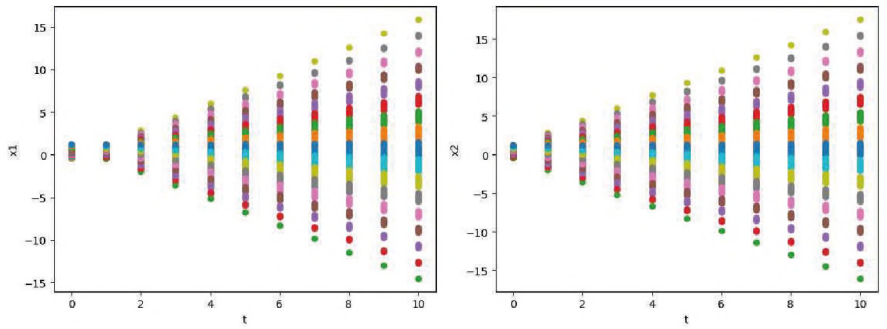


Рис. 4.41. Пучок траекторий в примере 4.10, найденный при помощи SOMA

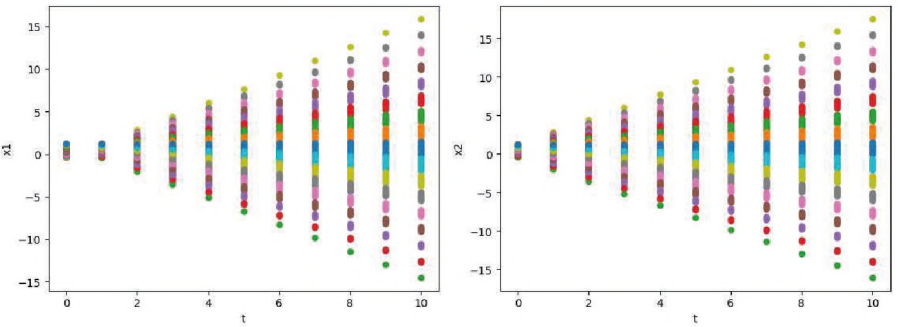


Рис. 4.42. Пучок траекторий в примере 4.10, найденный при помощи MSOMA

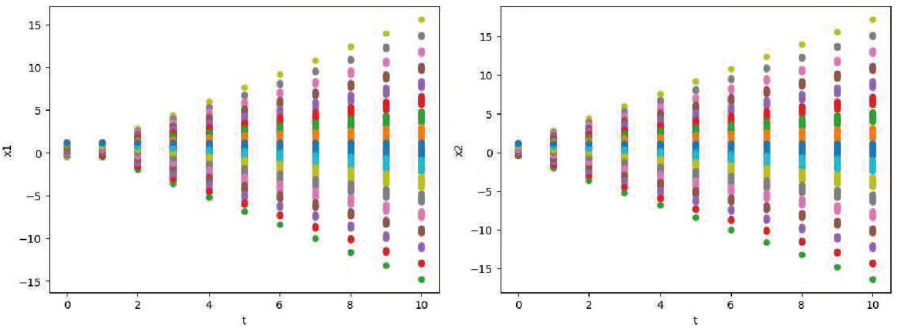


Рис. 4.43. Пучок траекторий в примере 4.10, найденный при помощи MPCMA

Как следует из анализа полученных результатов, алгоритмами SOMA и MSOMA получен фактически одинаковый результат. Относительное отклонение по величине функционала результата работы MPCMA составляет примерно 1 %, что свидетельствует о том, что он незначительно уступает первым двум. В целом характер поведения пучка траекторий один и тот же. Можно сделать вывод о том, что с помощью оптимального в среднем программного управления не удается сфокусировать траектории пучка. Этот недостаток может быть устранен с помощью синтеза систем управления с полной и неполной обратными связями.

4.5. ПОИСК ОПТИМАЛЬНОГО В СРЕДНЕМ ПРОГРАММНОГО УПРАВЛЕНИЯ ДИСКРЕТНЫМИ СТОХАСТИЧЕСКИМИ ДИНАМИЧЕСКИМИ СИСТЕМАМИ

4.5.1. Постановка задачи

Поведение нелинейной дискретной стохастической модели объекта управления описывается уравнением

$$X(t+1) = f(t, X(t), u(t), W(t)),\ t = 0,1,\ldots,N-1, \tag{4.30}$$

где t – дискретное время, $t \in T = \{0,1,\ldots,N\}$, число N задано; X – вектор состояния системы, $X \in R^n$; u – вектор управления, $u \in U \subseteq R^q$; U – множество допустимых значений управления, представляющее собой прямое произведение отрезков $[a_i, b_i]$, $i = 1,\ldots,q$; $f(t,x,u,w) = \left(f_1(t,x,u,w),\ldots,f_n(t,x,u,w)\right)^{\mathrm{T}} : T \times R^n \times U \times R^l \to R^n$ – непрерывная вектор-функция, $W(t)$ – l-мерный случайный вектор с известной плотностью вероятности $p(W,t)$.

Частным случаем модели (4.30) является уравнение

$$X(t+1) = g(t, X(t), u(t)) + \sigma(t, X(t))W(t)),$$

в которое случайное внешнее воздействие $W(t)$ на систему управления входит аддитивно; $g(t,x,u) : T \times R^n \times U \to R^n$ – вектор-функция размеров $(n \times 1)$, $\sigma(t,x) : T \times R^n \to R^{n \times l}$ – матричная функция размеров $(n \times l)$.

Начальные условия

$$X(0) = X_0 \tag{4.31}$$

заданы начальной плотностью вероятности $p_0(x)$.

Предполагается, что при управлении используется информация только о дискретном времени t.

Множество допустимых управлений $\mathcal{U}_0$ образуют последовательности $u(\cdot) = \{u(0), u(1), \ldots, u(N-1)\}$ такие, что $\forall t \in \{0,1,\ldots,N-1\}$ $u(t) \in U$, а функция $f\left(t, x, u(t), w\right)$ такова, что решение уравнения (4.30) с начальным условием (4.31) существует и единственно.

Определим функционал качества управления отдельной траекторией

$$I\left(X_0, u(\cdot)\right) = \sum_{t=0}^{N-1} f^0(t, X(t), u(t)) + F\left(X(N)\right), \tag{4.32}$$

где $f^0(t,x,u) : T \times R^n \times U \to R, F(x) : R^n \to R$ – заданные непрерывные функции.

Качество управления системой будем оценивать величиной функционала

$$J\left[u(\cdot)\right] = M[I\left(X_0, u(\cdot)\right)]\ . \tag{4.33}$$

Требуется найти такое управление , что

$$J\left[u^*(\cdot)\right] = \min_{u(\cdot) \in \mathcal{U}_0} J\left[u(\cdot)\right]. \tag{4.34}$$

Искомое управление называется оптимальным в среднем, поскольку минимизируется среднее значение функционала (4.32).

4.5.2. Алгоритм решения задачи

Предположим, что закон распределения начального состояния является равномерным на множестве Ω. При моделировании системы (4.30) будем использовать K реализаций.

Алгоритм решения задачи выглядит следующим образом.

1. Выбрать мультиагентный или биоинспирированный метод оптимизации и задать его параметры.

2. Реализовать заданное число итераций выбранным методом. Для вычисления значений $I(X_0^k, u(\cdot))$ функционала (4.32) следует K раз последовательно решить уравнение (4.30) с начальными условиями $X_0{}^k, k = 1,...,K$, которые генерируются при помощи равномерного закона распределения на заданном множестве Ω. Затем усреднить полученные результаты, т.е. найти приближенное значение функционала (4.33) по формуле

$$J = \frac{\sum_{k=1}^{K} I(X_0^k, u(\cdot))}{K}.$$

3. В результате найти управление $u^*(\cdot) = \{u^*(0), u^*(1), ..., u^*(N-1)\}$ и соответствующее ему семейство траекторий, образованное K реализациями.

4.5.3. Модельные примеры

Решения модельных примеров 4.11 и 4.12 получены методами SOMA и MSOMA, описанными в разд. 1.8. Они реализованы с помощью программного обеспечения Jupiter notebook на языке Python. Использовался компьютер MacBook Pro с 16 GB оперативной памяти и процессором M1 Pro с частотой 3,2 GHz.

Пример 4.11. Поведение модели объекта управления описывается разностным уравнением:

$$X(t+1) = X(t) + u(t) + \sigma W(t),$$

где $X \in R$, $t = 0,1,...,N-1$, ограничение на управление: $-2 \cdot 10^4 \le u \le 0$, $W(t)$ – гауссовская случайная величина с нулевым математическим ожиданием и единичной дисперсией, $\sigma = 20$.

Начальные состояния заданы равномерным законом распределения на отрезке: $x(0) \in [-200; 200]$.

Функционал (4.32) имеет вид

$$I(X_0, u(\cdot)) = \frac{1}{2}\sum_{t=0}^{N-1} \gamma^{-t} u^2(t) + X(N), \quad \gamma > 1.$$

Требуется найти оптимальное программное управление, минимизирующее значение функционала (4.33):

$$J[u(\cdot)] = M[I(X_0, u(\cdot))].$$

Зададим число шагов дискретного времени: $N = 50$, число реализаций K = 5, величину параметра $\gamma = 1,1$.

Результаты работы алгоритма SOMA (параметры приведены в табл. 4.50): полученное значение функционала качества $\min J = -428,0443$, время расчетов 3173,5971 сек. На рис. 4.45 изображены найденные оптимальное управление и семейство траекторий.

Таблица 4.50. Параметры алгоритма SOMA

Параметры				
NP	*Nstep*	*PRT*	*Migrations*	*MinDist*
1500	100	0,4	100	10^{-5}

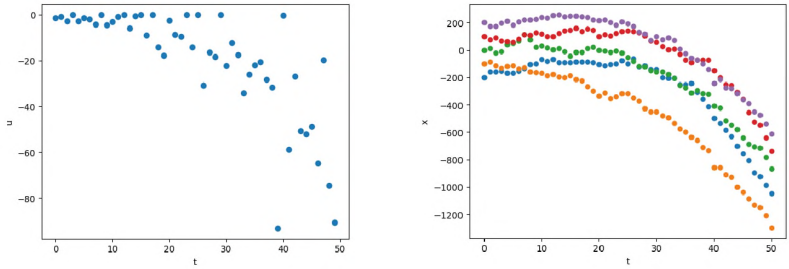


Рис. 4.45. Оптимальные управление и траектории в примере 4.11, найденные при помощи SOMA

Результаты работы алгоритма MSOMA (параметры приведены в табл. 4.51): полученное значение функционала качества $\min J = -593,329$, время расчетов 1783,9852 сек. На рис. 4.46 показаны найденные оптимальное управление и семейство траекторий.

