Федеральный исследовательский центр «Информатика и управление»
Российской академии наук (ФИЦ ИУ РАН)

На правах рукописи

Константинов Сергей Валерьевич

Решение задачи синтеза системы управления на основе аппроксимации множества оптимальных траекторий методом сетевого оператора

Специальность 2.3.1 —
«Системный анализ, управление и обработка информации, статистика»

Диссертация на соискание учёной степени
кандидата технических наук

Научный руководитель:
доктор технических наук, профессор
Дивеев Асхат Ибрагимович

Москва — 2022

Оглавление

Стр.

Стр.

Стр.

Введение

Начавшаяся в середине XX века третья промышленная революция характеризовалась автоматизацией производства, выполнением рутинных и типовых действий без участия человека или под его контролем в качестве оператора. Плодами третьей промышленной революции стали создание программируемых контроллеров и промышленных роботов, большой скачок в области самолетостроения и ракетостроения.

В настоящее время мы являемся свидетелями четвертой промышленной революции. Развитие вычислительной техники, технологические инновации и масштабная роботизация позволили выйти на новый виток развития автоматизированных систем управления.

В современных технических системах всё чаще наблюдается переход от систем управления, где функцию выбора управляющего воздействия выполняет человек — оператор, водитель, пилот и т.д., к системам, где функция выбора управляющего воздействия возложена на блок управления. При этом блок управления, как и в случае с управлением человеком, должен вырабатывать управляющие воздействия в соответствии с информацией о текущем состоянии системы, поступающей от приборов, датчиков или внешних источников. Реализация такого блока управления позволяет учитывать больше информации о состоянии объекта управления, обеспечить более точное соответствие цели управления и исключить человеческий фактор.

Диссертация посвящена решению задачи общего синтеза системы управления. Технически решение данной задачи — это создание блока управления, обеспечивающего выработку управления на основе информации о состоянии системы. Вырабатываемое управление должно обеспечивать достижение цели управления в соответствии с определенными критериями.

Задача общего синтеза системы управления как задача нахождения многомерной функции управления от компонент вектора состояния объекта управления была сформулирована В.Г. Болтянским в конце 60-х годов прошлого века [12; 51]. Найденная функция управления по имеющейся информации о состоянии объекта управления должна предоставлять значение в виде вектора управления, обеспечивающего перемещение объекта из начального состояния в терминальное по оптимальной по заданному критерию качества траектории.

Позднее в формулировке появилось уточнение, что достижение терминального состояния объекта управления должно обеспечиваться для любого начального состояния из всего пространства состояний.

До недавнего времени для систем управления, в которых требуется достижение цели в соответствии с заданным критерием качества, в основном использовалось программное управление без реализации принципа обратной связи. Реализация такого программного управления производилась на основе решения задачи оптимального управления. Найденная таким образом функция управления является функцией от времени и зависит только от состояния объекта управления в начальный момент времени, а не от текущего. В отличие от задачи оптимального управления, решение задачи общего синтеза системы управления реализует принцип обратной связи и позволяет найти функцию управления от координат пространства состояний объекта, при этом начальное состояние объекта уже не будет играть роли и может быть любым.

Данное отличие задачи оптимального управления от задачи общего синтеза системы управления говорит о большей значимости и одновременно большей сложности последней. Так как решение задачи синтеза позволяет найти оптимальное управление для любого начального состояния объекта, то в каком-то смысле его можно сравнить с решением бесконечного множества задач оптимального управления. В работе [24] задача общего синтеза системы управления отнесена к основной задаче современной теории управления и названа задачей тысячелетия.

Ввиду высокой сложности задачи общего синтеза системы управления, наиболее распространенным способом её решения является параметрический синтез. При таком подходе структура функциональной зависимости управляющих воздействий от текущего состояния системы управления задаётся разработчиком с точностью до некоторого числа неизвестных параметров. Далее производится настройка данных параметров как правило с помощью одного из оптимизационных алгоритмов. При выборе структуры функциональной зависимости разработчик может руководствоваться определенными знаниями и принимать решение на основе собственного опыта и предварительных исследований или даже на основе интуиции. Настройка параметров сделает предполагаемую функциональную зависимость более точной. Однако такой подход не является универсальным, а в случае отсутствия предварительной информации о форме функциональной связи будет вовсе неэффективным.

Гораздо больший интерес представляют методы структурно-параметрического решения задачи общего синтеза системы управления, направленные на одновременное получение и структуры функциональной зависимости, и оптимального значения параметров. Такие методы при поиске решения используют только математическую модель объекта управления и информацию об ограничениях на векторы состояния и управления.

Структурно-параметрические методы синтеза можно поделить на два класса. К первому классу относятся методы аналитического поиска решения задачи синтеза. Среди методов данного класса можно выделить метод аналитического конструирования оптимальных регуляторов, синтез на основе функций Ляпунова, метод бэкстеппинга и метод аналитического конструирования агрегированных регуляторов. Аналитические методы синтеза сильно ограничены размерностью и видом объекта управления и не являются универсальным способом решения поставленной задачи.

Ко второму классу относятся численные методы синтеза, в которых поиск структуры и параметров функции управления осуществляется на основе методов машинного поиска. В отличии от аналитических методов, методы данного класса не накладывают ограничений на размерность и вид объекта управления и не требуют предварительных аналитических преобразований.

Численный структурно-параметрический поиск стал возможен только в конце прошлого века. Именно в это время появляются первые методы машинного поиска. Пионером в этой области считается Джон Коза, который в 1992 году представил метод генетического программирования, относящийся к классу методов символьной регрессии. Метод задумывался для реализации автоматического составления компьютерных программ на языке LISP. В его основе лежит генетический алгоритм, а поиск осуществляется на нечисловом пространстве символов путем их перебора.

Метод генетического программирования и другие реализуемые с помощью вычислительных машин методы символьной регрессии позволили программно реализовывать структурно-параметрический поиск для целей решения задачи синтеза системы управления. В настоящее время известно более 10 методов символьной регрессии [23]. Метод генетического программирования является первым и, пожалуй, наиболее широко известным из всех. Отдельно следует упомянуть метод сетевого оператора, предложенный А.И. Дивеевым [96]. При его создании учитывалась специфика задачи синтеза системы управления.

Для осуществления поиска на нечисловом пространстве символов в каждом методе символьной регрессии предусмотрена обратимая операция кодирования. Кодирование позволяет представлять элементы символьной структуры в виде специфичного для данного метода кода. При решении задачи синтеза системы управления кодированию и дальнейшему поиску подвергается математическое выражение многомерной функции управления. Поиск оптимальной структуры осуществляется на основе генетического алгоритма.

Решение задачи синтеза системы управления на основе методов символьной регрессии рассмотрено в работах [95; 96]. В представленном в них подходе методы символьной регрессии используются для структурно-параметрического синтеза путем выбора оптимального решения на пространстве всех возможных решений. Основным недостатком такого подхода является отсутствие метрики для определения близости найденного решения к оптимальному.

Таким образом, известные численные методы синтеза системы управления позволяют осуществлять структурно-параметрический поиск функции управления, но не позволяют оценить близость найденного решения к оптимальному. Блок управления на основе такой функции управления позволит обеспечить достижение цели управления, но не оптимального значения критерия качества управления.

В настоящее время актуальной задачей является создание метода решения задачи синтеза системы управления, для которого будет определена оценка близости найденного решения к оптимальному. В составе блока управления такое решение будет обеспечивать достижение цели управления по траектории близкой к оптимальной, т.е. доставлять оптимальное значение критерия качества. Создание такого блока управления является необходимой частью этапа проектирования современных устройств в области самолетостроения и ракетостроения, создания роботизированных устройств, мобильных роботов, беспилотных летательных аппаратов, беспилотных автомобилей и др.

Решению данной задачи посвящены работы автора [156—180] и подытоживающая их настоящая диссертация.Автором разработан новый подход численного решения задачи синтеза системы управления и нахождения структуры многомерной функции управления на основе аппроксимации множества предварительно найденных оптимальных траекторий. Поиск решения в предлагаемом подходе осуществляется в два этапа. На первом этапе многократно решается задача оптимального управления для разных начальных условий. По-

лученные решения задачи оптимального управления для разных начальных условий позволяют на основе найденных оптимальных траекторий сформировать множество данных для аппроксимации. На втором этапе с помощью метода символьной регрессии осуществляется поиск математического выражения функции управления, аппроксимирующего данные из множества. В логике использования методов символьной регрессии и оценки качества аппроксимации в предложенном подходе используются принципы обучения с подкреплением. Использование данных, полученных на основе оптимальных траекторий, и функционала качества их аппроксимации позволяют для найденного решения задачи синтеза оценить его близость к оптимальному.

Целью диссертационной работы является разработка эффективного численного метода решения задачи синтеза системы управления с оценкой близости численного решения к оптимальному.

Для достижения поставленной цели были решены следующие **задачи**:

1. разработка численного метода решения задачи синтеза системы управления на основе аппроксимации оптимальных траекторий;
2. формирование множества данных для аппроксимации из оптимальных траекторий путём численного решения задачи оптимального управления для разных начальных условий;
3. сравнительное исследование численных методов решения задачи оптимального управления на основе прямого подхода и выбор наиболее эффективных из них;
4. разработка гибридного алгоритма оптимизации для решения задачи оптимального управления;
5. формализация вида функционала качества аппроксимации и разработка алгоритма поиска оптимального решения на основе методов символьной регрессии;
6. разработка комплекса программ, реализующих все этапы предложенного численного метода синтеза системы управления;
7. применение предложенного подхода и разработанного комплекса программ для решения прикладной задачи синтеза системы управления автомобилеподобным роботом.

Научная новизна диссертации состоит в следующем.

1. Впервые предложен двухэтапный численный метод решения задачи синтеза системы управления на основе аппроксимации множества оптимальных траекторий:
    а) предложено использовать информацию об оптимальных траекториях в качестве данных для аппроксимации;
    б) разработан алгоритм аппроксимации данных, полученных на основе оптимальных траекторий, с помощью методов символьной регрессии, позволяющий оценить близость найденного решения к оптимальному.
2. Выполнены сравнительные исследования и выявлены эффективные алгоритмы для решения задачи оптимального управления.
3. Разработан новый гибридный алгоритм для решения задачи оптимального управления.
4. Впервые с помощью разработанного метода и комплекса программ, его реализующего, численными экспериментами подтверждена его эффективность.

Теоретическая значимость. Разработан и апробирован численный метод решения задачи синтеза системы управления на основе аппроксимации оптимальных траекторий. Показано, что данный метод при поиске численного решения задачи синтеза позволяет определять близость найденного решения к оптимальному.

Практическая значимость. Разработка проблемно-ориентированного комплекса программ решения задачи синтеза системы управления была выполнена в рамках проекта 075-15-2020-799 «Методы построения и моделирования сложных систем на основе интеллектуальных и суперкомпьютерных технологий, направленные на преодоление больших вызовов» Минобрнауки России. Предложенный метод решения задачи синтеза системы управления используется в научно-исследовательской и опытно-конструкторской работе инжинирингового центра «Интеллектуальные роботизированные системы и технологии» Белгородского государственного технологического университета им. В.Г. Шухова (Белгород, Россия), а также внедрен в деятельность ООО «Экспериментальная мастерская НаукаСофт» (Москва, Россия) при проектировании и разработке программно-аппаратных комплексов, что подтверждается соответствующими актами, представленными в Приложении к диссертационной

работе. Разработанные алгоритмы применяются в научно-исследовательской работе роботоцентра Федерального исследовательского центра «Информатика и управление» Российской академии наук (Москва, Россия).