Таблица 4.51. Параметры алгоритма MSOMA

Параметры				
NP	*Nstep*	*PRT*	*Migrations*	*MinDiv*
600	40	0,4	50	10^{-10}

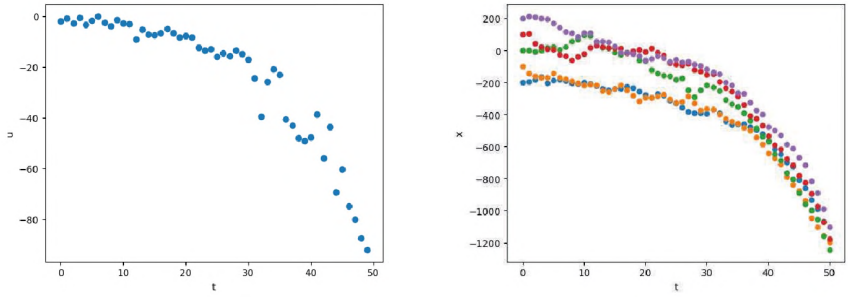


Рис. 4.46. Оптимальные управление и траектории в примере 4.11, найденные при помощи MSOMA

Анализ результатов свидетельствует, что алгоритм MSOMA в данной задаче существенно превосходит SOMA.

Пример 4.12. Поведение модели объекта управления описывается системой разностных уравнений:

$$X_1(t+1) = X_2(t) + \sigma_1 W_1(t),$$

$$X_2(t+1) = 2X_2(t) - X_1(t) + \frac{1}{N^2}u(t) + \sigma_2 W_2(t),$$

где $X \in R^2, u \in R$, $W_1(t), W_2(t)-$ независимые гауссовские случайные величины с нулевым математическим ожиданием и единичной дисперсией, $\sigma_1 = 0,02$, $\sigma_2 = 0,03$. Задано ограничение на управление: $0 \le u \le 100$.

Начальные состояния описываются равномерным законом распределения на множестве: $[-2;2]\times[-2;2]$.

Функционал (4.32) имеет вид

$$I(X_0, u(\cdot)) = -X_1(N) + \frac{1}{2N}\sum_{t=0}^{N-1} u^2(t).$$

Требуется найти оптимальное программное управление и минимальное значение функционала (4.33):

$$J\left[u(\cdot)\right] = M[I\left(X_0, u(\cdot)\right)].$$

Положим $N = 10$, число реализаций $K = 25$.

Результаты работы алгоритма SOMA (параметры приведены в табл. 4.52): полученное значение функционала качества $\min J = -0,46146$, время расчетов 8504,02 сек. На рис. 4.47, 4.48 показаны полученное оптимальное управление и траектории пучка.

Таблица 4.52. Параметры алгоритма SOMA

Параметры				
NP	*Nstep*	*PRT*	*Migrations*	*MinDist*
900	70	0,3	100	10^{-5}

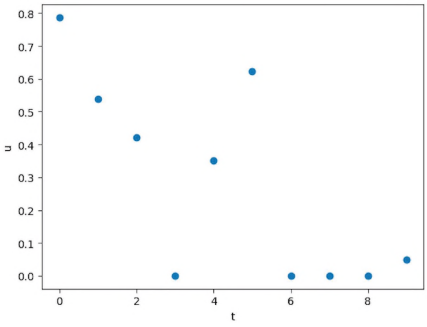


Рис. 4.47. Оптимальное управление в примере 4.12, найденное при помощи SOMA

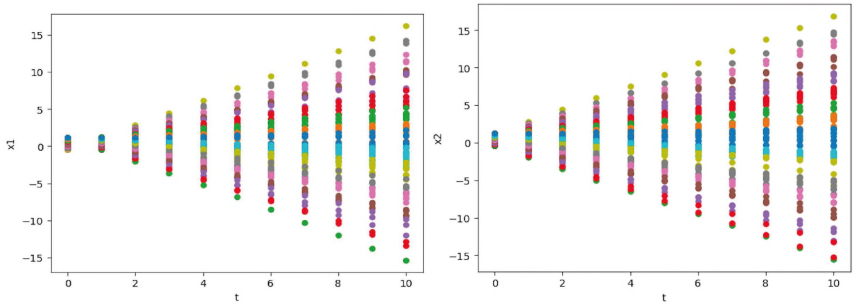


Рис. 4.48. Графики изменения координат вектора состояния в примере 4.12, найденные при помощи SOMA

Результаты работы алгоритма MSOMA (параметры указаны в табл. 4.53): полученное значение функционала качества $\min J = -0,527295$, время расчетов 14270,38 сек.

На рис. 4.49 , 4.50 показаны полученные оптимальное управление и траектории дискретной системы.

Таблица 4.53. Параметры алгоритма MSOMA

Параметры				
NP	*Nstep*	*PRT*	*Migrations*	*MinDiv*
600	50	0,3	50	10^{-10}

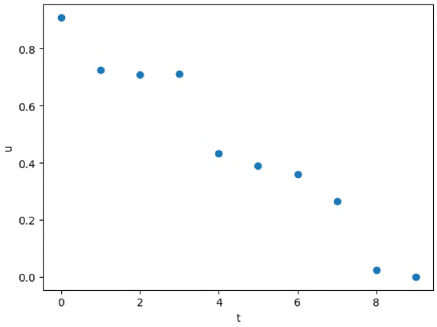


Рис. 4.49. Оптимальное управление в примере 4.12, найденное при помощи MSOMA

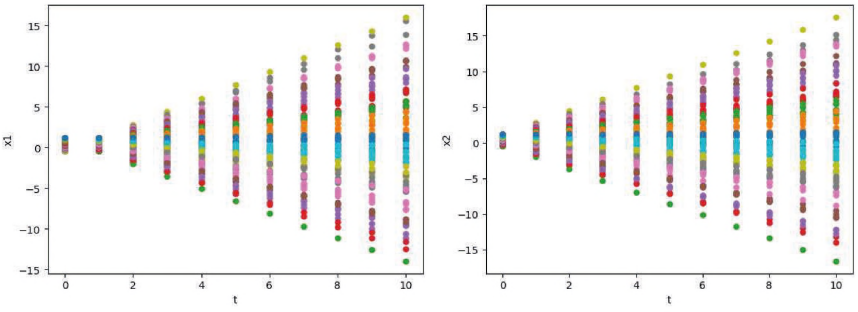


Рис. 4.50. Графики изменения координат вектора состояния в примере 4.12, найденные при помощи MSOMA

В обоих рассмотренных примерах результаты, полученные алгоритмом MSOMA, превосходят найденные SOMA. Это свидетельствует о том, что для задач достаточно большой размерности модифицированный миграционный алгоритм MSOMA эффективнее базового (SOMA).

В целом, следует сделать вывод о том, что оптимальное в среднем программное управление как пучками траекторий (разд. 4.4), так и стохастическими системами, не позволяет получать пучки или семейства траекторий с небольшим разбросом значений координат вектора состояния. Это свидетельствует о необходимости синтеза систем управления с обратной связью, что является более сложной задачей. Одним из путей развития применения мультиагентных и биоинспирированных методов оптимизации является использование различных способов параметризации законов управления, описанных в разд. 3.2 и 4.2, что позволит уменьшить размерность решаемых задач и, как следствие, сократить вычислительные затраты.

ПРИЛОЖЕНИЕ. НАБОР ТЕСТОВЫХ ЗАДАЧ ОПТИМИЗАЦИИ

Оценку работоспособности метаэвристических методов глобальной оптимизации обычно проводят на наборе стандартных тестовых функций, который включает в себя многомерные функции со сложной структурой линий уровня и большим количеством локальных экстремумов. Поиск глобального экстремума таких функций является сложной, а зачастую и невыполнимой задачей для классических методов оптимизации.

Наиболее часто используемые для тестирования многомерные функции приведены в [102, 131, 136].

Для наглядности изображения работы метаэвриристических методов глобальной оптимизации в набор тестовых функций, приведенный в данной книге, включены функции двух переменных (табл. П.1). В набор входят:

- унимодальные функции: параболическая функция (Sphere function), функция Швефеля 2.22 (Schwefel's Problem 2.22), функция Швефеля 1.2 (Schwefel's Problem 1.2);
- унимодальные функции со сложной структурой линий уровня в виде щели: функция Розенброка (Rosenbrock's saddle), функция Букина 6 (Bukin's function 6);
- функции с одним глобальным и локальными экстремумами: двухэкстремальная функция, синусоидальная функция Швефеля (Schwefel's Sine Root function), функция Растригина (Rastrigin's function), функция Экли (Ackley's function), трехгорбая функция, функция Гриванка (Griewangk's function), функция «Кожа» (Skin function), функция «Ячейки» (Eggholder function), функция Леви (Levy's function);
- функция с глобальным экстремумом в виде иголки: функция «Западня» (Trapfall function);
- функция с одним глобальным и бесконечным числом локальных экстремумов: функция Шаффера (Schaffer's function);
- функции с несколькими глобальными экстремумами: мульти-функция (Multi function), корневая функция (Roots function), функция «Птица» (Bird function).

В главах книги, где решалась задача поиска глобального минимума, в качестве тестовых функций использовались функции из таблицы П.1, взятые со знаком «минус» ($-f(x,y)$). В таком случае глобальный минимум функции равен глобальному максимуму с противоположным знаком, а координаты точки глобального минимума совпадает с координатами точки глобального максимума.

Таблица П.1. Тестовые функции двух переменных

Название функции	Формула	Множество допустимых решений	Величина $f(x^*; y^*)$	Точки глобального максимума $(x^*; y^*)^T$
Параболическая функция (рис. П.1,*а*, П.1,*б*)	$f(x,y) = -x^2 - y^2$	$x \in [-2;2]$ $y \in [-2;2]$	0	$(0;0)^T$
	$f(x,y) = -2x^2 - xy - y^2 + 3x$		$-1,2857$	$(0,8571; -0,4285)^T$
Функция Розенброка (рис. П.2)	$f(x,y) = -a\left(y - x^2\right)^2 - (1-x)^2$	$x \in [-2;2]$ $y \in [-2;2]$	0	$(1;1)^T$
Синусоидальная функция Швефеля (рис. П.3)	$f(x,y) = x\sin(\sqrt{\lvert x \rvert}) + y\sin(\sqrt{\lvert y \rvert})$	$x \in [-500;500]$ $y \in [-500;500]$	837,9657	$(420,9687; 420,9687)^T$
Мультифункция (рис. П.4)	$f(x,y) = x\sin(4\pi x) + y\sin(4\pi y)$	$x \in [-2;2]$ $y \in [-2;2]$	4,2539	$(-1,6288; -1,6288)^T$ $(1,6288; 1,6288)^T$ $(-1,6288; 1,6288)^T$ $(1,6288; -1,6288)^T$
Корневая функция (рис. П.5)	$f(z) = \dfrac{1}{1+\lvert z^6 - 1 \rvert}, z \in C, z = x + iy$	$x \in [-2;2]$ $y \in [-2;2]$	1	$(-1;0)^T$ $(0,5; -0,866)^T$ $(-0,5; 0,866)^T$ $(1;0)^T$ $(0,5; 0,866)^T$ $(-0,5; -0,866)^T$
Функция Шаффера (рис. П.6)	$f(x,y) = \dfrac{1}{2} - \dfrac{\sin^2(\sqrt{x^2+y^2}) - 0.5}{(1+0.001(x^2+y^2))}$	$x \in [-10;10]$ $y \in [-10;10]$	1	$(0;0)^T$
Функция Растригина (рис. П.7)	$f(x,y) = (10\cos(2\pi x) - x^2) +$ $+(10\cos(2\pi y) - y^2) - 20$	$x \in [-5;5]$ $y \in [-5;5]$	0	$(0;0)^T$
Трехгорбая функция (рис. П.8)	$f(x,y) = -2x^2 + 1,05x^4 - \dfrac{1}{6}x^6 -$ $-xy - y^2$	$x \in [-5;5]$ $y \in [-5;5]$	0	$(0;0)^T$
Функция Экли (рис. П.9)	$f(x,y) = -e + 20\exp\left(-\sqrt{\dfrac{x^2+y^2}{50}}\right) +$ $+\exp\left(\dfrac{1}{2}(\cos(2\pi x) + \cos(2\pi y))\right)$	$x \in [-10;10]$ $y \in [-10;10]$	20	$(0;0)^T$

Продолжение таблицы П.1. Тестовые функции двух переменных

Функция «Птица» (рис П.10)	$f(x,y)=(x-y)^2-$ $-\sin x\exp\left(\left(1-\cos y\right)^2\right)-$ $-\cos y\exp\left(\left(1-\sin x\right)^2\right)$	$x\in[-2\pi;2\pi]$ $y\in[-2\pi;2\pi]$	106,7645	$(4,7124;3,1416)^T$ $(-1,5708;-3,1416)^T$
Функция Букина 6 (рис. П.11)	$f(x,y)=-100\sqrt{\lvert -y+0,01x^2\rvert}-$ $-0,01\lvert x+10\rvert$	$x\in[-15;5]$ $y\in[-3;3]$	0	$(-10;1)^T$
Функция Швефеля 2.22 (рис. П.12)	$f(x,y)=-\lvert x\rvert-\lvert y\rvert-\lvert x\rvert\lvert y\rvert$	$x\in[-10;10]$ $y\in[-10;10]$	0	$(0;0)^T$
Функция Швефеля 1.2 (рис. П.13)	$f(x,y)=-x^2-(x+y)^2$	$x\in[-10;10]$ $y\in[-10;10]$	0	$(0;0)^T$
Двухэкстре-мальная функция (рис. П.14)	$f(x,y)=-3x^2-4y^2-$ $-23\cos(x-0,5)$	$x\in[-6;6]$ $y\in[-6;6]$	6,4892	$(-2,0709;0)^T$
Функция Гриванка (рис. П.15)	$f(x,y)=-1-\frac{x^2+y^2}{4000}+$ $+\cos\left(\frac{x}{\sqrt{1}}\right)\cos\left(\frac{y}{\sqrt{2}}\right)$	$x\in[-600;600]$ $y\in[-600;600]$	0	$(0;0)^T$
Функция «Кожа» (рис. П.16)	$f(x,y)=\left(\cos(2x^2)-1,1\right)^2+$ $+\left(\sin\frac{x}{2}-1,2\right)^2-\left(\cos(2y^2)-1,1\right)^2+$ $+\left(\sin\frac{y}{2}-1,2\right)^2$	$x\in[-5;5]$ $y\in[-5;5]$	14,0606	$(-3,315699;-3,072485)^T$
Функция «Западня» (рис. П.17)	$f(x,y)=(\lvert\sin(\sin(\sqrt{\lvert\sin(x-1)\rvert}+$ $\sqrt{\lvert\sin(y+2)\rvert}))\rvert)^{\frac{1}{2}}$	$x\in[-5;5]$ $y\in[-5;5]$	0	$(1;-2)^T$
Функция Леви 13 (рис. П.18)	$f(x,y)=-\sin^2(3\pi x)-$ $-(x-1)^2(1+\sin^2(3\pi y))-$ $-(y-1)^2(1+\sin^2(2\pi y))$	$x\in[-10;10]$ $y\in[-10;10]$	0	$(1;1)^T$
Функция «Ячейки» (рис. П.19)	$f(x,y)=x\sin\left(\sqrt{\lvert x-y-47\rvert}\right)+$ $+(y+47)\sin\left(\sqrt{\lvert y+x/2+47\rvert}\right)$	$x\in[-512;512]$ $y\in[-512;512]$	959,6407	$(512;404,2319)^T$

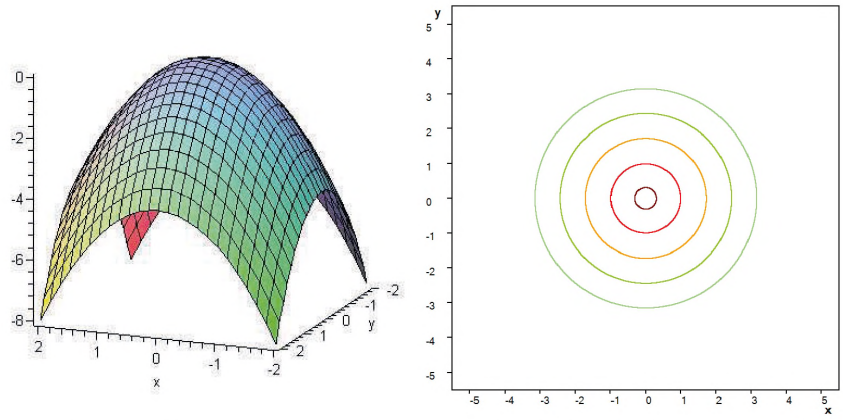


Рис. П.1, *а*. Изображения поверхности и линий уровня параболической функции

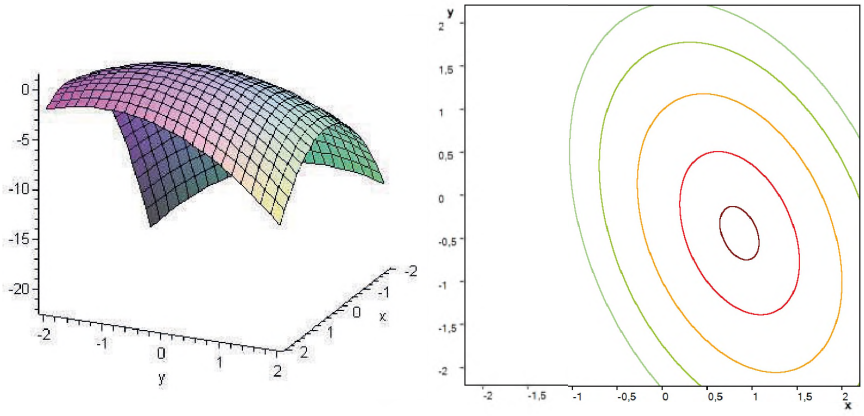


Рис. П.1, *б*. Изображения поверхности и линий уровня параболической функции

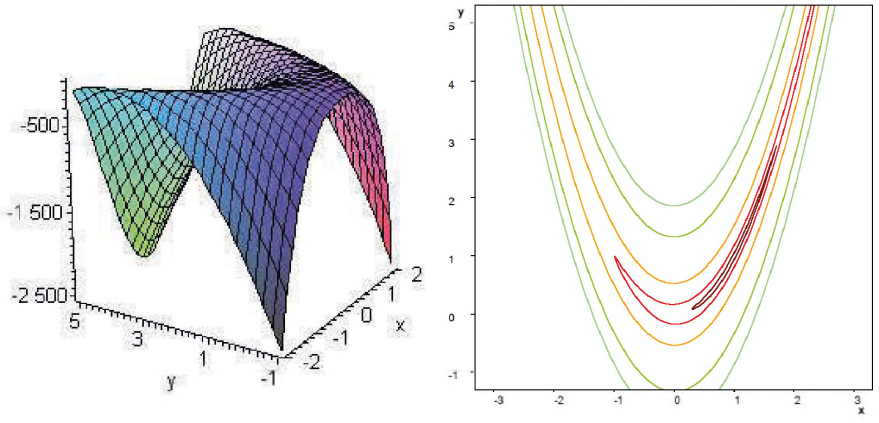


Рис. П.2. Изображения поверхности и линий уровня функции Розенброка при $a = 100$