Соответствие диссертации паспорту научной специальности. В соответствии с формулой специальности 2.3.1 «Системный анализ, управление и обработка информации, статистика» (технические науки) в диссертации разработаны методы совершенствования управления сложными прикладными объектами, а именно — численный метод решения задачи синтеза системы управления. Разработанный метод ориентирован на повышение эффективности управления. Полученные результаты исследования соответствуют следующим пунктам областей исследований, перечисленных в паспорте научной специальности: пункт 4 — "Разработка методов и алгоритмов решения задач системного анализа, оптимизации, управления, принятия решений, обработки информации и искусственного интеллекта"; пункт 7 — "Методы и алгоритмы структурно-параметрического синтеза и идентификации сложных систем".

Методология и методы исследования. Методологическую основу диссертационного исследования составляют методы системного и сравнительного анализа в области теории оптимального управления, численных методов оптимизации, методов символьной регрессии и дифференциальных уравнений. При проведении вычислительного эксперимента применяется метод моделирования с использованием системного подхода.

Основные положения, выносимые на защиту:

1. Новый численный метод решения задачи общего синтеза системы управления на основе аппроксимации оптимальных траекторий методами символьной регрессии. Предложенный метод не накладывает ограничений на размерность и вид объекта управления, не требует предварительных аналитических преобразований. При поиске решения он позволяет оценить близость текущего решения к оптимальному.
2. Методы на основе эволюционных алгоритмов для решения задачи оптимального управления и получения оптимальных траекторий. Показано, что при прямом подходе решения задачи оптимального управления, для которого характерны увеличение размерности пространства поиска и многоэкстремальность целевой функции, эволюционные алгоритмы позволяют получить значительно более точные решения.

3. Новый гибридный алгоритм для решения задачи оптимального управления. Новый алгоритм сочетает свойства двух наиболее эффективных в области решения задачи оптимального управления алгоритмов.
4. Комплекс программ, реализующий предложенный метод для решения задачи общего синтеза системы управления.
5. Решение прикладной задачи синтеза системы управления автомобилеподобным роботом с помощью разработанного метода. На основании оценки полученного решения прикладной задачи подтверждена эффективность разработанного численного метода синтеза системы управления.

Достоверность Достоверность результатов следует из применения строгих математических методов и известных теоретических оценок погрешностей численных решений и на основании вычислительных экспериментов. Достоверность результатов подтверждается сравнением полученных решений с известными результатами других авторов.

Апробация работы. Основные результаты диссертационной работы докладывались на следующих конференциях: International Symposium “Intelligent Systems” (INTELS) — 2020, 2018, 2016, 2014; International Conference on Control, Decision and Information Technologies (CoDIT) — 2022, 2020, 2018; International Conference “Optimization and Applications” (OPTIMA) — 2021, 2020; European Control Conference (ECC) — 2020; Международная научно-практическая конференция, посвященная 110-летию со дня рождения академика Н.А. Пилюгина, “Фундаментально-прикладные проблемы безопасности, живучести, надёжности, устойчивости и эффективности систем” — 2019; International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD) — 2017; Всероссийская научная конференция с международным участием Моделирование коэволюции природы и общества: проблемы и опыт. К 100-летию со дня рождения академика Н.Н. Моисеева — 2017, Международная научно-практическая конференция “Инженерные системы” — 2015 и на научном семинаре “Всероссийский семинар с международным участием «Эволюционные вычисления»” (ФИЦ ИУ РАН) — 2019.

Личный вклад. Содержание диссертации и основные положения, выносимые на защиту, являются итогом самостоятельной работы автора. Подготовка к публикации полученных результатов проводилась совместно с соавторами. В совместных работах автор принимал непосредственное участие в постановке

задачи и выборе направления исследования, в программной реализации и обсуждении результатов исследований. При подготовке работ [162; 165; 173—178; 180] вклад диссертанта был определяющим. Все представленные в диссертации результаты получены лично автором.

Публикации. Основные результаты по теме диссертации изложены в 25 научных изданиях, среди них: публикаций в журналах, рекомендованных ВАК — 7, публикаций в журналах, индексируемых Web of Science и Scopus — 15, публикаций в трудах российских и зарубежных конференций — 3. Зарегистрированы 8 программ для ЭВМ.

Объем и структура работы. Диссертация состоит из введения, 5 глав, заключения, списка литературы и 1 приложения. Полный объём диссертации составляет 180 страниц, включая 12 рисунков и 4 таблицы. Список цитируемой литературы содержит 180 наименований.

Во Введении приведено обоснование актуальности темы диссертационной работы, сформулированы цели и задачи проводимых исследований, аргументирована научная новизна исследований, показана теоретическая и практическая значимость полученных результатов, представлены положения, выносимые на защиту, перечислены конференции и семинары, на которых докладывались основные результаты исследований и кратко изложено содержание разделов диссертации.

В Главе 1 приведены постановка и методы решения задачи синтеза системы управления: рассмотрена формальная постановка задачи общего синтеза системы управления; приведен обзор известных аналитических методов синтеза системы управления; рассмотрены их преимущества и недостатки; приведена постановка задачи численного синтеза системы управления на основе многокритериальной структурно-параметрической оптимизации; рассмотрен численный метод её решения и указаны его недостатки; сформулированы выводы по характеристикам рассмотренных методов.

В Главе 2 рассмотрены класс методов символьной регрессии и возможность их применения для задач синтеза системы управления: представлено описание класса методов символьной регрессии; приведен краткий обзор наиболее известных методов данного класса; рассмотрены их преимущества и недостатки; приведено описание принципа малых вариаций и метода сетевого оператора.

Глава 3 посвящена новому численному методу решения задачи синтеза системы управления на основе аппроксимации оптимальных траекторий: приведена постановка задачи численного метода синтеза системы управления на основе аппроксимации оптимальных траекторий методами символьной регрессии; представлены его описание и последовательность шагов при поиске решения; показаны преимущества предлагаемого метода.

В Главе 4 рассмотрены вопросы численного решения задачи оптимального управления: приведена постановка задачи оптимального управления; рассмотрены численные методы её решения; приведен обзор методов безусловной оптимизации для прямого подхода решения задачи оптимального управления; рассмотрены эволюционные алгоритмы; проанализированы результаты сравнительных экспериментов, демонстрирующих высокую эффективность применения эволюционных алгоритмов для решения задачи оптимального управления; рассмотрены методы гибридизации алгоритмов; предложен новый гибридный алгоритм; приведены его описание и основные преимущества.

В Главе 5 приведены описание и результаты вычислительного эксперимента: представлено описание математической модели автомобилеподобного робота; приведено описание задачи синтеза системы управления для рассматриваемого объекта; рассмотрены этап поиска множества оптимальных траекторий и этап синтеза системы управления на основе аппроксимации этих траекторий методом сетевого оператора; представлены результаты вычислительного эксперимента; сформулированы выводы об успешном решении поставленной задачи и эффективности предложенного метода.

В Заключении сформулированы основные результаты диссертации.

В Приложении А представлены акты о внедрении, подтверждающие применение результатов диссертационного исследования в научной и коммерческой деятельности.

Глава 1. Обзор методов решения задачи общего синтеза системы управления

Задача общего синтеза системы управления была так названа и сформулирована В.Г. Болтянским в конце 60-х годов прошлого века [12; 51]. Ей предшествовала постановка задачи оптимального управления Л.С. Понтрягиным [1]. Решение задачи общего синтеза системы управления состоит в нахождении многомерной функции управления от компонент вектора состояния объекта управления. Такая функция, получая в качестве аргумента вектор текущего состояния объекта управления, предоставляет значение в виде вектора управления, обеспечивающего перемещение объекта управления из начального состояния в терминальное по оптимальной по некоторому заданному критерию качества траектории. В работе [12] задача синтеза системы управления формулируется как нахождение функции управления для начальных условий из всего пространства состояний.

Принадлежность начальных условий всему пространству состояний или его некоторой области, наряду с необходимостью нахождения оптимального управления в виде многомерной функции от координат пространства состояний, составляет основную сложность и главное отличие задачи синтеза от задачи оптимального управления, в которой требуется найти управление в виде функции времени для определенных начальных условий. С практической точки зрения решение задачи синтеза более значимо, чем решение задачи оптимального управления в виде функции времени. С учетом того, что решение задачи оптимального управления позволяет найти оптимальное управление только для одного заданного начального состояния объекта, то решение задачи синтеза можно сопоставить с решением бесконечного множества задач оптимального управления.

Довольно часто в литературных источниках под синтезом системы управления рассматривают задачу поиска стабилизирующих управлений [35; 49]. Однако задача обеспечения устойчивости динамической системы является лишь частным случаем задачи синтеза системы управления. Решение задачи общего синтеза системы управления обеспечивает не только устойчивость объекта в области аттрактора, но и оптимальное по заданному критерию качества перемещение объекта из любого возможного начального состояния в данную область.

Задача общего синтеза системы управления по праву считается основной задачей в современной теории управления [24].

1.1 Постановка задачи общего синтеза системы управления

Рассмотрим постановку задачи синтеза оптимального управления. Пусть задана математическая модель объекта управления в виде

$$\begin{cases} \dfrac{dx_1}{dt} = f_1\left(x_1, \ldots, x_n, u_1, \ldots, u_m\right) \\ \ldots \\ \dfrac{dx_n}{dt} = f_n\left(x_1, \ldots, x_n, u_1, \ldots, u_m\right) \end{cases} \tag{1.1}$$

или в более краткой векторной форме

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, \mathbf{u}\right), \tag{1.2}$$

где $\mathbf{x} = [x_1 \ \ldots \ x_n]^T$ — вектор состояния объекта, $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{u} = [u_1 \ \ldots \ u_m]^T$ — вектор управления объектом, $\mathbf{u} \in \mathrm{U} \subseteq \mathbb{R}^m$, U — ограниченное замкнутое множество [51].

Задано множество начальных состояний

$$\mathrm{X}_0 \subseteq \mathbb{R}^n \tag{1.3}$$

и терминальные условия

$$\mathbf{x}\left(t_f\right) = \mathbf{x}^f, \tag{1.4}$$

где t_f — ограниченное время процесса управления, которое может быть задано или определяться в процессе решения задачи по условию достижения терминальных условий.

Функционал качества имеет вид

$$J = \int_0^{t_f} f_0\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right)\right) dt \to \min. \tag{1.5}$$

Требуется найти управление в форме многомерной функции от компонент вектора пространства состояний объекта

$$\mathbf{u} = \mathbf{h}(\mathbf{x}), \tag{1.6}$$

где $\mathbf{h}(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$, $\mathbf{h}(\mathbf{x}) \in \mathrm{U}$, $\forall \mathbf{x} \in \mathbb{R}^n$.

С учётом (1.6) представим математическую модель объекта управления (1.2) в виде

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, \mathbf{h}\left(\mathbf{x}\right)\right), \tag{1.7}$$

решением которой для любого начального состояния объекта из заданной области $\forall \mathbf{x}\left(0\right) = \mathbf{x}^0 \in \mathrm{X}_0$ будет векторная функция времени $\mathbf{x}\left(\mathbf{x}^0, t\right)$, которая обеспечивает перемещение объекта управления из начального состояния $\mathbf{x}\left(0\right)$ в терминальное положение $\mathbf{x}^f$ за конечное время $t_f < \infty$ и при этом доставляет минимум функционалу качества (1.5)

$$\min_{\mathbf{u} \in \tilde{\mathrm{U}}} \int_0^{t_f} f_0\left(\mathbf{x}\left(\mathbf{x}^0, t\right), \mathbf{u}\left(t\right)\right) dt, \tag{1.8}$$

где $\tilde{\mathrm{U}} = \{\tilde{\mathbf{u}}\left(\cdot\right)\}$ — множество всех допустимых управлений, удовлетворяющих ограничениям $\tilde{\mathbf{u}}\left(t\right) \in \mathrm{U}$, $0 \leqslant t \leqslant t_f$ и обеспечивающих достижение терминальных условий (1.4).