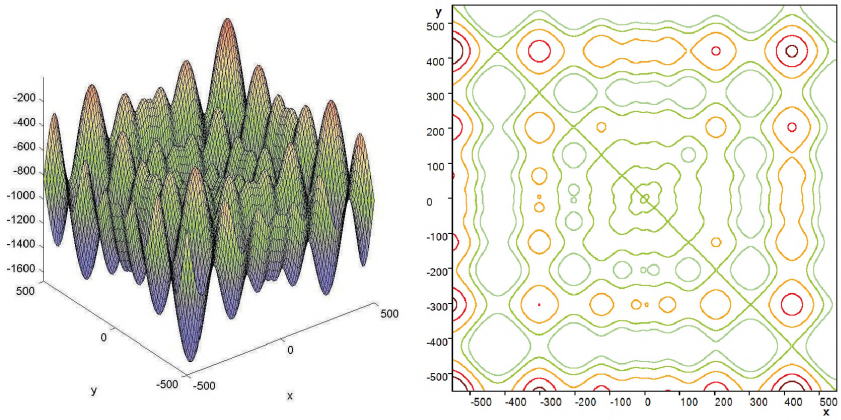


Рис. П.3. Изображения поверхности и линий уровня синусоидальной функции Швефеля

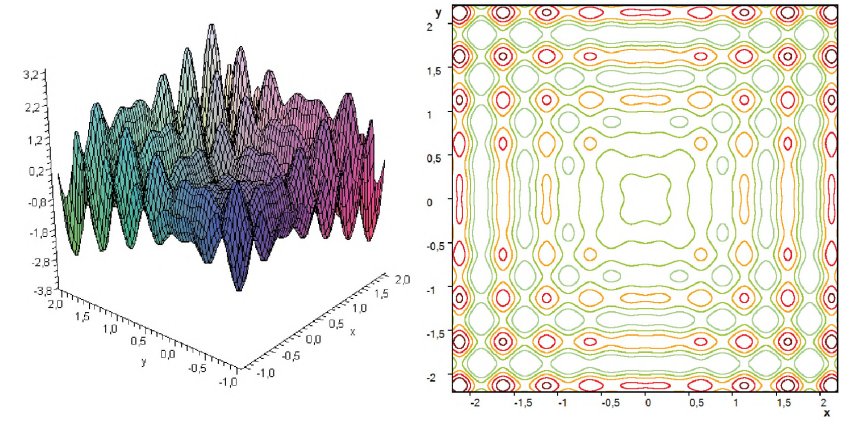


Рис. П.4. Изображения поверхности и линий уровня мульти-функции

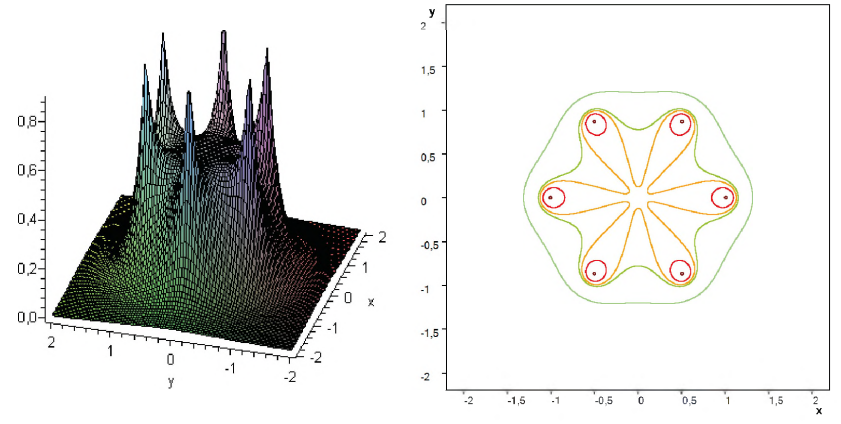


Рис. П.5. Изображения поверхности и линий уровня корневой функции

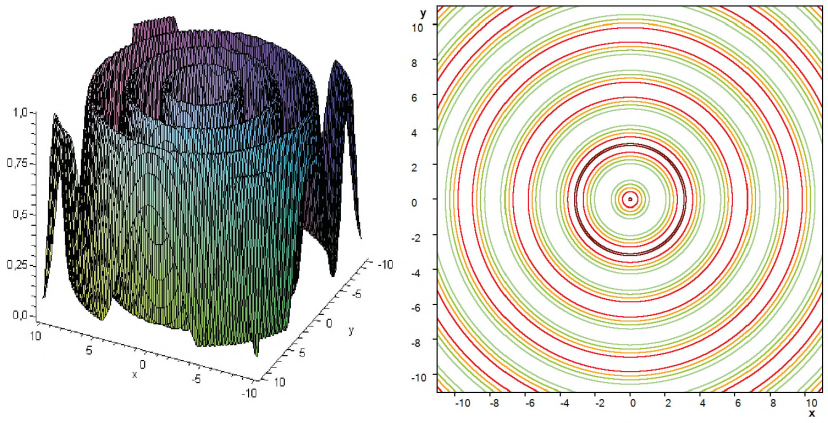


Рис. П.6. Изображения поверхности и линий уровня функции Шаффера

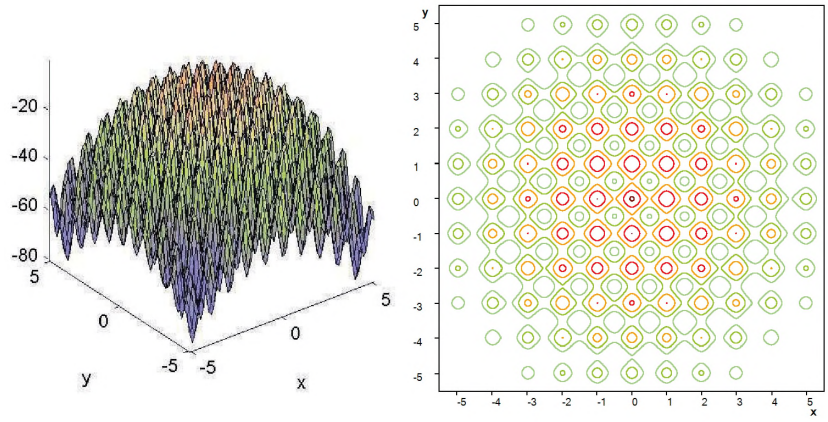


Рис. П.7. Изображения поверхности и линий уровня функции Растригина

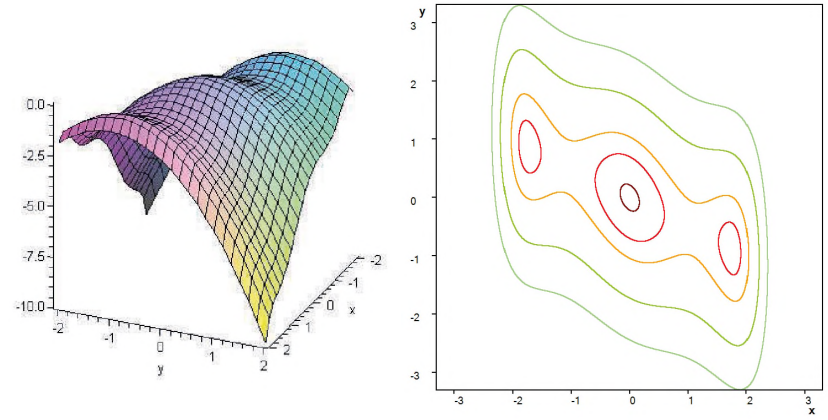


Рис. П.8. Изображения поверхности и линий уровня трехгорбой функции

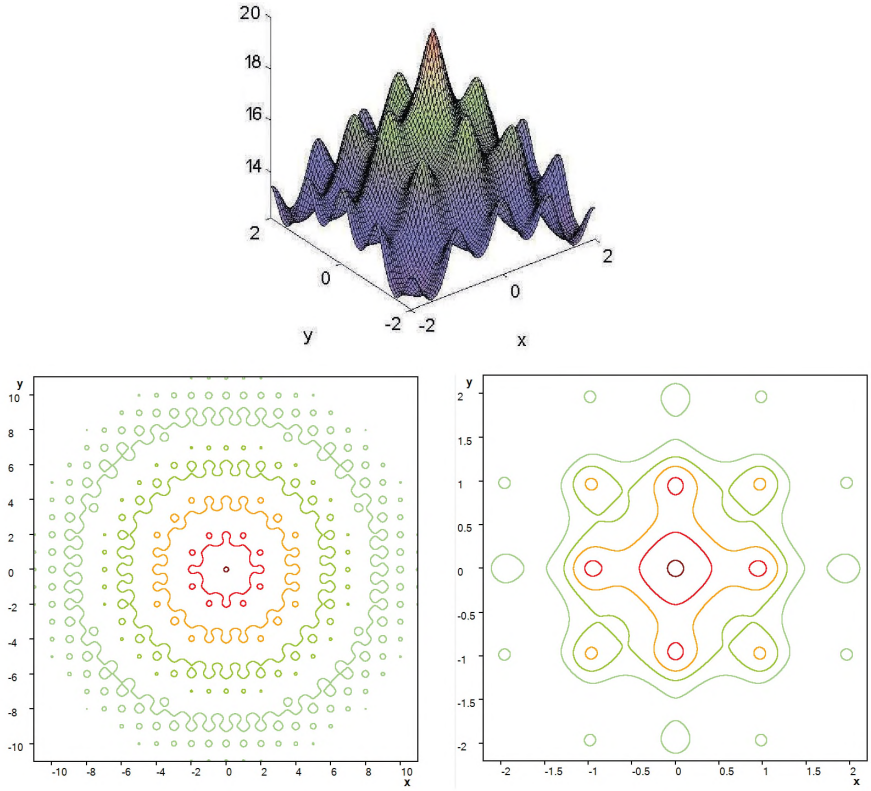


Рис. П.9. Изображения поверхности и линий уровня функции Экли

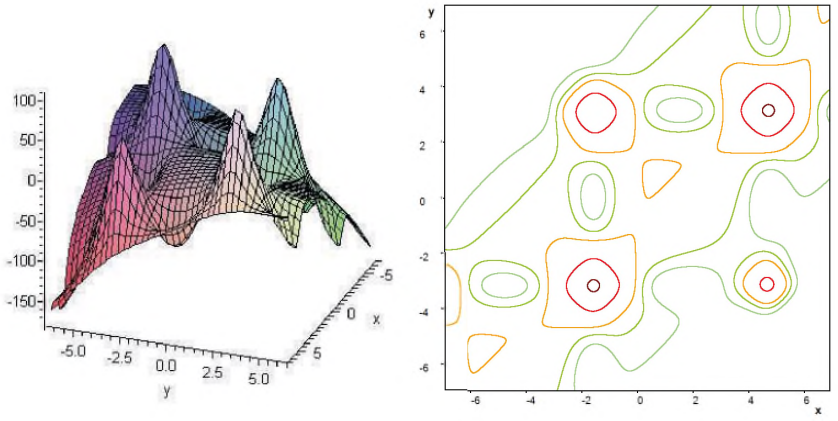


Рис. П.10. Изображения поверхности и линий уровня функции «Птица»