Таким образом, решение поставленной задачи синтеза оптимального управления требует поиска многомерной функции (1.6), удовлетворяющей условию оптимальности (1.8) для всех возможных начальных значений из множества (1.3).

Несмотря на важность задачи синтеза оптимального управления, точные методы ее решения отсутствуют по сей день. Аналитическое решение поставленной задачи возможно только для несложных объектов малой размерности, модели которых в форме (1.7) имеют решения. Для большинства прикладных объектов управления получить аналитическое решение задачи синтеза оптимального управления пока не удавалось.

1.2 Обзор методов решения задачи общего синтеза системы управления

Середина XX века считается моментом перерождения классической теории автоматического управления в так называемую современную теорию автоматического управления [75]. В её основе лежат работы Л.С. Понтрягина [51], В.Г. Болтянского [11—13], А.А. Красовского [42; 43; 74] Р. Беллмана [9;

10], Р. Калмана [30; 110] и других известных учёных. Существенное влияние на интенсивное развитие современной теории автоматического управления оказывала и продолжает оказывать высокая потребность в автоматизации управления технологическими и экономическими процессами. В 1960 году в Москве был проведен первый международный конгресс международной федерации по автоматическому управлению IFAC (International Federation of Automatic Control). Внимание ученых привлекло большое число актуальных задач в области оптимального управления, среди которых задача синтеза системы управления занимает одно из центральных мест.

За прошедшее время было разработано несколько методик решения поставленной задачи, которые показали свою эффективность в решении определенного круга задач. Достигнутый технологический прогресс, с одной стороны, позволил значительно увеличить эффективность существующих, как правило численных методов решения задачи синтеза. Но, с другой стороны, он предопределил значительное расширение сферы применения систем автоматического управления и усложнение объектов управления. Всё это делает поиск новых эффективных методов решения задачи синтеза системы управления актуальной задачей по сей день.

1.2.1 Синтез на основе метода динамического программирования

Метод динамического программирования был предложен в 50-х годах прошлого века коллективом авторов во главе с Р. Беллманом [66]. Метод основан на изучении множества оптимальных траекторий, переводящих объект управления из разных начальных состояний в одно терминальное состояние. Таким образом, данный метод позволяет находить решение задачи синтеза системы управления, а также применим к более широкому кругу технических и экономических задач.

Принцип оптимальности, сформулированный Р. Беллманом [9] и лежащий в основе метода динамического программирования, определяет зависимость системы управления в любой момент времени только от текущего состояния системы. Согласно данной формулировке, если для объекта управления известна оптимальная траектория перемещения из начального положения $\mathbf{x}^0$ в терми-

нальное положение $\mathbf{x}^f$, то для любой промежуточной точки $\mathbf{x}'$, расположенной на данной траектории и делящей оптимальную траекторию $\mathbf{x}^0\mathbf{x}^f$ на два произвольных участка, оба участка $\mathbf{x}^0\mathbf{x}'$ и $\mathbf{x}'\mathbf{x}^f$ также будут являться оптимальными траекториями.

При таком подходе определяется функционал

$$I = \int_{t_0}^{t_f} \mathbf{g}\left(\mathbf{x},\mathbf{u},t\right) dt \to \min, \tag{1.9}$$

и решается задача поиска такого вектора управления $\mathbf{u}$, при котором функционал (1.9) достигает своего минимального значения. Согласно принципу оптимальности Р. Беллмана оптимальное управление $\mathbf{u}$, являющееся решением исходной задачи, и вектор оптимального фазового состояния $\mathbf{x}$, полученный при управлении $\mathbf{u}$, для произвольного времени t, $\tilde{t} < t < t_f$ минимизирует функционал

$$\tilde{I} = \int_{\tilde{t}}^{t_f} \mathbf{g}\left(\mathbf{x},\mathbf{u},t\right) dt \to \min,$$

где $\tilde{t}$ — произвольно взятая точка на интервале $[0,t_f]$.

Следовательно, часть оптимального управления $\mathbf{u}$ и часть оптимального фазового состояния $\mathbf{x}$ для времени t, $\tilde{t} < t < t_f$ являются оптимальными по функционалу $\tilde{I}$ независимо от того, каким образом объект управления был переведен в начальное фазовое состояние $\mathbf{x}\left(\tilde{t}\right)$.

Для малого приращения времени dt значение функционала (1.9) примет вид

$$\hat{I} = \int_{t_0}^{t_0+dt} \mathbf{g}\left(\mathbf{x},\mathbf{u},t\right) dt \to \min, \tag{1.10}$$

тогда, согласно принципу оптимальности Беллмана, оставшаяся часть будет равна

$$\check{I} = \int_{t_0+dt}^{t_f} \mathbf{g}\left(\mathbf{x},\mathbf{u},t\right) dt \to \min.$$

Принимая во внимание, что для выражения (1.10) верно приближенное равенство

$$\hat{I} = \int_{t_0}^{t_0+dt} \mathbf{g}\left(\mathbf{x},\mathbf{u},t\right) dt \approx \mathbf{g}\left(\mathbf{x},\mathbf{u},t\right) dt,$$

функционал (1.9) можно перезаписать в следующем виде

$$I = \hat{I} + \check{I} = \mathbf{g}\left(\mathbf{x},\mathbf{u},t\right) dt + \int_{t_0+dt}^{t_f} \mathbf{g}\left(\mathbf{x},\mathbf{u},t\right) dt \to \min. \tag{1.11}$$

С учётом оптимальности по $\mathbf{u}$ необходимо найти такое решение, при котором функционал (1.11) принимает максимальное значение. Тогда, переходя к пределу при $dt \to 0$, из выражения (1.11) можно получить уравнение Беллмана в дифференциальной форме

$$-\frac{\partial I}{\partial t} = \min_{\mathbf{u}\in[\mathbf{u}^-,\mathbf{u}^+]} \left\{ \mathbf{g}\left(\mathbf{x},\mathbf{u},t\right) + \frac{\partial I}{\partial x}\mathbf{f}\left(\mathbf{x},\mathbf{u}\right) \right\}. \tag{1.12}$$

Использование данного метода позволяет получить множество значений векторов управлений для множества значений векторов состояний, и не позволяет получить структуру самой функции управления, описывающей данную зависимость. В большей части научных трудов метод динамического программирования применяется для поиска оптимального управления из одного начального состояния [83]. По сути это решение задачи оптимального управления, а не синтеза. Применение метода динамического программирования для решения именно задачи синтеза системы управления требует дополнительных нетривиальных ухищрений. Так в работе [28] осуществляется поиск приближенного решения задачи синтеза оптимального управления с помощью метода динамического программирования с последующим применением метода усреднения [55]. В работе [57] предлагается использовать итерационный подход к определению на основе метода динамического программирования оптимальных управлений в малой окрестности точки начального состояния объекта управления. В результате многократного применения метода динамического программирования получается матрица оптимальных управлений, размерность и точность которой прямо пропорциональны и зависят от числа точек разбиения множества начальных состояний и применения методов аппроксимации найденных решений.

1.2.2 Синтез на основе принципа максимума Понтрягина

В.Г. Болтянским в своих работах [12; 14], а также в коллективной работе [51] для решения задачи синтеза оптимального управления предлагается использовать подход на основе принципа максимума Понтрягина.

Для формулировки принципа максимума Понтрягина в дополнение к основной системе дифференциальных уравнений (1.1) вводится ещё одна система

уравнений относительно вектора вспомогательных переменных $\boldsymbol{\psi} = [\psi_1 \dots \psi_n]^T$

$$\begin{cases} \dfrac{d\psi_1}{dt} = -\sum\limits_{j=1}^{n} \dfrac{\partial f_j(\mathbf{x},\mathbf{u})}{\partial x_1}\psi_j \\ \dots \\ \dfrac{d\psi_n}{dt} = -\sum\limits_{j=1}^{n} \dfrac{\partial f_j(\mathbf{x},\mathbf{u})}{\partial x_n}\psi_j \end{cases}. \tag{1.13}$$

Системы (1.1) и (1.13) объединяют одной записью, вводя функцию H, которая в случае задачи оптимальности по быстродействию имеет вид

$$H(\boldsymbol{\psi},\mathbf{x},\mathbf{u}) = -f_0(\mathbf{x},\mathbf{u}) + \sum_{j=1}^{n} \psi_j f_j(\mathbf{x},\mathbf{u}). \tag{1.14}$$

На основе уравнения (1.14) можно записать уравнения (1.1) и (1.13) в виде гамильтоновой системы

$$\frac{dx_i}{dt} = \frac{\partial H}{\partial \psi_i}, \quad i = \overline{1,n}, \tag{1.15}$$

$$\frac{d\psi_i}{dt} = -\frac{\partial H}{\partial x_i}, \quad i = \overline{1,n}. \tag{1.16}$$

При фиксированных значениях векторов $\boldsymbol{\psi}$ и $\mathbf{x}$ функция H становится функцией от вектора управления $\mathbf{u}$. Таким образом, решением краевой задачи оптимального управления по быстродействию будет такой ненулевой вектор $\boldsymbol{\psi} = [\psi_1 \dots \psi_n]^T$, при котором функция H переменного вектора $\mathbf{u}$ достигает своего максимального значения

$$\max_{\mathbf{u}\in[\mathbf{u}^-,\mathbf{u}^+]} H(\boldsymbol{\psi},\mathbf{x},\mathbf{u}). \tag{1.17}$$

Здесь следует отметить, что описанный подход применяется в основном для решения задачи оптимального управления. При использовании данного подхода для решения задачи синтеза системы управления, на начальном этапе осуществляется поиск всех фазовых траекторий, удовлетворяющих принципу максимума Понтрягина. Движение по полученным траекториям ввиду принципа максимума будет оптимальным. Далее осуществляется поиск кривых переключения на фазовой плоскости, определяющих синтез оптимальных управлений объектом.

Применение принципа максимума Понтрягина для поиска оптимального управления в виде функций от переменных состояния требует большого числа

аналитических преобразований, при этом полученное решение не будет работать для режимов особого управления, например, в случаях наличия фазовых ограничений, что довольно часто встречается в прикладных задачах.

В качестве примера рассмотрим задачу поиска закона оптимального управления в плане быстродействия для простейшего управляемого объекта — материальной точки, двигающейся по горизонтальной прямой без учета силы трения и упругой силы [12]. Уравнения движения данного объекта имеют следующий вид

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = u \end{cases}. \tag{1.18}$$

Для рассматриваемого объекта решается задача о наискорейшем перемещении из произвольной начальной точки $\mathbf{x}^0 = \begin{bmatrix} x_1^0 & x_2^0 \end{bmatrix}^T$ в начало координат $\mathbf{x}^f = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$. Ограничения, наложенные на управление, имеют следующий вид

$$|u| \leqslant 1.$$

Функционал качества для рассматриваемой задачи быстродействия

$$J = \int_0^{t_f} f_0\left(\mathbf{x},\mathbf{u}\right) dt \to \min.$$

В соответствии с (1.14) функция Гамильтона для объекта (1.18) имеет вид

$$H = -1 + \psi_1 x_2 + \psi_2 u. \tag{1.19}$$

Используя соотношение (1.16), получаем следующую систему уравнений для вспомогательных переменных

$$\begin{cases} \dot{\psi}_1 = 0 \\ \dot{\psi}_2 = -\psi_1 \end{cases},$$

откуда находим значения ψ_1 и ψ_2

$$\begin{cases} \psi_1 = d_1 \\ \psi_2 = -d_1 t + d_2 \end{cases},$$

где d_1 и d_2 — постоянные интегрирования.

В соответствии с принципом максимума Понтрягина (1.17) решением задачи оптимального управления по быстродействию будет такой ненулевой вектор $\psi = \begin{bmatrix} \psi_1 & \psi_2 \end{bmatrix}^T$, при котором функция H (1.19) достигает своего максимального значения.

Отсюда получаем следующие условия для значений управления

$$\begin{cases} u(t) = 1, \text{ если } \psi(t) = -d_1 t + d_2 > 0 \\ u(t) = -1, \text{ если } \psi(t) = -d_1 t + d_2 < 0 \end{cases}.$$

В виду того, что линейная функция $-d_1 t + d_2$ меняет знак не более одного раза на интервале времени $0 \leqslant t \leqslant t_f$, то любое оптимальное управление на данном отрезке времени имеет не более двух интервалов постоянства.

Таким образом, можно построить семейство фазовых траекторий для отрезка времени, на котором $u(t) = 1$, и для отрезка времени, на котором $u(t) = -1$.

Для случая, когда $u(t) = 1$ из уравнения объекта (1.18), имеем $\frac{dx_1}{dx_2} = x_2$, откуда

$$x_1 = \frac{1}{2} x_2^2 + c. \tag{1.20}$$

Для случая, когда $u(t) = -1$ из уравнения объекта (1.18), имеем $\frac{dx_1}{dx_2} = -x_2$, откуда

$$x_1 = -\frac{1}{2} x_2^2 + c'. \tag{1.21}$$

Построив на основе (1.20) и (1.21) семейства фазовых траекторий, можно для любого начального состояния $\mathbf{x}^0$ определить траекторию движения фазовой точки и соответствующее этому движению оптимальное управление

$$\tilde{u} = 1 - 2\vartheta\left(h\left(\mathbf{x}\right)\right),$$

где $h\left(\mathbf{x}\right)$ — разрывная функция вида

$$h\left(\mathbf{x}\right) = \begin{cases} x_1 + \frac{x_2^2}{2}, \text{ если } x_2 > 0 \\ x_1 - \frac{x_2^2}{2} \text{ — иначе} \end{cases},$$

$\vartheta\left(h\left(\mathbf{x}\right)\right)$ — функция Хэвисайда

$$\vartheta\left(h\left(\mathbf{x}\right)\right) = \begin{cases} 1, \text{ если } h\left(\mathbf{x}\right) > 0 \\ 0 \text{ — иначе} \end{cases}. \tag{1.22}$$

В научной литературе рассмотренный подход в основном используется для аналитического исследования несложных систем невысокого порядка. Так в работе [65] рассматривается классический пример поиска управляющей функции, переводящей нелинейный маятник в устойчивое положение за минимальное время. В работе представлено аналитическое решение, дополненное численными расчетами.

В работе [69] для синтеза системы управления предлагается усовершенствованный метод на основе принципа максимума. Авторами приводится аналитическое решение задачи вывода спутника на заданную орбиту. Недостатком предлагаемого подхода является необходимость выполнения условия трансверсальности. Определение произвольных постоянных, при которых условие трансверсальности будет выполняться, возможно для весьма ограниченного круга задач.

В работе [45] авторами проводится сравнение результатов решения одной и той же задачи синтеза системы управления методом на основе принципа максимума Понтрягина и методом аналитического конструирования агрегированных регуляторов (АКАР). Авторами отмечается сложность применения метода на основе принципа максимума, связанная с необходимостью решения систем дифференциальных уравнений, что не всегда возможно для нелинейных случаев.

Сочетание метода динамического программирования и принципа максимума Понтрягина для задачи синтеза системы управления рассмотрено в работе [153]. Авторами рассматриваются аналитические методы построения поверхностей переключения управлений, а также методы редукции задачи синтеза системы управления к исследованию задачи Коши с указанием на существенные ограничения данного подхода.

1.2.3 Метод аналитического конструирования оптимальных регуляторов

Понятие «аналитическое конструирование» и последовавший за этим метод аналитического конструирования оптимальных регуляторов (АКОР) были предложены профессором А.М. Летовым в 1960 году [74]. В своих работах [46] А.М. Летов определил АКОР как чисто аналитических метод получения закона

управления. В настоящее время метод АКОР эффективно применяется к задачам с линейными объектами и квадратичными критериями качества. Данное направление получило название метода АКОР по Летову-Калману [3]. Более современная модификация метода с использованием критерия качества обобщенной работы была предложена А.А. Красовским и получила название метода АКОР по Красовскому [42].

В соответствии с постановкой задачи АКОР, требуется найти такой закон управления (1.6), который позволит перевести объект управления (1.2) из любого начального состояния (1.3) в терминальное состояние (1.4), соответствующее началу координат фазового пространства, обеспечивая устойчивость системы и минимум функционалу качества (1.5).

В основе решения задачи синтеза системы управления методом АКОР лежит метод динамического программирования. При этом уравнение Беллмана предлагается преобразовать в более простые для решения алгебраические или дифференциальные уравнения Риккати.

Представим математическую модель (1.2) в векторно-матричной форме для линейного объекта управления

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \tag{1.23}$$

где $\mathbf{x} = [x_1 \ \dots \ x_n]^T$, $\mathbf{x} \in \mathbb{R}^n$ — вектор состояния объекта, $\mathbf{u} = [u_1 \ \dots \ u_m]^T$, $\mathbf{u} \in \mathbb{R}^m$ — ограниченное управление объектом. Для данного объекта управления будем рассматривать классическую задачу поиска функции управления, переводящей объект из начального состояния (1.3) в терминальное состояние (1.4). Функционал качества (1.5) примет вид

$$J = \int_0^{t_f} \left(\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u}\right) dt \to \min. \tag{1.24}$$

Сразу следует отметить, что функционал вида (1.24) не учитывает возможные фазовые ограничения, накладываемые на вектор состояния объекта, что является довольно актуальным при решении прикладных задач.

Далее записываем основное функциональное уравнение метода динамического программирования для поставленной задачи (1.23) и (1.24)

$$\min_{\mathbf{u}} \left\{ \frac{dS(\mathbf{x})}{dt} + \mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u} \right\} = 0. \tag{1.25}$$

С учётом того, что

$$\frac{dS(\mathbf{x})}{dt} = \sum_{i=1}^{n} \frac{\partial S(\mathbf{x})}{\partial x_i} \dot{\mathbf{x}}_i,$$

выражение (1.25) можно представить в виде

$$\min_{\mathbf{u}} \left\{ \left(\frac{\partial S(\mathbf{x})}{\partial \mathbf{x}} \right)^T (\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}) + \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{u}^T \mathbf{R} \mathbf{u} \right\} = 0. \tag{1.26}$$

Дифференцирование выражения в фигурных скобках по управлению позволяет выразить функцию управления через функцию Беллмана

$$\mathbf{u} = -\frac{1}{2} \mathbf{R}^{-1} \mathbf{B}^T \frac{\partial S(\mathbf{x})}{\partial \mathbf{x}}. \tag{1.27}$$

Подстановка выражения для управления из (1.27) в уравнение (1.26) позволяет получить уравнение Гамильтона-Якоби-Беллмана

$$\left(\frac{\partial S(\mathbf{x})}{\partial \mathbf{x}} \right)^T \mathbf{A} - \frac{1}{4} \left(\frac{\partial S(\mathbf{x})}{\partial \mathbf{x}} \right)^T \mathbf{B} \mathbf{R}^{-1} \mathbf{B}^T \frac{\partial S(\mathbf{x})}{\partial \mathbf{x}} + \mathbf{x}^T \mathbf{Q} \mathbf{x} = 0. \tag{1.28}$$

Уравнение Гамильтона-Якоби-Беллмана (1.28) является уравнением в частных производных и позволяет определить функцию Беллмана $S(\mathbf{x})$. Таким образом, получив аналитическое решение уравнения (1.28) с краевым условием $S\left(\mathbf{x}(t_f)\right) = 0$ и подставив его в выражение (1.27), можно получить решение задачи синтеза системы управления для заданного объекта.

Однако общее решение уравнения в частных производных (1.28) известно только для линейных объектов [49]. В таком случае решение задачи синтеза системы управления сводится к решению алгебраических уравнений Риккати.

Для линейного объекта управления функция Беллмана имеет решение в квадратичной форме

$$S(\mathbf{x}) = \mathbf{x}^T \mathbf{P} \mathbf{x}, \tag{1.29}$$

где $\mathbf{P}$ — симметричная, положительно определенная матрица размера $n \times n$. Тогда, подставляя выражение (1.29) в уравнение Гамильтона-Якоби-Беллмана (1.28), получаем

$$\mathbf{x}^T \left(\mathbf{P}\mathbf{A} + \mathbf{A}^T \mathbf{P} - \mathbf{P}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T \mathbf{P} + \mathbf{Q} \right) \mathbf{x} = 0. \tag{1.30}$$

В выражении (1.30) $\mathbf{x} \neq 0$, следовательно имеем алгебраическое матричное уравнение Рикатти

$$\mathbf{PA} + \mathbf{A}^T\mathbf{P} - \mathbf{PBR}^{-1}\mathbf{B}^T\mathbf{P} + \mathbf{Q} = 0. \tag{1.31}$$

Подставив выражение (1.29) в (1.27) и найдя матрицу $\mathbf{P}$ как положительно определенное решение уравнения Рикатти (1.31), получаем систему управления для линейного объекта (1.23)

$$\mathbf{u} = -\frac{1}{2}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{Px}.$$

В качестве примера рассмотрим модель короткопериодического движения самолета, которая описывается следующей системой дифференциальных уравнений [15]

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = a_1x_1 + a_2x_2 + a_3u \end{cases}, \tag{1.32}$$

где x_1 — угол атаки, x_2 — скорость изменения угла атаки, u — отклонение руля высоты, a_1, a_2 и a_3 — коэффициенты системы, соответствующие определенному режиму полета самолета. Для заданной системы (1.32) необходимо найти функцию управления рулем высоты $\tilde{u}(\mathbf{x})$, обеспечивающую стабилизирующее движение самолета по углу атаки $x_1^* = x_2^* = 0$.