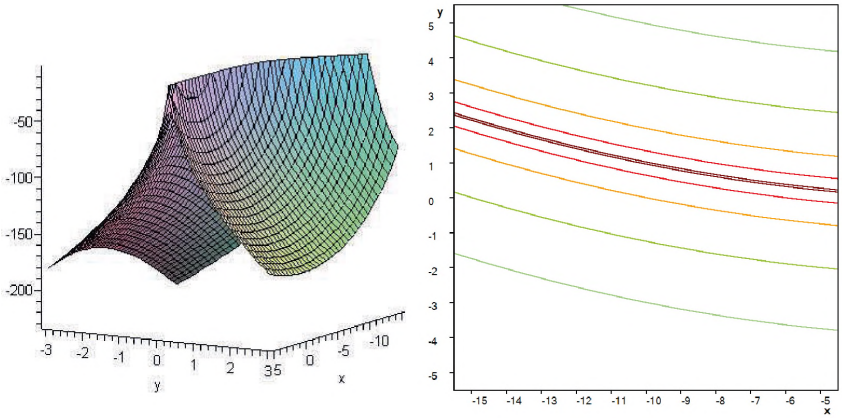


Рис. П.11. Изображения поверхности и линий уровня функции Букина 6

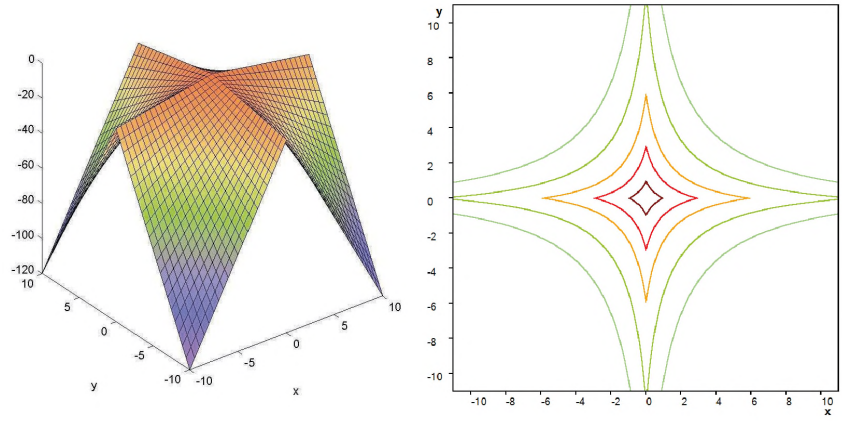


Рис. П.12. Изображения поверхности и линий уровня функции Швефеля 2.22

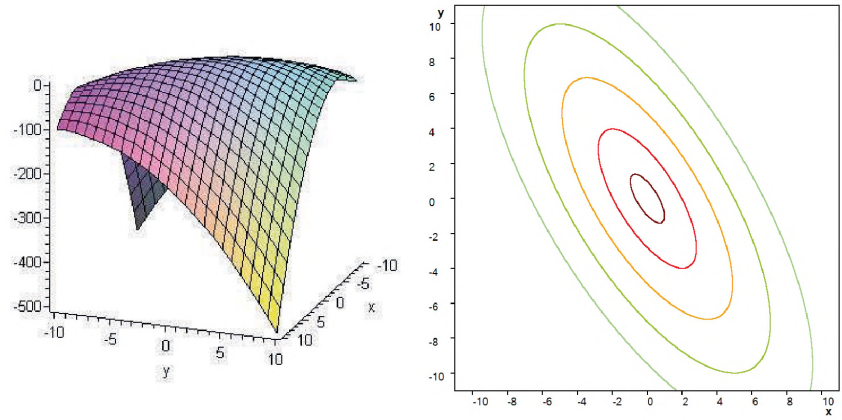


Рис. П.13. Изображения поверхности и линий уровня функции Швефеля 1.2

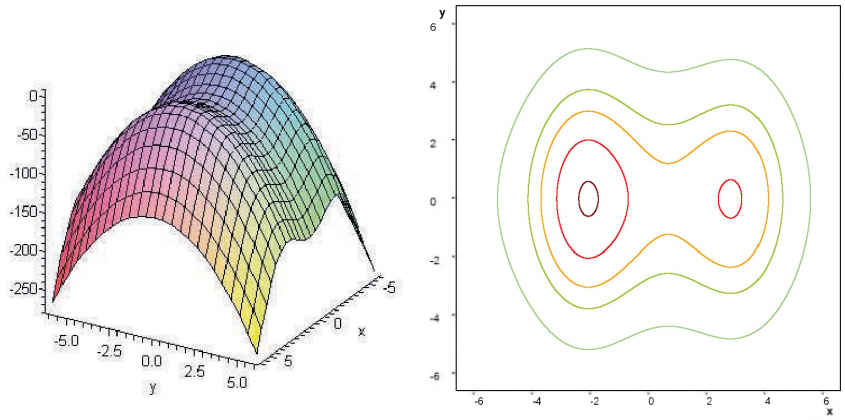


Рис. П.14. Изображения поверхности и линий уровня двухэкстремальной функции

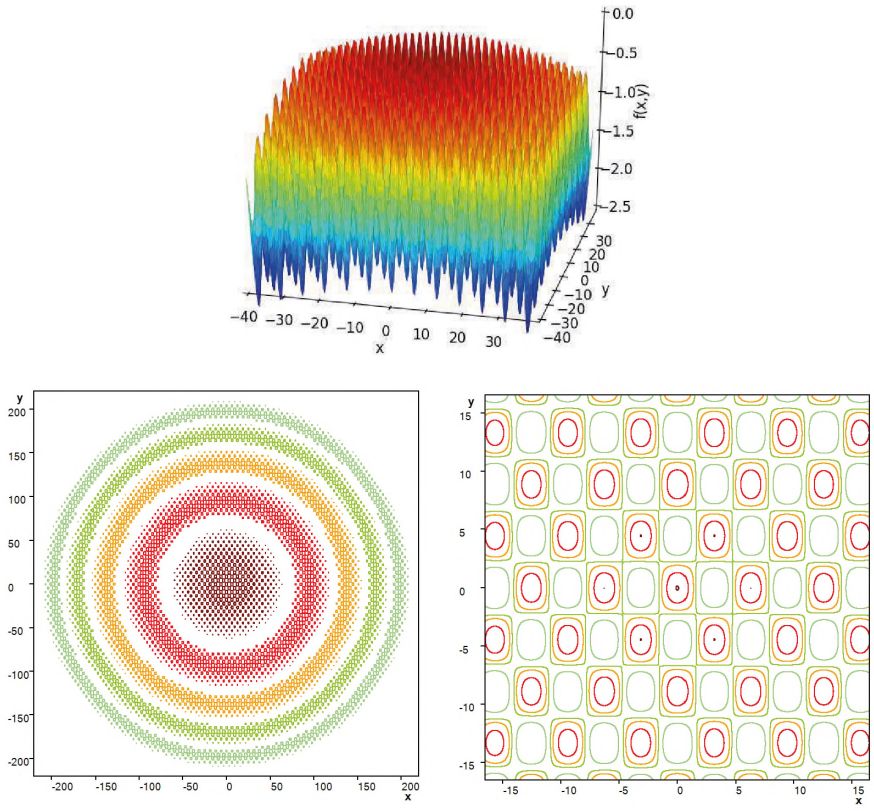


Рис. П.15. Изображения поверхности и линий уровня функции Гриванка

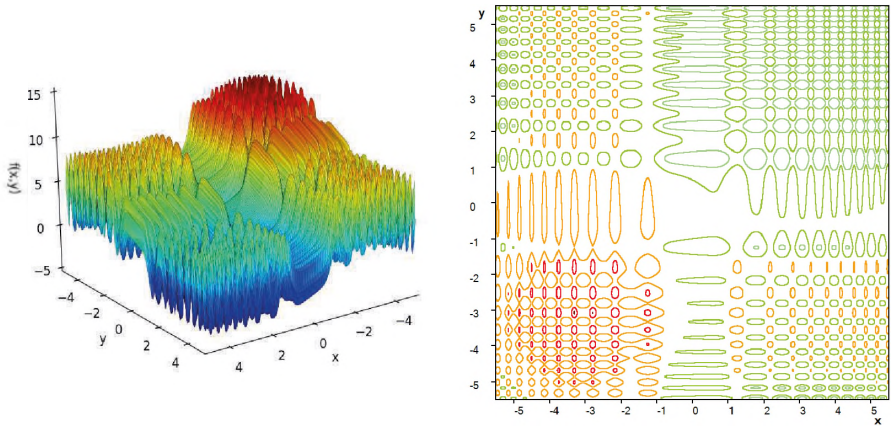


Рис. П.16. Изображения поверхности и линий уровня функции «Кожа»

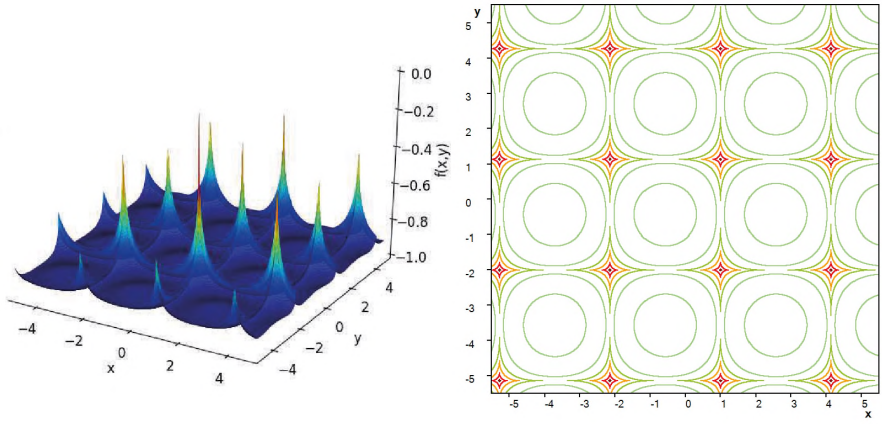


Рис. П.17. Изображения поверхности и линий уровня функции «Западня»

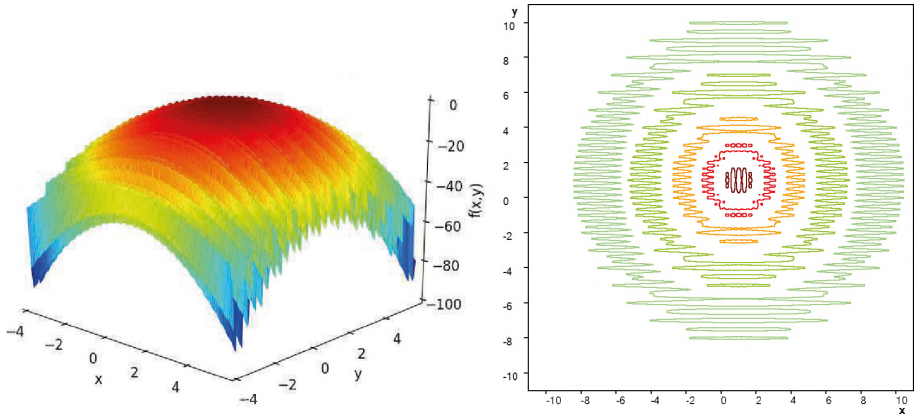


Рис. П.18. Изображения поверхности и линий уровня функции Леви 13

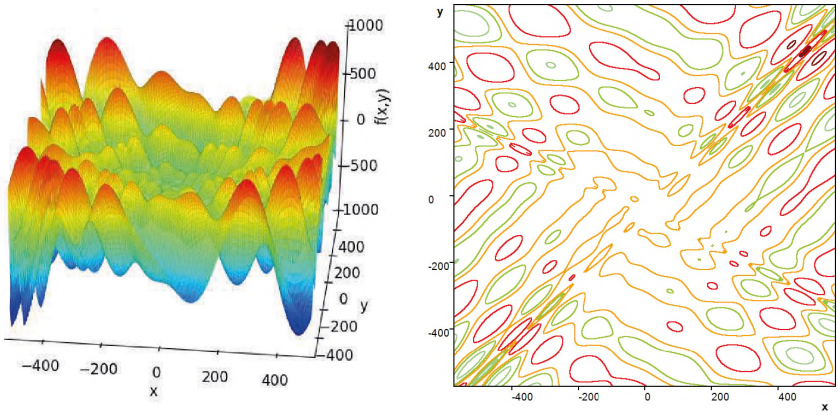


Рис. П.19. Изображения поверхности и линий уровня функции «Ячейки»

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Атанс М., Фалб П.Л.* Оптимальное управление. М.: Машиностроение, 1968.

2. *Аттетков А.В., Зарубин В.С., Канатников А.Н.* Методы оптимизации. М.: РИОР: ИНФРА–М, 2019.

3. *Базара М., Шетти К.* Нелинейное программирование. Теория и алгоритмы. М.: Мир, 1982.

4. *Бертсекас Д.* Условная оптимизация и методы множителей Лагранжа. М.: Радио и связь, 1987.

5. *Бортаковский А.С.* Оптимальное и субоптимальное управления пучками траекторий детерминированных систем автоматного типа // Изв. РАН. Теория и системы управления. 2016. №1. С. 5–26.

6. *Бортаковский А.С., Немыченков Г.И.* Оптимальное в среднем управление детерминированными переключаемыми системами при наличии дискретных неточных измерений // Изв. РАН. Теория и системы управления. 2019. №1. С. 52–57.

7. *Гасников А.В.* Современные численные методы оптимизации. Метод универсального градиентного спуска. М.: МФТИ, 2018.

8. *Гладков В.А., Курейчик В.В.* Биоинспирированные методы в оптимизации. М.: Физматлит, 2009.

9. *Горнов А.Ю.* Вычислительные технологии решения задач оптимального управления. Новосибирск: Наука, 2009.

10. *Горнов А.Ю., Зароднюк Т.С., Аникин А.С., Финкельштейн Е.А.* Практическая оптимизация в задачах оптимального управления. Материалы общероссийского семинара “Информатика, управление и системный анализ”, 28 сент. 2017. http://www.commonmind.ru.

11. *Гурман В.И.* Принцип расширения в задачах управления. М.: Наука, 1997.

12. *Давтян Л.Г., Пантелеев А.В.* Метод параметрической оптимизации нелинейных непрерывных систем совместного оценивания и управления // Изв. РАН. Теория и системы управления. 2019. № 3. С. 34–47.

13. *Дасгупта Д.* Искусственные иммунные системы и их применение. М.: Физматлит, 2006.

14. *Евтушенко Ю.Г.* Методы решения экстремальных задач и их применение в системах оптимизации. М.: Наука, 1982.

15. *Жиглявский А.А., Жилинскас А.Г.* Методы поиска глобального экстремума. М.: Наука, 1991.

16. *Каранэ М.М.С.* Программный комплекс мультиагентных алгоритмов условной оптимизации // Федеральная служба по интеллект. собственности. Св-во о гос. регистрации программы для ЭВМ № 2021662276. 2021.

17. *Каранэ М.М.С.* Программный комплекс мультиагентных алгоритмов оптимизации пучков траекторий детерминированных систем // Федеральная служба по интеллект. собственности. Св-во о гос. регистрации программы для ЭВМ № 2023617644. 2023.

18. *Каранэ М.С.* Псевдоспектральный метод поиска оптимального управления пучками траекторий на базе мультиагентных алгоритмов оптимизации // Моделирование и анализ данных. 2023. Т. 13. № 2. С. 99–122. doi: 10.17759/mda.2023130206.

19. *Каранэ М.М.С., Пантелеев А.В.* Мультиагентные алгоритмы оптимизации пучков траекторий детерминированных систем с неполной мгновенной обратной связью // Изв. РАН. Теория и системы управления. 2022. №5. С. 5–75.

20. *Карпенко А.П.* Современные алгоритмы поисковой оптимизации. Алгоритмы, вдохновленные природой. М.: Изд-во МГТУ им. Н.Э. Баумана, 2021.

21. *Киреев В.И., Пантелеев А.В.* Численные методы в примерах и задачах. СПб.: Издательство «Лань», 2015.

22. *Кротов В.Ф., Гурман В.И.* Методы и задачи оптимального управления. М.: Наука, 1973.

23. *Крылов И.А.* Численное решение задачи об оптимальной стабилизации спутника // ЖВМ и МФ. 1968. Т.8. №1. С. 203–208.

24. *Куржанский А.Б.* Управление и наблюдение в условиях неопределенности. М.: Наука, 1977.

25. *Куржанский А.Б., Месяц А.И.* Управление эллипсоидальными траекториями. Теория и вычисления // ЖВМ и МФ. 2014. Т. 54. № 3. С. 404–414.

26. *Летов А.М.* Динамика полета и управление. М.: Наука, 1969.

27. *Немировский А.С., Юдин Д.Б.* Сложность задач и эффективность методов оптимизации.– М.: Наука, 1979.

28. *Нестеров Ю.Е.* Эффективные методы в нелинейном программировании. М.: Радио и Связь, 1989.

29. *Нестеров Ю.Е.* Методы выпуклой оптимизации. М.: Изд-во МЦНМО, 2010.

30. *Овсянников Д.А.* Математические методы управления пучками. Л.: Изд-во ЛГУ, 1980.

31. *Овсянников Д.А.* Моделирование и оптимизация динамики пучков заряженных частиц. Л.: Изд-во ЛГУ, 1990.

32. *Овсянников Д.А., Егоров Н.В.* Математическое моделирование систем формирования электронных ионных пучков. СПб: Изд-во СПбГУ, 1998.

33. *Овсянников Д.А., Мизинцева М.А., Балабанов М.Ю., Дуркин А.П., Едаменко Н.С., Котина Е.Д., Овсянников А.Д.* Оптимизация динамики пучков траекторий с использованием гладких и негладких функционалов. Ч. 1 // Вестник СПбГУ. Сер. 10. Прикладная математика. Информатика. Процессы управления. 2020. Т. 16. №1. С. 73–84.

34. *Пановский В.Н., Пантелеев А.В.* Метаэвристические интервальные методы поиска оптимального в среднем управления нелинейными детерминированными системами при неполной информации о ее параметрах // Изв. РАН. Теория и системы управления. 2017. №1. С. 53–64.

35. *Пантелеев А.В.* Вариационное исчисление в примерах и задачах. М.: Высшая школа, 2006.

36. *Пантелеев А.В.* Метаэвристические алгоритмы оптимизации законов управления динамическими системами. М.: Факториал, 2020.

37. *Пантелеев А.В.* Применение методов «роевого» интеллекта в задачах оптимального в среднем управления нелинейными динамическими системами // Известия Института инженерной физики. 2019. № 3 (53). С. 89–93.

38. *Пантелеев А.В., Беляков И.А.* Разработка программного обеспечения метода глобальной оптимизации, имитирующего поведение стаи серых волков // Моделирование и анализ данных. 2021. №11. С. 59–73.

39. *Пантелеев А.В., Беляков И.А.* Применение биоинспирированных методов оптимизации в задаче оптимального программного управления солнечным парусом // Труды Московского авиационного института. 2022. № 122.

40. *Пантелеев А.В., Бортаковский А.С.* Теория управления в примерах и задачах. М.: ИНФРА–М, 2016.

41. *Пантелеев А.В., Каранэ М.М.С.* Анализ эффективности мультиагентных методов оптимизации элементов конструкций летательных аппаратов // Научный вестник МГТУ ГА. 2019. № 22(2). С. 96–108.

42. *Пантелеев А.В., Каранэ М.С.* Мультиагентный алгоритм поиска оптимального программного управления одним классом детерминированных систем // Моделирование и анализ данных. 2019. № 3. С. 58–64.

43. *Пантелеев А.В., Каранэ М.С.* Параметрический синтез оптимального программного управления на основе спектрального метода и мультиагентных алгоритмов оптимизации // Известия Института инженерной физики. 2020. №3(57). С. 74–78.

44. *Пантелеев А.В., Каранэ М.М.С.* Применение гибридного мультиагентного метода интерполяционного поиска в задаче о стабилизации спутника // Труды Московского авиационного института. 2021. № 117.

45. *Пантелеев А.В., Ковтунов С.С., Ракитянский В.М.* Методика определения свойств компонентов композиционного материала на основе миграционных алгоритмов глобальной оптимизации // Моделирование и анализ данных. 2023. Т. 13. № 2. С. 123–141. doi: 10.17759/mda.2023130207

46. *Пантелеев А.В., Летова Т.А.* Теория оптимизации для инженеров и экономистов. М.: Вузовская книга, 2016.

47. *Пантелеев А.В., Пановский В.Н.* Применение гибридного меметического алгоритма в задачах оптимального управления нелинейными стохастическими системами с неполной обратной связью // Научный вестник МГТУ ГА. 2018. Т. 21. №2. С. 59–70.

48. *Пантелеев А.В., Письменная В.А.* Применение меметического алгоритма в задаче оптимального управления пучками траекторий нелинейных детерминированных систем с неполной обратной связью // Изв. РАН. Теория и системы управления. 2018. № 1. С. 27–38.

49. *Пантелеев А.В., Ракитянский В.М.* Разработка модифицированного самоорганизующегося миграционного алгоритма оптимизации (MSOMA) // Моделирование и анализ данных. 2020. Т. 10. № 2. С. 62–73. doi:10.17759/mda.2020100205.

50. *Пантелеев А.В., Рыбаков К.А.* Прикладной вероятностный анализ нелинейных систем управления спектральным методом. М.: Изд-во МАИ-ПРИНТ, 2010.

51. *Пантелеев А.В., Рыбаков К.А.* Методы и алгоритмы синтеза оптимальных стохастических систем управления при неполной информации. М.: Изд-во МАИ, 2012.

52. *Пантелеев А.В., Семенов В.В.* Синтез оптимальных систем управления при неполной информации. М.: Изд-во МАИ, 1992.

53. *Пантелеев А.В., Скавинская Д.В., Алешина Е.А.* Метаэвристические алгоритмы поиска оптимального программного управления. М.: ИНФРА–М, 2016.

54. *Пантелеев А.В., Скавинская Д.В.* Метаэвристические стратегии и алгоритмы глобальной оптимизации. М.: Факториал, 2023.

55. *Поляк Б.Т.* Введение в оптимизацию. М.: Наука, 1983.

56. *Понтрягин Л.С., Болтянский В.Г., Гамкрелидзе Р.В., Мищенко Е.Ф.* Математическая теория оптимальных процессов. М.: Наука, 1983.

57. *Пропой А.И.* Элементы теории оптимальных дискретных процессов. М.: Наука, 1973.

58. *Сергеев Я.Д., Квасов Д.Е.* Диагональные методы глобальной оптимизации. М.: ФИЗМАТЛИТ, 2008.

59. *Тятюшкин А.И.* Многометодные алгоритмы для решения задач оптимального управления // Известия Иркутского государственного университета. Сер. Математика. Т.2. №1. 2009. С. 268–282.

60. *Федоренко Р.П.* Приближенное решение задач оптимального управления. М.: Наука, 1978.