В соответствии с методом АКОР по Красовскому используется функционал качества общей работы следующего вида

$$J = \int_0^{t_f} \left(q_1x_1^2 + q_2x_2^2 + \frac{1}{2}ru^2 + \frac{1}{2}r(u^*)^2 \right) dt.$$

Далее записывается основное функциональное уравнение метода динамического программирования

$$\min_u \left\{ \frac{\partial S}{\partial x_1}x_2 + \frac{\partial S}{\partial x_2}\dot{x}_1 + q_1x_1^2 + q_2x_2^2 + \frac{1}{2}ru^2 + \frac{1}{2}r(u^*)^2 \right\} = 0. \tag{1.33}$$

Дифференцирование выражения (1.33) по управлению позволяет выразить управление через функцию Беллмана

$$\tilde{u} = -\frac{a_3}{r}\frac{\partial S}{\partial x_2}. \tag{1.34}$$

Подставив управление (1.34) в уравнение (1.33) получаем уравнение Гамильтона-Якоби-Беллмана

$$\frac{\partial S}{\partial x_1}x_2 + \frac{\partial S}{\partial x_2}\left(a_1x_1 + a_2x_2\right) + q_1x_1^2 + q_2x_2^2 = 0. \tag{1.35}$$

Функцию Беллмана можно представить в виде положительно определенной квадратичной формы

$$S(x_1,x_2) = A_{11}x_1^2 + A_{12}x_1x_2 + A_{22}x_2^2, \tag{1.36}$$

где коэффициенты A_{11}, A_{12} и A_{22} определяются через параметры a_1, a_2, a_3, q_1, q_2 и r путем подстановки $\partial S/\partial x_i$, $i = 1{,}2$ в уравнение Гамильтона-Якоби-Беллмана (1.35)

$$A_{12} = -\frac{q_1}{a_1}, \quad A_{22} = \frac{q_1 - a_1q_2}{2a_1a_2}. \tag{1.37}$$

Таким образом, получив выражение функции Беллмана (1.36) через коэффициенты (1.37) и подставив ее в (1.34), получаем функцию управления для поставленной задачи

$$\tilde{u}(\mathbf{x}) = \frac{a_3}{r}\left(\frac{q_1}{a_1}x_1 + \frac{a_1q_2 - q_1}{a_1a_2}x_2\right).$$

При практическом применении метода АКОР для синтеза системы управления нелинейными объектами возникает ряд существенных математических и вычислительных сложностей, связанных с необходимостью решения уравнения в частных производных Гамильтона-Якоби-Беллмана (1.28) относительно производящей функции [73; 82]. Данные сложности имеют свойство стремительно нарастать с увеличением порядка объекта управления, что делает метод АКОР непригодным для использования при поиске функции управления современными объектами высокого порядка. В [2; 16] отмечается, что решение уравнения Гамильтона-Якоби-Беллмана найдено только для отдельных объектов второго и третьего порядков. В работе [48] утверждается, что функция Беллмана даже для линейных объектов управления при определенных значениях своих аргументов оказывается разрывной функцией, что делает её недифференцируемой.

Предложенный А.А. Красовским модифицированный метод АКОР с использованием так называемого функционала обобщенной работы [42], позволил преодолеть некоторые трудности при решении нелинейной задачи АКОР, но не

избавиться от них полностью. Использование функционала обобщенной работы позволило заменить сложное уравнение Гамильтона-Якоби-Беллмана (1.28) более простым для решения линейным уравнением в частных производных Ляпунова [4; 44]. Вычислительные трудности, связанные с решением уравнения Ляпунова, для задач с высокой размерностью объекта управления существенно меньше трудностей, с которыми приходится сталкиваться при решении уравнения Рикатти.

Таким образом, метод АКОР предлагает удобные инструменты поиска аналитического решения задачи синтеза системы управления только для линейных систем, тогда как поиск решения нелинейной задачи АКОР, спустя более полувека с момента её формулировки, является одной из основных проблем теории автоматического управления [73].

1.2.4 Синтез на основе функций Ляпунова

В конце XIX века А.М. Ляпуновым был предложен общий подход к исследованию устойчивости динамических систем. Позднее данный подход получил название метода функций Ляпунова [7]. На основе функции Ляпунова для конкретной системы управления можно в первую очередь получить оценку её устойчивости. Также функции Ляпунова позволяют осуществить синтез системы управления заданным объектом [29; 44].

В соответствии с постановкой задачи синтеза системы управления (1.6) для объекта (1.2) методом на основе функций Ляпунова, требуется определить такой вид обратной связи, чтобы положение равновесия системы управления (1.2) в области целевой точки (1.4) было асимптотически устойчивым в целом. Для обеспечения асимптотической устойчивости системы необходимо выбрать такое управление (1.6), чтобы производная по времени функции Ляпунова являлась отрицательно определенной функцией на траекториях рассматриваемой системы.

В общей постановке задачи для объекта управления, заданного в виде (1.2), определен функционал качества

$$J = \int_0^{t_f} \boldsymbol{\omega}\left(\mathbf{x},\mathbf{u},t\right) dt \to \min, \tag{1.38}$$

где $\boldsymbol{\omega}(\mathbf{x},\mathbf{u},t)$ — некоторая положительно определенная функция.

Для поиска функции управления (1.6), доставляющей минимум функционалу качества (1.38), введем положительно определенную функцию Ляпунова $V(\mathbf{x},t)$, для которой существуют непрерывные частные производные первого порядка на всей области её определения.

Производная функции Ляпунова по времени для уравнения (1.2) имеет вид

$$\dot{V} = \frac{\partial V}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x},\mathbf{u}) + \frac{\partial V}{\partial t}. \tag{1.39}$$

Требуется найти такое управление $\mathbf{u} = \mathbf{u}(\mathbf{x},t)$, при котором выполняется неравенство

$$\dot{V} \leqslant -\boldsymbol{\omega}(\mathbf{x},\mathbf{u},t). \tag{1.40}$$

С учетом (1.40) уравнение (1.39) примет вид

$$\boldsymbol{\mu}(\mathbf{x},\mathbf{u},t) = \frac{\partial V}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x},\mathbf{u}) + \frac{\partial V}{\partial t} + \boldsymbol{\omega}(\mathbf{x},\mathbf{u},t) \leqslant 0. \tag{1.41}$$

Разрешая неравенство (1.41) относительно управления $\mathbf{u}$ при фиксированном значении функции Ляпунова $V(\mathbf{x},t)$, получаем решение поставленной задачи.

Рассмотрим задачу синтеза системы управления на основе функций Ляпунова для линейного объекта управления. Пусть математическая модель объекта управления имеет вид (1.23). Тогда в качестве функции Ляпунова может быть выбрана квадратичная функция вида

$$V(\mathbf{x},t) = \mathbf{x}^T\mathbf{P}\mathbf{x}, \tag{1.42}$$

где $\mathbf{P}$ — симметричная, положительно определенная матрица размера $n \times n$. Функцию $\boldsymbol{\omega}(\mathbf{x},\mathbf{u},t)$ также можно представить в квадратичной форме

$$\boldsymbol{\omega}(\mathbf{x},t) = \mathbf{x}^T\mathbf{Q}\mathbf{x}, \tag{1.43}$$

где $\mathbf{Q}$ — симметричная, положительно определенная матрица размера $n \times n$. Неравенство (1.41) для системы (1.23) примет вид

$$\boldsymbol{\mu}(\mathbf{x},\mathbf{u},t) = \frac{\partial V}{\partial \mathbf{x}}\mathbf{A}\mathbf{x} + \frac{\partial V}{\partial \mathbf{x}}\mathbf{B}\mathbf{u} + \frac{\partial V}{\partial t} + \boldsymbol{\omega}(\mathbf{x},\mathbf{u},t) \leqslant 0. \tag{1.44}$$

В соответствии с достаточным условием стабилизируемости системы, для положительно определенной функции Ляпунова $V(\mathbf{x},t)$ и положительно определенной функции $\boldsymbol{\omega}(\mathbf{x},t)$ и при действительных значениях вектора состояния

системы $\mathbf{x}$, для которых выполняется уравнение

$$\sigma(\mathbf{x},t) = \frac{\partial V}{\partial \mathbf{x}}\mathbf{B} = 0, \tag{1.45}$$

система стабилизируема, если выполняется неравенство

$$\mu(\mathbf{x},0,t) = \frac{\partial V}{\partial \mathbf{x}}\mathbf{A}\mathbf{x} + \frac{\partial V}{\partial t} + \omega(\mathbf{x},0,t) \leqslant 0. \tag{1.46}$$

С учетом (1.42) и (1.43) неравенство (1.46) и уравнение (1.45) примут вид соответственно

$$\mu(\mathbf{x},0,t) = \mathbf{x}^T\left(\mathbf{P}\mathbf{A} + \mathbf{A}^T\mathbf{P} + \mathbf{Q}\right)\mathbf{x} \leqslant 0, \tag{1.47}$$

$$\sigma(\mathbf{x},t) = 2\mathbf{x}^T\mathbf{P}\mathbf{B} = 0. \tag{1.48}$$

Тогда для значений вектора состояния системы $\mathbf{x}$, для которых уравнение (1.48) не выполняется, функция управления выделяется ограничением

$$\mathbf{u}(\mathbf{x},t)\,\mathrm{sgn}\left(\sigma(\mathbf{x},t)\right) \leqslant -\frac{\mu(\mathbf{x},0,t)}{|\sigma(\mathbf{x},t)|}.$$

При значениях вектора состояния системы $\mathbf{x}$, для которых уравнение (1.48) истинно, значения управления $\mathbf{u}(\mathbf{x},t)$ могут быть произвольными в рамках наложенных на вектор управления ограничений.

Рассмотрим пример на основе уравнения маятника [80]

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = a\sin(x_1) + u \end{cases}, \tag{1.49}$$

где a — заданный параметр системы. Для данной системы необходимо найти функцию управления, стабилизирующую маятник в точке равновесия $\mathbf{x} = 0$.

В качестве функции Ляпунова возьмем квадратичную функцию следующего вида

$$V(\mathbf{x},t) = \frac{1}{2}x_1^2 + \frac{1}{2}x_2^2.$$

Пусть функция $\omega(\mathbf{x},\mathbf{u},t)$ также представлена в квадратичной форме

$$\omega(\mathbf{x},t) = x_2^2.$$

Тогда, в силу (1.39) и (1.40), имеем следующее уравнение

$$x_1x_2 + x_2\left(a\sin(x_1) + u\right) \leqslant -x_2^2. \tag{1.50}$$

Из полученного уравнения (1.50) выразим управление $\mathbf{u} = \mathbf{u}(\mathbf{x},t)$

$$u = -x_1 - a\sin(x_1) - x_2. \tag{1.51}$$

На сегодняшний день не существует универсального метода построения функций Ляпунова [35]. Для некоторых нелинейных систем определенного вида разработаны различные методы нахождения функций Ляпунова, среди которых метод разделения переменных Е.А. Барбашина [6], метод Лурье [50] и др.

Следует отметить связь метода АКОР с использованием критерия общей работы при решении нелинейных задач и метода функций Ляпунова, о которой говорилось выше. Отсюда следует применимость метода АКОР с критерием общей работы только к устойчивым объектам управления. В работе [60] также отмечается наличие взаимосвязи между подходом к решению задачи синтеза системы управления на основе функций Ляпунова и методом на основе принципа максимума.

Метод функций Ляпунова применим только к устойчивым объектам управления. Также как и при использовании описанных выше методов поиска функции управления, трудности при поиске решения методом функций Ляпунова имеют свойство стремительно расти с увеличением порядка системы. Данные факты, наряду с отсутствием общего подхода к нахождению функций Ляпунова для нелинейных систем, делают этот метод пригодным только для линейных систем управления невысокого порядка.

1.2.5 Метод бэкстеппинга

Метод бэкстеппинга (Backstepping), также известный как метод адаптивного обхода интегратора, был предложен в 1991 году коллективом авторов во главе с П. Кокотовичем [111; 114]. В настоящий момент данный метод является достаточно популярным и широко освещен как в иностранной, так и отечественной научной литературе [79]. Бэкстеппинг применим к классу задач нелинейного синтеза и нашел применение к широкому кругу задач управления в аэрокосмической и технической отраслях [70; 78; 155].

В основе метода бэкстеппинга лежит рекурсивная процедура делающая каждый интегратор объекта управления устойчивым по Ляпунову путём до-

бавления обратной связи. Отсюда происходит другое название метода — метод адаптивного обхода интегратора. Алгоритм бэкстеппинга совмещает задачу нахождения функций Ляпунова и поиск соответствующей функции управления объектом. Согласно алгоритму исходная задача синтеза системы управления для всей системы разбивается на последовательность подзадач для подсистем меньшего порядка. Для каждого дифференциального уравнения, описывающего исходный объект управления, осуществляется добавление обратной связи, делающей интегратор устойчивым по Ляпунову. Иными словами, на каждой итерации для рассматриваемого уравнения исходной системы управления (1.2) перед разработчиком встаёт задача определения функции Ляпунова и коэффициентов. По теореме Ляпунова для обеспечения устойчивости интегратора производная соответствующей функции Ляпунова должна быть отрицательно полуопределенной (1.40). После последовательного обхода всех интеграторов возможно получить функцию управления для всей системы.

Пусть математическая модель объекта управления задана в общем виде

$$\begin{cases} \dot{x}_1 = f_1(x_1) + g_1(x_1)\, x_2 \\ \dot{x}_2 = f_2(x_1,x_2) + g_2(x_1,x_2)\, u \end{cases}, \tag{1.52}$$

где f_i и g_i, $i = 1,2$ — известные функции.

В системе (1.52), при условии выполнения неравенства $g_2(x_1,x_2) \neq 0$ в рассматриваемой области допустимых значений вектора состояния $\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T$, возможно преобразование к виду, соответствующему простому интегратору, путем замены входа системы на выражение

$$u = \frac{1}{g_2(x_1,x_2)} \left(u' - f_2(x_1,x_2)\right). \tag{1.53}$$

Тогда систему (1.52) можно преобразовать к виду

$$\begin{cases} \dot{x}_1 = f_1(x_1) + g_1(x_1)\, x_2 \\ \dot{x}_2 = u' \end{cases}. \tag{1.54}$$

Целью рассматриваемой задачи является построение системы управления с обратной связью по координатам вектора состояния, которая обеспечивала бы стабилизацию системы в начале координат. При необходимости стабилизации системы в области другой точки координат состояния, путем преобразования координат можно перейти к стабилизации в начале координат.

В соответствии с методом бэкстеппинга, рассмотрим подсистему системы управления (1.54), состоящую из первого уравнения. Пусть данная подсистема может быть стабилизирована обратной связью вида

$$x_2 = s\left(x_1\right),$$

где s — стабилизационная функция, причем

$$s\left(0\right) = 0. \tag{1.55}$$

Из (1.55) следует, что начало координат подсистемы $\dot{x}_1 = f_1\left(x_1\right) + g_1\left(x_1\right) s\left(x_1\right)$ является асимптотически устойчивым.

Пусть известны положительно определенные функция Ляпунова $V\left(x_1\right)$ и функция $\boldsymbol{\omega}\left(x_1\right)$, для которых в соответствии с теоремой Ляпунова выполняется неравенство

$$\frac{\partial V}{\partial x_1}\left(f_1\left(x_1\right) + g_1\left(x_1\right) s\left(x_1\right)\right) \leqslant -\boldsymbol{\omega}\left(x_1\right).$$

Представим первое уравнение системы (1.54) следующим образом

$$\dot{x}_1 = f_1\left(x_1\right) + g_1\left(x_1\right) x_2 + g_1\left(x_1\right) s\left(x_1\right) - g_1\left(x_1\right) s\left(x_1\right).$$

Тогда модифицированная система (1.54) примет вид

$$\begin{cases} \dot{x}_1 = f_1\left(x_1\right) + g_1\left(x_1\right) s\left(x_1\right) + g_1\left(x_1\right)\left(x_2 - s\left(x_1\right)\right) \\ \dot{x}_2 = u' \end{cases}. \tag{1.56}$$

Выражение $x_2 - s\left(x_1\right)$ соответствует отклонению значения стабилизационной функции от желаемого. Данное отклонение, называемое также ошибкой состояния, можно представить как

$$z = x_2 - s\left(x_1\right). \tag{1.57}$$

Подставив (1.57) в (1.56), получим

$$\begin{cases} \dot{x}_1 = f_1\left(x_1\right) + g_1\left(x_1\right) s\left(x_1\right) + g_1\left(x_1\right) z \\ \dot{z} = u' - \dot{s}\left(x_1\right) \end{cases}, \tag{1.58}$$

где

$$\dot{s}\left(x_1\right) = \frac{\partial s}{\partial x_1}\left(f_1\left(x_1\right) + g_1\left(x_1\right) x_2\right), \tag{1.59}$$

Возьмем

$$\upsilon = u' - \dot{s}\left(x_1\right), \tag{1.60}$$

тогда система (1.58) примет вид

$$\begin{cases} \dot{x}_1 = f_1\left(x_1\right) + g_1\left(x_1\right) s\left(x_1\right) + g_1\left(x_1\right) z \\ \dot{z} = \upsilon \end{cases}. \tag{1.61}$$

В полученной системе (1.61) первая подсистема асимптотически устойчива в начале координат и требуется найти такое управление υ, которое стабилизирует всю систему.

Для этого в качестве функции Ляпунова для системы (1.61) возьмем

$$V\left(x_1,x_2\right) = V\left(x_1\right) + \frac{1}{2}z^2.$$

В качестве функции $\omega\left(x_1,z\right)$ возьмем

$$\omega\left(x_1,z\right) = \omega\left(x_1\right) + kz^2.$$

На основе теоремы Ляпунова имеем неравенство

$$\frac{\partial V}{\partial x_1}\left(f_1(x_1) + g_1(x_1)s(x_1)\right) + \frac{\partial V}{\partial x_1}g_1(x_1)z + z\upsilon \leqslant -\omega(x_1) - kz^2, \tag{1.62}$$

где $k > 0$ — настраиваемый коэффициент.

Подставив в неравенство (1.62) выражения для z, $\dot{s}$ и υ из (1.57), (1.59) и (1.60) соответственно, получим функцию управления u' для системы (1.54)

$$u' = \frac{\partial s}{\partial x_1}\left(f_1(x_1) + g_1(x_1)x_2\right) - \frac{\partial V}{\partial x_1}g_1(x_1) - k\left(x_2 - s(x_1)\right). \tag{1.63}$$

Возвращаясь к модели объекта управления в общем виде (1.52) и подставляя (1.63) в (1.53), получим функцию управления u для системы (1.52)

$$\begin{aligned} u = \ & \frac{1}{g_2(x_1,x_2)}\Big(\frac{\partial s}{\partial x_1}\left(f_1(x_1) + g_1(x_1)x_2\right) - \\ & -\frac{\partial V}{\partial x_1}g_1(x_1) - k\left(x_2 - s(x_1)\right) - f_2(x_1,x_2)\Big). \end{aligned}$$

В качестве примера рассмотрим задачу стабилизации маятника [80; 81]. Объект описывается системой уравнений (1.49). В соответствии с алгоритмом бэкстеппинга, рассматриваем каждое уравнение системы (1.49), делая интегратор устойчивым по Ляпунова, путем добавления обратной связи.

Так для первого уравнения x_2 является своего рода виртуальным управлением. Тогда для первого уравнения функция Ляпунова будет иметь вид

$$V(x_1) = \frac{1}{2}x_1^2,$$

а ее производная в соответствии с (1.39) будет равна

$$\dot{V}(x_1) = x_1 x_2. \tag{1.64}$$

Чтобы обеспечить устойчивость интегратора, производная функции Ляпунова должна быть отрицательно полуопределенной

$$\dot{V}(x_1) \leqslant -\omega(x_1). \tag{1.65}$$

В качестве функции $\omega(x_1)$ возьмем функцию вида

$$\omega(x_1) = k_1 x_1^2. \tag{1.66}$$

Тогда, подставив (1.64) и (1.66) в (1.65), можно выразить вид обратной связи в форме стабилизационной функции, которая будет стабилизировать один интегратор

$$s(x_1) = -k_1 x_1.$$

Введем ошибку состояния z как отклонение x_2 от $s(x_1)$

$$z = x_2 - s(x_1) = x_2 + k_1 x_1. \tag{1.67}$$

С учетом (1.67) система (1.49) примет вид

$$\begin{cases} \dot{x}_1 = z + s(x_1) = z - k_1 x_1 \\ \dot{z} = \dot{x}_2 - \dot{s}(x_1) = a\sin(x_1) + u + k_1 \dot{x}_1 = a\sin(x_1) + u + k_1(z - k_1 x_1) \end{cases}. \tag{1.68}$$

Ошибку состояния z необходимо ввести в функцию Ляпунова. Тогда в качестве новой возможной функции Ляпунова можно принять квадратичную функцию

$$V(x_1, z) = \frac{1}{2}x_1^2 + \frac{1}{2}z^2. \tag{1.69}$$

Аналогично описанной выше схеме необходимо вычислить производную функции Ляпунова $\dot{V}(x_1, z)$, которая для обеспечения устойчивости уже всей системы должна быть отрицательно полуопределенной

$$\dot{V}(x_1, z) \leqslant -\omega(x_1, z). \tag{1.70}$$

В качестве функции $\omega\,(x_1,z)$ можно принять сумму квадратов вектора состояния модифицированной системы (1.68)

$$\omega\,(x_1,z) = k_1 x_1^2 + k_2 z^2. \tag{1.71}$$

Подставив производную функции Ляпунова (1.69) и функцию (1.71) в неравенство (1.70), получим

$$x_1 + a\sin(x_1) + u + k_1 x_1 \leqslant -k_2 z,$$

откуда выразим функцию управления

$$u = -x_1(1 + k_1 k_2) - x_2(k_1 + k_2) - a\sin(x_1).$$

Подбор значений коэффициентов k_1 и k_2 позволяет найти баланс между величиной перерегулирования и временем переходного процесса. При значении одного из коэффициентов равном нулю, а другого — единице получаем выражение, эквивалентное управлению (1.51), найденного методом функций Ляпунова.

В сравнении с методом на основе функций Ляпунова, бэкстеппинг позволяет избегать трудностей, связанных с линеаризацией объекта, и сравнительно легче находить решения исходной задачи. Однако данный метод нельзя отнести к классу универсальных. При использовании бэкстеппинга от разработчика требуется наличие определенного опыта в выборе функций Ляпунова и коэффициентов обратной связи. На практике решения многих прикладных задач оказываются нестандартными, а производимые расчёты — довольно громоздкими [70].

Определенная эффективность бэкстеппинга ощутима только в случаях синтеза системы управления простыми нелинейными объектами малых порядков. Так в работе [81] метод бэкстеппинг реализован только для некоторых систем первого и второго порядков. Увеличение порядка и сложности модели объекта управления приводит к значительному увеличению трудностей получения аналитического решения. Известные модификации метода [77; 119] позволяют получать решения для более сложных объектов управления, но не избавляют разработчика от сложности и громоздкости вычислений.

1.2.6 Метод аналитического конструирования агрегированных регуляторов

Метод аналитического конструирования агрегированных регуляторов (АКАР) был предложен А.А. Колесниковым в 1985 году [36] и получил развитие в работах [37—39]. Данный подход позиционируется как эффективный метод решения задачи синтеза системы управления нелинейными многомерными и многосвязными объектами. Метод АКАР базируется на принципе «расширения – сжатия» фазового пространства, сформулированного А.А. Колесниковым. По словам автора, данных принцип позволят методу АКАР преодолевать так называемое «проклятие размерности» при решении задач синтеза для сложных многомерных систем [40].

В основе метода лежит синергетическая концепция анализа и синтеза нелинейных обратных связей. Согласно методу АКАР делается предположение о возможности формирования в фазовом пространстве системы управления притягивающих многообразий — аттракторов. В зоне действия аттрактора дальнейшее поведение системы определяется его свойствами. Построенные обратные связи должны обеспечивать асимптотическую устойчивость системы по отношению к аттракторам. При этом движение системы управления разбивается на этап устремления к аттрактору и этап асимптотически устойчивого движения на желаемом аттракторе [68].