61. *Финкельштейн Е.А.* Вычислительные технологии аппроксимации множества достижимости управляемой системы: автореф. дис. ... канд. техн. наук: 05.13.01. Иркутск: Сиб. аэрокосм. акад. им. акад. М.Ф. Решетнева, 2018.

62. *Черноусько Ф.Л.* Оценивание фазового состояния динамических систем. М.: Наука, 1988.

63. *Atashpaz-Gargari E., Lucas C.* Imperialist competitive algorithm: an algorithm for optimization inspired by imperialist competition // Proc. of IEEE Congress on Evolutionary Computation, 2007. P. 4661–4667.

64. *Bacanin N., Pelevic B., Tuba M.* Krill herd (KH) algorithm for portfolio optimization // Mathematics and Computers in Business, Manufacturing and Tourism. 2013. P. 39–44.

65. *Barty K., Roy J.-S., Strugarek C.* A stochastic gradient type algorithm for closed-loop problems // Math. Program. 2009. Vol. 119. P. 51–78.

66. *Bastos-Filho C., de Lima Neto F., Lins A., Nascimento A., Lima M.* A Novel search algorithm based on fish school behavior // 2008 IEEE Int. Conf. on Systems, Man and Cybernetics. – SMC 2008.

67. *Bastos-Filho C., de Lima Neto F., Lins A., Nascimento A., Lima M.* Fish school search: overview // In: Chiong, R. (ed.) Nature-Inspired Algorithms for Optimization. SCI, Springer, Heidelberg. 2009. V. 193. P. 261–277.

68. *Bastos-Filho C., de Lima Neto F., Lins A., Nascimento A., Lima M.* On the Influence of the swimming operators in the fish school search algorithm // IEEE Int. Conf. on Systems, Man and Cybernetics. – SMC 2009.

69. *Beheshti Z., Shamsuddin S.M.* A review of population-based meta-heuristic algorithms // Int. J. Adv. Soft Comput. Appl. 2013. Vol. 5. P. 1–35.

70. *Boyd J.P.* Chebyshev and Fourier Spectral Methods. Dover Publications, Inc., 2001.

71. *Brockett R.W.* Optimal control of the Liouville equation // AMS IP Studies in Advanced Mathematics. 2007. Vol. 39. P. 23–35.

72. *Brownlee J.* Clever Algorithms: Nature-inspired Programming Recipes. LuLu.com: Raleigh, USA , 2011.

73. *Buhmann M.D.* Radial Basis Functions: Theory and Implementations. N.Y.: Cambridge University Press, 2003.

74. *Cagnina L.C., Esquivel S.C.* Solving engineering optimization problems with the simple constrained particle swarm optimizer // Informatica. 2008. No. 32. P. 319–326.

75. *Chambers D.L.* Practical Handbook of Genetic Algorithms, Applications. Chapman & Hall/CRC: London, UK, 2001.

76. *Clerc M.* Particle Swarm Optimization. ISTE Ltd, 2006.

77. *Davendra D., Zelinka I.* Self-Organizing Migrating Algorithm. Methodology and Implementation // Studies in Computational Intelligence. Vol. 626. Springer, 2016.

78. *Del Ser J., Osaba E., Molina D., Yang X.-S., Salcedo-Sanz S., Camacho D., Das S., Suganthan P.N., Coello Coello C.A., Herrera F.* Bio-inspired computation: Where we stand and what's next. Swarm and Evolutionary Computation. 2019. Vol. 48. P. 220–250.

79. *Duarte A., Marti R., Glover F., Gortazar F.* Hybrid scatter tabu search for unconstrained global optimization // Annals of Operations Research. 2011. Vol. 183. No. 1. P. 95–123.

80. Encyclopedia of Optimization / Eds C.A. Floudas, P.M. Pardalos. N.Y.: Springer, 2009.

81. *Elbeltagi E., Hegazy T., Grierson D.* A Modified shuffled frog-leaping optimization algorithm // Structure and Infrastructure Engineering. 2007. URL: http//www.tandf.co.uk/J.s.

82. *Eusuff M.M., Lansey K.E., Pasha F.* Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization // Engineering Optimization. 2006. Vol. 38. No. 2. P. 129–154.

83. *Eusuff M.M., Lansey K.E.* Optimization of water distribution network design using the shuffled frog leaping algorithm // Journal of Water Resources Planning and Management. 2003. Vol. 129. No. 3. P. 210–225.

84. *Fahroo F., Ross I.M.* Direct trajectory optimization by a Chebyshev pseudospectral method // J. Guidance, Control, and Dynamics. 2002. Vol. 25. No. 1. P. 160–166.

85. *Fister I. Jr., Yang X.-S., Fister I., Brest J., Fister D.* A brief review of nature-inspired algorithms for optimization, ArXiv:1307.4186 [Cs], 2013.

86. *Fleming W.H., Rishel R.W.* Deterministic and Stochastic Optimal Control. Springer: New York City, 2012.

87. *Fleming W.H., Soner H.M.* Controlled Markov Processes and Viscosity Solutions. Springer-Verlag: New York, Heidelberg, Berlin, 1993.

88. *Floudas C.A. Pardalos P.M., Adjiman C.S.* Handbook of Test Problems in Local and Global Optimization. Dordrecht: Kluwer Acad. Publ., 1999.

89. *Gandomi A.H., Alavi A.H.* Krill herd: A New bio-inspired optimization algorithm // Commun. Nonlinear Sci. Numer. Simulat. 2012. Vol.17. No. 12. P. 4831–4845.

90. *Garg D., Patterson M.A., Hager W.W., Rao A.V., Benson D., Huntington G.T.* A Unified framework for the numerical solution of optimal control problems using pseudospectral methods // Automatica. 2010. Vol. 46. No. 11. P. 1843–1851.

91. *Garg D., Patterson M., Hager W., Rao A., Benson D.* An overview of three pseudospectral methods for the numerical solution of optimal control problems // Advances in the Astronautical Sciences. 2017. Vol. 135. P. 1–17.

92. *Glover F., Marti R., Laguna M.* Fundamentals of scatter search and path-relinking // Control & Cybernetics. 2000. Vol. 39. P. 653–684.

93. *Goldberg D.* Genetic algorithms in search, optimization and machine learning. Addison-Wesley Publishing Company: Boston, USA, 1989.

94. *Golinski J.* An adaptive optimization system applied to machine synthesis //Mech. Mash.Theory. 1973, No. 8(3). P. 419–436.

95. *Gong B., Liu W., Tang T., Zhao W., Zhou T.* An efficient gradient projection method for stochastic optimal control problems // SIAM J. Numer. Anal. 2017. Vol. 55(6). P. 2982–3005.

96. *Gornov A.Yu., Zarodnyuk T.S., Madzhara T.I., Daneeva A.V., Veyalko I.A.* A Collection of Test Multiextremal Optimal Control Problems / A. Chinchuluun et al. (eds.), Optimization, Simulation, and Control, Springer Optimization and Its Applications.

Vol. 76, Springer Science+Business Media New York. 2013. DOI 10.1007/978-1-4614-5131-0 16.

97. *Gustafson E.D., Scheeres D.J.* Spacecraft stochastic optimal control. AIAA/AAS Spaceflight Mechanics Meeting, No. AAS 10-109, 2010.

98. *Halder A., Bhattacharya R.* Dispersion analysis in hypersonic flight during planetary entry using stochastic Liouville equation // J. Guidance, Control, and Dynamics. 2011. Vol. 34. No. 2. P. 459–474.

99. Handbook of Memetic Algorithms / Eds F. Neri, C. Cotta, P. Moscato. N.Y.: Springer, 2012.

100. Handbook of Metaheuristics / Eds M. Gendreau, J-Y. Potvin. N.Y.: Springer, 2019.

101. Handbook of Metaheuristics / Eds F.W. Glover, G.A. Kochenberger. Boston, MA: Kluwer Acad. Publ., 2003.

102. *Hansen E., Walster G.W.* Global Optimization Using Interval Analysis / E. Hansen, G.W. Walster, New York: Marcel Dekker, 2004.

103. *Haussmann U.G.* A stochastic maximum principle for optimal control of diffusions. Pitman Research Notes in Mathematics Series 151, Longman, 1986.

104. *Issa B.A., Rashid A.T.* A survey of multi-mobile robot formation control // International Journal of Computer Applications. 2019. Vol. 181(48). P. 12–16.

105. *Jadbabaie A., Lin J., Morse A.S.* Coordination of groups of mobile autonomous agents using nearest neighbor rules // IEEE Trans. on Automatic Control. 2003. Vol. 48. P. 988–1001.

106. *Jaulin L., Kieffer M., Didrit O., Walter E.* Applied Interval Analysis. London: Springer-Velag, 2001.

107. *Jin Z., Qiu M., Tran K.Q., Yin G.* A survey of numerical solutions for stochastic control problems: some recent progress // Numerical Algebra, Control & Optimization. 2022. Vol. 12(2). P. 213–253.

108. *Karane M.M.S.* Comparative analysis of multi-agent methods for constrained global optimization // 2018 IV Intern. Conf. on Information Technologies in Engineering Education (Inforino), Moscow, 2018. P. 1–6.

109. *Karane M., Panteleev A.* Hybrid multi-agent optimization method of interpolation search // AIP Conference Proceedings (Proceedings of Computational Mechanics and Modern Applied Software Systems (CMMASS'2019), Alushta, May 24–31, 2019). – 2019. Vol. 2181. Id 020028. https://doi.org/10.1063/1.5135688.

110. *Karane M., Panteleev A.* Benchmark analysis of novel multi-agent optimization algorithm using linear regulators for agents motion control // IOP Conf. Series: Materials Science and Engineering. Alushta, 2020. / 927/ 012023. 10.1088/1757-899X/927/1/012023.

111. *Karane M., Panteleev A.* Multi-agent optimization algorithms for optimal control of trajectory pencils // Proceedings - 2021 17th International Asian School-Seminar "Optimization Problems of Complex Systems", OPCS 2021.17. 2021. P. 73-77.

112. *Karane M.M.S., Panteleev A.V.* Multi-agent algorithms for optimizing bundles of trajectories of deterministic systems with incomplete instant feedback // J. of Computer and Systems Sciences International. 2022. Vol. 61(5). P. 751–775.

113. *Kaveh A., Talatahari S.* Imperialist competitive algorithm for engineering design problems // Asian Journal of civil engineering (building and housing). 2010. Vol.11. No. 6. P. 675–697.

114. *Kennedy J., Eberhart R.* Particle Swarm Optimization // URL: http://www.engr.iupui.edu/ ~shi/ Conference/psopap4.html.

115. *Kurzhanski A.B., Filippova T.F.* On the theory of trajectory tubes – A Mathematical formalism for uncertain dynamics, viability and control, Advances in Nonlinear Dynamics and Control: A Report from Russia. Progress in Systems and Control Theory. V. 17. Boston, MA: Birkhäuser, 1993.