Пусть объект управления задан в виде (1.2), пусть также заданы множество начальных условий (1.3) и терминальные условия (1.4). Тогда, в соответствии с постановкой задачи аналитического конструирования агрегированных регуляторов, требуется найти такую функцию управления

$$\mathbf{u}(\boldsymbol{\psi}) = \mathbf{u}(\mathbf{x}), \tag{1.72}$$

которая обеспечивает перемещение объекта управления (1.2) из любого начального состояния из множества (1.3) сначала в окрестность многообразия

$$\boldsymbol{\psi}(\mathbf{x}) = 0, \tag{1.73}$$

называемого аттрактором, а затем асимптотически устойчивое движение вдоль него в заданное терминальными условиями (1.4) состояние.

В методе АКАР система основных функциональных уравнений имеет следующий вид

$$\mathbf{T}\dot{\boldsymbol{\psi}}(t) + \boldsymbol{\varphi}(\boldsymbol{\psi}) = 0. \tag{1.74}$$

Движение объекта управления должно удовлетворять системе основных функциональных уравнений (1.74). При этом функции $\boldsymbol{\varphi}(\boldsymbol{\psi})$ следует выбирать таким образом, чтобы обеспечивать асимптотическую устойчивость системы (1.74) и желаемый вид фазовой траектории движения к аттракторам (1.73). В простейшем случае $\boldsymbol{\varphi}(\boldsymbol{\psi}) = \boldsymbol{\psi}$. Притягивающие многообразия (1.73) являются своего рода целевыми множествами, к которым притягивается объект управления. Движение может осуществляться от одного аттрактора к другому до достижения конечного аттрактора и движения по нему до терминального состояния.

Для поставленной задачи функционал качества имеет вид

$$J = \int_0^{t_f} \left(\boldsymbol{\varphi}^2(\boldsymbol{\psi}) + \mathbf{T}^2 \dot{\boldsymbol{\psi}}^2(t) \right) dt \to \min. \tag{1.75}$$

Синтезированная системы управления (1.72) должна обеспечивать минимум функционалу (1.75), однако сам функционал (1.75) непосредственного участия в процедуре синтеза системы управления не принимает. Это является основным отличием метода АКАР от других рассмотренных выше методов.

В качестве примера решения прикладной задачи синтеза системы управления методом АКАР рассмотрим задачу синтеза системы управления рулем высоты, обеспечивающей стабилизирующее движение самолета по углу атаки [15; 41]. Математическая модель данной системы описывается системой дифференциальных уравнений (1.32). Необходимо найти функцию управления рулем высоты $\tilde{u}(\mathbf{x})$, обеспечивающую стабилизирующее движение самолета $x_1^* = x_2^* = 0$.

В соответствии с методом АКАР введем функцию преобразования (1.73) следующего вида

$$\psi(x_1,x_2) = \beta x_1 + x_2. \tag{1.76}$$

На основе (1.76) построим функционал качества (1.75), который для данной задачи примет вид

$$J = \int_0^{t_f} \left(\psi^2 + T^2 \dot{\psi}^2(t) \right) dt \to \min. \tag{1.77}$$

Подстановка в (1.77) функции (1.76) и ее первой производной позволяет привести функционал качества (1.77) к следующему виду

$$J = \int_0^{t_f} \left(\left(\frac{\beta^2}{T^2} + a_1 \right) x_1^2 + \left(\beta^2 + a_2^2 + \frac{1}{T^2} \right) x_2^2 + \right. \\ \left. + 2a_3 \left(a_1 x_1 + a_2 x_2 \right) u + a_3^2 u^2 \right) dt \to \min . \tag{1.78}$$

Далее используем основное функциональное уравнение метода АКАР (1.74), которое для данной задачи имеет вид

$$T\dot{\psi}(t) + \psi = 0. \tag{1.79}$$

Данное основное функциональное уравнение доставляет минимум функционалу (1.77), а значит и функционалу (1.78). Тогда, с учетом (1.76) и (1.79), функция управления для системы (1.32) будет иметь вид

$$\tilde{u}(x_1,x_2) = -\frac{1}{a_3} \left(a_1 + \frac{\beta}{T} \right) x_1 - \frac{1}{a_3} \left(a_2 + \beta + \frac{1}{T} \right) x_2.$$

Метод АКАР показывает более высокую эффективность и простоту реализации, чем рассмотренные выше методы. Данный метод значительно лучше работает с нелинейными системами. В тоже время для сложных динамических систем не всегда представляется возможным произвести выбор функции преобразования. Сложность такого выбора может быть сравнима с интуитивным построением самой системы управления. Также нужно учитывать ряд серьёзных ограничений, накладываемых на исходную задачу, вплоть до отсутствия ее аналитического решения. Следует особо отметить, что сам алгоритм поиска решения методом АКАР, заключающийся в нахождении притягивающих многообразий — аттракторов и движении в их окрестности, содержит своего рода подводные камни, которые могут оказать существенное негативное влияние на решение. В самом деле, известно, что движение объекта в окрестности оптимальной траектории может сильно отличаться по значению критерия качества от движения по самой оптимальной траектории.

1.3 Постановка задачи численного синтеза системы управления на основе многокритериальной структурно-параметрической оптимизации

Высокая сложность большинства прикладных задач синтеза системы управления делает невозможным применение методов их аналитического решения. Действительно, для большого класса прикладных задач рассмотренные в разделе 1.2 методы аналитического синтеза системы управления принимают излишне громоздкий вид или эффективны в весьма узких диапазонах условий применения.

Когда применение аналитических методов не является возможным, целесообразно использовать численные методы поиска решения. Технологический прогресс, достигнутый за прошедшее с момента появления первых ЭВМ время, позволил значительно увеличить эффективность, точность и быстродействие применяемых численных методов решения поставленной задачи. С другой стороны, тот же технологический прогресс предопределил как значительное расширение сферы применения систем автоматического управления, так и усложнение объектов управления. Всё это делает поиск новых эффективных численных методов решения задачи синтеза системы управления актуальным по сей день.

Приведем постановку задачи численного синтеза системы управления на основе многокритериальной структурно-параметрической оптимизации [23].

Для системы (1.2) можно получить частное численное решение для одного начального состояния $\mathbf{x}(0)$ из множества (1.3), оптимальное по значению функционала (1.5). Такое решение для данного начального состояния будет совпадать с решением, которое можно было бы получить с помощью искомой системы управления. Однако определить структуру многомерной функции управления по одному частному решению нельзя, так как найденное частное решение может не доставлять оптимального управления из иного начального состояния из множества (1.3). Согласно постановке задачи синтеза системы управления многомерная функция управления должна обеспечивать оптимальное решение для всех начальных состояний из множества (1.3).

Заменим непрерывное множество (1.3) конечным множеством из N элементов

$$\tilde{\mathrm{X}}_0 = \left\{\mathbf{x}^{0,1},\ldots,\mathbf{x}^{0,N}\right\}. \tag{1.80}$$

Введем в рассмотрение функционалы качества

$$J_1 = \sum_{j=1}^{N}\left(\int_0^{t_f} f_0\left(\mathbf{x}(t),\mathbf{u}(t)\right)dt\right)_{\mathbf{x}^{0,j}} \to \min, \tag{1.81}$$

$$J_2 = \sum_{j=1}^{N}\left(\sum_{k=1}^{L}\left|\varphi_k\left(\mathbf{x}(t_f)\right)\right|\right)_{\mathbf{x}^{0,j}} \to \min, \tag{1.82}$$

где $\varphi_k\left(\mathbf{x}(t_f)\right)$, $k=\overline{1,L}$, $1 \leqslant L \leqslant n$ — функции оценки достижения терминального условия (1.4), при $\mathbf{x}(t_f) = \mathbf{x}^f \Rightarrow \varphi_k\left(\mathbf{x}(t_f)\right) = 0$.

Тогда решением поставленной задачи будет система управления в виде функции от координат пространства состояний

$$\tilde{\mathbf{u}} = \mathbf{h}(\mathbf{x}) \in \Pi \tag{1.83}$$

где Π — множество Парето оптимальных решений по значениям функционалов (1.81) и (1.82).

Пусть известны функции управления для N одноточечных задач из N разных начальных состояний $\mathbf{x}\,(0) = \mathbf{x}^{0,j}$, $j=\overline{1,N}$,

$$\begin{cases} \tilde{\mathbf{u}}(\mathbf{x}^{0,1}) = \mathbf{h}^1(\mathbf{x}) \\ \tilde{\mathbf{u}}(\mathbf{x}^{0,2}) = \mathbf{h}^2(\mathbf{x}) \\ \ldots \\ \tilde{\mathbf{u}}(\mathbf{x}^{0,N}) = \mathbf{h}^N(\mathbf{x}) \end{cases} \tag{1.84}$$

такие, что доставляют минимум соответствующим функционалам качества

$$\begin{cases} \tilde{J}(\mathbf{x}^{0,1}) \\ \tilde{J}(\mathbf{x}^{0,2}) \\ \ldots \\ \tilde{J}(\mathbf{x}^{0,N}) \end{cases}, \tag{1.85}$$

где $\tilde{J}(\mathbf{x}^{0,j})$ — свертка функционалов (1.81) и (1.82), вычисленных для начального значения $\mathbf{x}\,(0) = \mathbf{x}^{0,j}$, $j=\overline{1,N}$.

Тогда существует функция управления для N-точечной задачи синтеза в виде функции от координат пространства состояний

$$\mathbf{u}(\mathbf{x}^{0,1},\mathbf{x}^{0,2},\ldots,\mathbf{x}^{0,N}) = \mathbf{h}(\mathbf{x}),$$

которая доставляет значение функционала $\tilde{J}$ не хуже, чем сумма функционалов (1.85) для N одноточечных задач

$$\tilde{J}(\mathbf{x}^{0,1},x^{0,2},\ldots,\mathbf{x}^{0,N}) = \tilde{J}(\mathbf{x}^{0,1}) + \tilde{J}(\mathbf{x}^{0,2}) + \ldots + \tilde{J}(\mathbf{x}^{0,N}).$$

С учетом (1.84) функция управления (1.83), обеспечивающая перевод объекта управления (1.2) из множества начальных состояний (1.80) в терминальное состояние (1.4) и обеспечивающая оптимальное по Парето значение функционалов (1.81) и (1.82), примет вид

$$\tilde{\mathbf{u}} = \mathbf{h}(\mathbf{x}) = \sum_{j=1}^{N} \vartheta\left(\varepsilon - \left\|\mathbf{x}(0) - \mathbf{x}^{0,j}\right\|\right) \mathbf{h}^{j}(\mathbf{x}),$$

где $\vartheta(A)$ — функция Хэвисайда (1.22), ε — малая величина, $\varepsilon < \dfrac{\left\|\mathbf{x}^{0,j} - \mathbf{x}^{0,k}\right\|}{2}$, $j,k = \overline{1,N}$.

Рассмотренную постановку задачи численного синтеза системы управления также можно назвать многокритериальным синтезом ввиду наличия сразу нескольких функционалов качества (1.81) и (1.82). Такая форма постановки задачи синтеза в большей степени свойственна прикладным инженерным задачам.

1.4 Численные методы решения задачи синтеза системы управления на основе многокритериальной структурно-параметрической оптимизации

До последнего времени одним из наиболее распространенных способов решения задачи синтеза системы управления являлся параметрический синтез. При таком подходе предварительный анализ объекта управления и обычно известный заранее диапазон изменения его возможных состояний, как правило, позволяет разработчику сделать предположение о структуре функциональной зависимости управляющих воздействий от текущего состояния объекта.