116. *Kushner H.J.* A partial history of the early development of continuous-time nonlinear stochastic systems theory // Automatica. 2014. Vol. 50. P. 303–334.

117. *Lakhdari I.E., Miloudi H., Hafayed M.* Stochastic maximum principle for partially observed optimal control problems of general McKean–Vlasov differential equations// Bull. Iran. Math. Soc. 2021. Vol. 47. P. 1021–1043. https://doi.org/10.1007/s41980-020-00426-1

118. *Locatelli M., Schoen F.* (Global) Optimization: Historical notes and recent developments // EURO Journal on Computational Optimization. 2021. Vol. 9, 100012.

119. *Lopez Cruz I.L., Van Willigenburg L.G., Van Straten G.* Evolutionary algorithms for optimal control of chemical processes // Proceedings of the IASTED Intern. Conference on Control Applications. 2000. P. 155–161.

120. *Luus R.* Iterative Dynamic Programming. Chapman & Hall/CRC: London, UK, 2000.

121. *Luus R., Jaakola T.H.I.* Optimization by direct search and systematic reduction of the size of search region // American Institute of Chemical Engineers Journal. 1973. Vol. 19. No 4. P. 760–766.

122. *Madeiro S., Bastos-Filho C., Lima M., Figueiredo E.* Density as the segregation mechanism in fish school search for multimodal optimization problems // INCI 2011 Second Int. Conf. On Swarm Intelligence. Springer. Lecture Notes in Computer Science. Vol. 6729. P. 563–572. 2011.

123. *Michalewicz Z., Fogel D.* How to Solve it: Modern Heuristics, 2nd ed.; Springer: New York City, USA, 2004.

124. *Mehrabian A.R., Lucas C.* A Novel numerical optimization algorithm inspired from weed colonization // Ecological Informatics. 2006. Vol. 1. P. 355–366.

125. *Moral P.D.* Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applications. Springer: New York, 2004.

126. *Nanda S.J., Panda G.* A survey on nature inspired methaheuristic algorithms for partitional clustering // Swarm and Evolutionary Computation. 2014. Vol. 16. P. 1–18.

127. *Niculina Dragoi E., Dafinescu V.* Review of metaheuristics inspired from the animal kingdom // Mathematics. 2021. Vol. 9, 2335.

128. *Panteleev A.V., Belyakov I.A., Kolessa A.A.* Comparative analysis of optimization strategies by software complex metaheuristic nature-inspired methods of global optimization // Journal of Physics: Conference Series. 2022. Vol. 2308(1), 012002.

129. *Panteleev A., Karane M.* Multi-agent optimization algorithms for optimal control of trajectory pencils // Proceedings -2021 17th International Asian School-Seminar "Optimization Problems of Complex Systems", OPCS 2021. 17. 2021. P. 73–77.

130. *Panteleev A., Karane M.* Application of multi-agent optimization methods based on the use of linear regulators and interpolation search for a single class of optimal deterministic control systems // Applied Mathematics and Computational Mechanics for Smart Applications. Singapore: Springer, 2021. https:// doi.org/ 10.1007/978-981-33-4826-4_16.

131. *Panteleev A.V., Karane M.M.S.* Multi-agent optimization algorithms for a single class of optimal deterministic control systems // Smart Innovation, Systems and Technologies. 2020. Vol. 173. chap. 20. Singapore: Springer, P. 271–291. https://doi.org/ 10.1007/978-981-15-2600-8_20.

132. *Panteleev Andrei, Karane Maria.* Application of a novel multi-agent optimization algorithm based on PID controllers in stochastic control problems // Mathematics. 2023. Vol. 11 (13), 2903, https://doi.org/10.3390/math11132903.

133. *Panteleev A.V., Kolessa A.A.* Optimal open-loop control of discrete deterministic systems by application of the perch school metaheuristic optimization algorithm // Algorithms. 2022. Vol. 15(5). 157. https://doi.org/10.3390/a15050157.

134. *Panteleev A.V., Kolessa A.A.* Application of the tomtit flock metaheuristic optimization algorithm to the optimal discrete time deterministic dynamical control problem // Algorithms. 2022. Vol. 15(9). 301. https://doi.org/10.3390/a15090301.

135. *Panteleev A.V., Letova T.A., Pomazueva E.A.* Parametric design of optimal in average fractional-order PID controller in flight control problem //Automation and Remote Control. 2017. Vol. 72. No. 9. P. 345–358.

136. *Panteleev A.V., Lobanov A.V.* Application of mini–batch metaheuristic algorithms in problems of optimization of deterministic systems with incomplete information about the state vector // Algorithms. 2021. Vol. 14(11). 332. https://doi.org/10.3390/a14110332.

137. *Panteleev A.V., Lobanov A.V.* Application of the mini-batch adaptive method of random search (MAMRS) in problems of optimal in mean control of the trajectory pencils // Journal of Physics: Conference Series. 2021. Vol. 1925. Id 012006.

138. *Panteleev A.V., Lobanov A.V.* Application of mini-batch adaptive optimization method in stochastic control problems // Smart Innovation, Systems and Technologies. 2022. Vol. 274. P. 345–361.

139. *Panteleev A., Panovskiy V.* Self-organizing interval bacterial colony evolution optimization algorithm // Advances in Intelligent Systems and Computing. 2016. Vol. 450. P. 451–463.

140. *Panteleev A., Panovskiy V.* Robust and Constrained Optimization. Methods and Applications / ed. D. Clark / Chapter 5: Application of Metaheuristic Algorithms of Global Constrained Optimization to Optimal Open Loop Control Problems. P. 149–190. Nova Science publishers, New York, 2019.

141. *Panteleev A.V., Panovskiy V.N.* Development of metaheuristic interval minimization methods for optimal program control design // Automation and Remote Control. – 2019. Vol. 80. No. 2. P. 334–347.

142. *Panteleev A.V., Rakitianskii V.M.* Application of the modified self-organizing migration algorithm MSOMA in optimal open-loop control problems // Journal of Physics: Conference Series. 2021. Vol. 1925. Id 012017.

143. *Panteleev A.V., Rakitianskii V.M.* Application of the modified self-organizing migration algorithm MSOMA in optimal open-loop control problems of switching systems // MATEC Web of Conferences 362, 01020 (2022) https://doi.org/10.1051/matecconf/202236201020.

144. *Panteleev A., Rakitianskii V.* Application of migrating optimization algorithms in problems of optimal control of discrete-time stochastic dynamical systems // Axioms. 2023. Vol. 12(11). 1014. https://doi.org/10.3390/axioms12111014

145. Practical Handbook of Genetic Algorithms. Applications / Ed. D.L. Chambers. N.Y.: Chapman and Hall; CRC Press, 2001.

146. *Press W.H., Teukolsky S.A., Vetterling W.T., Flannery B.P.* Radial Basis Function Interpolation. Numerical Recipes: The Art of Scientific Computing (3rd ed.). N.Y.: Cambridge University Press, 2007.

147. *Ragsdell K., Phillips D.* Optimal design of a class of welded structures using geometric programming // J. Eng. Ind. 1976. Vol. 98(3). P. 1021–1025.

148. *Roni Md.H.K., Rana Md.S., Pota H., Hasan Md.M., Hussain Md.S.* Recent trends in bio-inspired meta-heuristic optimization techniques in control applications for electrical systems: a review // Int. J. of Dynamics and Control. 2022. Vol. 10(2). P.1–13.

149. *Ruder S.* An overview of gradient descent optimization algorithms. arXiv:1609.04747v2 [cs.LG] 15 Jun 2017.

150. *Sangren E.* Nonlinear integer and discrete programming in mechanical design optimization // J.Mech.Des.-T.ASME. 1990. No.112(2). P. 223–229.

151. *Sergeyev Y.D., Kvasov D.E.* Deterministic Global Optimization: An Introduction to the Diagonal Approach. Springer: New York City, USA, 2017.

152. *Sergeyev Y.D., Kvasov D.E., Mukhametzhanov M.S.* On the efficiency of nature-inspired metaheuristics in expensive global optimization with limited budget // Scientific Reports. 2018. Vol. 8. 453.

153. *Slowik A., Kwasnicka H.* Evolutionary algorithms and their applications to engineering problems // Neural Computing and Applications. 2020. Vol. 32. P. 12363–12379.

154. *Tzanetos A., Fister I., Dounias G.* A comprehensive database of nature-inspired algorithms // Data in Brief. 2020. Vol. 31,105792.

155. *Yang X.S.* Nature-Inspired Metaheuristic Algorithms. Frome, UK: Luniver Press, 2010.

156. *Yang X.S., Chien S.F., Ting T.O.* Bio-inspired Computation and Optimization. Morgan Kaufmann: New York City, USA, 2015.

157. *Yang X.S., Deb S.* Cuckoo search via Levy flights // Proceedings of World Congress on Nature and Biologically Inspired Computing, 2009, I.: IEEE. P. 210–214.

158. *Yang X., Deb S.* Engineering optimization by cuckoo search // International Journal of Mathematical Modelling and Numerical Optimization. 2010. Vol. 1. No. 4. P. 330–343.

159. *Wang Gai-Ge, Gandomi A., Alavi A., Dunwei G.* A Comprehensive review of krill herd algorithm: variants, hybrids and applications. 2019. Artificial Intelligence Review. Vol. 51. P. 119–148. https://doi.org/10.1007/s10462-017-9559-1.

160. *Wolpert D.H., Macready W.G.* No free lunch theorems for optimization // IEEE Transactions on Evolutionary Computation. 1997. Vol. 1. No.1. P. 67–82.

161. *Zelinka I., Lampinen J.* SOMA–Self-organizing migrating algorithm// Proceedings of the 6th Intern. Conference on Soft Computing (Mendel 2000), Brno, Czech Republic. P. 177–187.

162. *Zelinka I., Lampinen J., Noulle L.* On the theoretical proof of convergence for a class of SOMA search algorithms // Proceedings of 7th Intern. Conference on Soft Computing (Mendel 2001), Brno, Czech Republic, P. 103–110.

ОГЛАВЛЕНИЕ

Научное издание

Пантелеев Андрей Владимирович

Каранэ Мария Магдалина Сергеевна

МУЛЬТИАГЕНТНЫЕ И БИОИНСПИРИРОВАННЫЕ МЕТОДЫ ОПТИМИЗАЦИИ ТЕХНИЧЕСКИХ СИСТЕМ

Монография

Издательство «Доброе слово и Ко»
http:/dobroeslovo.info

Макет подготовлен в ИИЦ «ПЯТЬ ПЛЮС» Института № 5 МАИ

Подписано в печать 29.01.2024.
П. л. 21. Формат 70× 100/16.
Тираж 500 экз.

ISBN 978-5-6050159-7-0