В отсутствии предварительных данных разработчик может выбрать структуру системы управления на основе собственного опыта или даже интуиции. Функциональная зависимость задается до некоторого числа неизвестных параметров, а дальнейшая настройка системы управления сводится к поиску оптимальных значений данных параметров. Использование разных оптимизационных алгоритмов позволяет заметно увеличить точность выбранной функциональной зависимости. Однако более значимую роль в достижении требуемых критериев управления играет наличие предварительной информации о форме функциональной связи, а не настройка её параметров. Таким образом, параметрический синтез не является универсальным подходом и может быть совершенно неэффективен для сложных систем.

Далее речь пойдет о численных методах структурно-параметрического решения задачи синтеза управления. В нем производится одновременный поиск и формы функциональной зависимости управления, и параметров, делающих эту зависимость более точной. В отличие от параметрического синтеза, такой подход не требует наличия информации о форме функциональной связи управляющих воздействий от текущего состояния объекта управления. Также не требуется предварительной подготовки к поиску решения со стороны разработчика. Поиск решения основывается на математической модели, описывающей объект управления, на данных об ограничениях на векторы состояния и управления. Отсюда следует универсальность и бóльшая научная и прикладная значимость такого подхода.

Долгое время не существовало эффективных подходов к численному структурно-параметрическому синтезу. Для подбора оптимальной структуры функциональной зависимости требовалось обеспечить возможность поиска на пространстве математических выражений. Прорывом в данной области послужило бурное развитие вычислительной техники в конце прошлого века и появление методов машинного обучения. В 1989 году американским ученым Джоном Козой был предложен метод генетического программирования [117], в основе которого лежал поиск на нечисловом пространстве символов. Изначально метод задумывался для автоматического составления компьютерных программ. Поиск оптимального выражения осуществлялся с помощью генетического алгоритма [106]. Позднее в 1999 году Джон Коза с соавторами представил работу, посвященную структурно-параметрическому синтезу системы управления на основе метода генетического программирования [86].

Широкого прикладного распространения метод генетического программирования в области структурно-параметрического синтеза не получил. Возможно, это обусловлено некоторыми недостатками метода, связанными с необходимостью проверки корректности выражения на каждом этапе поиска. Однако данный метод стал отправной точкой для целого класса новых методов поиска на нечисловом пространстве, получивших общее название методов символьной регрессии [23]. Среди методов данного класса следует отметить метод декартового генетического программирования [124], метод грамматической эволюции [136], метод аналитического программирования [154] и метод сетевого оператора, предложенный профессором А.И. Дивеевым в 2008 году [96]. Различия в методах символьной регрессии заключаются в методике кодирования математических выражений и осуществления преобразования одной записи в другую. В случае с задачей синтеза управления искомой функцией управления будет наиболее оптимальное математическое выражение, полученное в ходе итерационного процесса преобразования записей. Таким образом, любой метод символьной регрессии может с той или иной эффективностью быть использован для решения задачи синтеза управления. Более подробно методы символьной регрессии рассмотрены в Главе 2.

Несмотря на то, что любые методы символьной регрессии могут применяться для поиска решения задачи синтеза управления, наибольшее распространение в этом направлении получил метод сетевого оператора. Отличительной особенностью, выделяющей метод сетевого оператора среди других методов класса, является то, что он создавался для целей применения именно к задачам синтеза системы управления и учитывает их специфику [22]. Использование данного метода позволило достичь значительных успехов в решении многих прикладных задач идентификации математических моделей и задач синтеза системы управления. В своих работах А.И. Дивеев и его ученики рассматривали такие сложные объекты управления как химические реакторы, гусеничные и колесные мобильные роботы, беспилотные летательные аппараты и квадрокоптеры, космические спутники, космические аппараты и многие другие [21—23; 163; 96]. Метод сетевого оператора, а также другие методы символьной регрессии в данных работах использовались для многокритериальной оптимизации структуры и параметров искомой функции управления.

Рассмотрим задачу численного синтеза системы управления на основе структурно-параметрической многокритериальной оптимизации с помощью ме-

тодов символьной регрессии [22]. Пусть объект управления задан в виде (1.2), множество начальных значений дискретно и имеет вид (1.80), а терминальные условия описываются уравнением (1.4). При решении задачи используем функционалы качества (1.81) и (1.82).

Введем обозначение записи функции управления

$$\mathbf{g}\left(\mathbf{x},\mathbf{s}\right), \tag{1.86}$$

где $\mathbf{s}$ — вектор постоянных параметров, заданной размерности K, $\mathbf{s} = \left[s_1 \ldots s_K\right]^T$. В методах символьной регрессии возможность записи структуры функции управления в виде (1.86) ограничена набором используемых символов.

Решением задачи численного синтеза системы управления на основе структурно-параметрической многокритериальной оптимизации с помощью методов символьной регрессии будет запись оптимальной структуры функции

$$\tilde{\mathbf{u}} = \mathbf{g}^*\left(\mathbf{x},\mathbf{s}^*\right) \tag{1.87}$$

и оптимальные значения ее параметров $\mathbf{s}^*$.

Выбор решения в форме записи структуры функции управления (1.87) осуществляется на множестве Парето в пространстве функционалов (1.81) и (1.82)

$$\Pi = \left\{\tilde{\mathbf{g}}^i\left(\mathbf{x},\tilde{\mathbf{s}}^i\right) :\ i = 1,2,\ldots\right\}. \tag{1.88}$$

Для поиска решения методами символьной регрессии строится множество возможных решений в форме записей функций

$$\mathrm{G} = \left\{\mathbf{g}^j\left(\mathbf{x},\mathbf{s}^j\right) :\ j = 1,2,\ldots\right\} \tag{1.89}$$

и применяются операции генетического алгоритма для получения записей $\tilde{\mathbf{g}}^i\left(\mathbf{x},\tilde{\mathbf{s}}^i\right)$, удовлетворяющих задаче (1.2), (1.4), (1.80) — (1.82). Таким образом, множество Парето (1.88) является подмножеством множества возможных решений (1.89)

$$\Pi \subseteq \mathrm{G}.$$

Как видно из описания, синтез системы управления на основе многокритериальной оптимизации структуры функции управления и одновременного

поиска оптимальных значений ее параметров реализует подход прямого подбора оптимального решения на пространстве всех возможных решений. С помощью данного подхода впервые были получены решения для большого числа задач, ранее для которых получить ни аналитическое, ни численное решение не представлялось возможным. Однако данный подход обладает рядом недостатков.

Численное решение задачи структурно-параметрической многокритериальной оптимизации требует дискретизации множества начальных состояний (1.3) и, как следствие, перехода от функционала (1.5) к функционалу (1.81) в виде суммы функционалов качества для каждого начального состояния из конечного множества (1.80). Для задач, учитывающих расстояния от начального состояния системы до цели управления, при таком переходе начальные состояния из множества (1.80), расположенные дальше от терминального состояния (1.4), будут вносить в сумму функционала качества (1.81) более существенный вклад, чем начальные состояния, расположенные ближе к цели (1.4). Таким образом, найденное решение с использованием функционала (1.81) может оказаться не универсальным, так как оптимизация решения для расположенных далеко от цели начальных состояний будет сильней влиять на уменьшение значения функционала.

Чтобы проверить и убедиться в универсальности найденного решения задачи синтеза в работе [22] предлагается использовать критерий оптимальности на основе численного решения задачи оптимального управления

$$\max_t \left|\tilde{\mathbf{v}}^j(t) - \left(\tilde{\mathbf{g}}^i\left(\mathbf{x},\tilde{\mathbf{s}}^i\right)\right)_{\mathbf{x}^{0,j}}\right| \leqslant \delta,\ j = \overline{1,N}, \tag{1.90}$$

где $\tilde{\mathbf{v}}^j(t)$ — численное решение задачи оптимального управления для объекта (1.2), терминального условия (1.4) и начального состояния из множества (1.80) $\mathbf{x}(0) = \mathbf{x}^{0,j} \in \tilde{\mathrm{X}}_0$, $\left(\tilde{\mathbf{g}}^i\left(\mathbf{x},\tilde{\mathbf{s}}^i\right)\right)_{\mathbf{x}^{0,j}}$ — решение задачи структурно-параметрической многокритериальной оптимизации из множества (1.88), вычисленное для того же начального условия $\mathbf{x}(0) = \mathbf{x}^{0,j} \in \tilde{\mathrm{X}}_0$. Следует отметить, что такая проверка является достаточно трудоёмкой.

Другой недостаток подхода на основе структурно-параметрической многокритериальной оптимизации вытекает из описания приведенного выше недостатка. Поиск решения в форме оптимальной записи структуры функции управления (1.87) среди всех возможных решений из множества (1.89) осуществляется без учета его близости к оптимальному. Для определения близости

найденного решения к оптимальному в данном подходе отсутствует соответствующая метрика. Это в свою очередь не позволяет оценить качество решения без проведения дополнительных вычислений. Для оценки уже найденного решения можно воспользоваться методикой на основе вычисления критерия оптимальности (1.90), описанной выше. Но, во-первых, это связано с необходимостью проведения трудоемких вычислений. Во-вторых, такая оценка не может быть использована для уточнения найденного решения.

Таким образом, известные численные методы структурно-параметрического синтеза системы управления позволяют осуществлять поиск функции управления, но не позволяют оценить близость найденного решения к оптимальному, а также обладают другими недостатками. В прикладных задачах такое решение в составе блока управления даст возможность получать неплохие траектории достижения цели управления. Однако достаточных оснований для утверждения, что с заданной величиной погрешности полученная траектория является оптимальной, такое решение не предоставит.

Создание метода решения задачи синтеза системы управления, для которого будет определена оценка близости возможного решения к оптимальному, позволило бы качественно оценивать получаемые в процессе поиска решения. В составе блока управления найденное таким образом решение будет обеспечивать достижение цели управления по траектории близкой к оптимальной с известной оценкой данной близости. Создание такого блока управления является необходимой частью этапа проектирования современных устройств в области самолетостроения и ракетостроения, создания роботизированных устройств, мобильных роботов, роверов, беспилотных летательных аппаратов, беспилотных автомобилей и др.

Далее в Главе 3 автором предлагается новый подход численного решения задачи синтеза системы управления и нахождения структуры многомерной функции управления на основе аппроксимации множества оптимальных траекторий методами символьной регрессии. Предлагаемый метод лишен некоторых недостатков численных подходов, рассмотренных выше, и больше отвечает требованиям современных прикладных задач.

Выводы по Главе 1

Задача общего синтеза системы управления считается основной задачей в современной теории управления. Решением задачи синтеза является многомерная функция управления от компонент вектора состояния объекта управления, обеспечивающая оптимальное по заданному критерию качества достижение терминального состояния. При этом начальное состояние объекта управления может принадлежать некой области или даже всему пространству состояний.

Обзор известных методов решения задачи общего синтеза системы управления показал целый ряд их недостатков. Данные методы не предоставляют универсальных инструментов решения поставленной задачи и в основном применимы только для несложных объектов малой размерности. Для сложных нелинейных систем рассмотренные методы как правило неприменимы.

Метод динамического программирования, применяемый для решения задачи синтеза системы управления не позволяет получить структуру функции управления в явном виде. Полученное решение будет представлять собой множество значений вектора управления для множества значений вектора состояния объекта управления. Данное решение не предоставит оптимальное управление в случае изменения начального состояния объекта. То есть, также как и решение задачи оптимального управления, синтез на основе динамического программирования даёт решение только для одного начального состояния объекта. При этом сложность получения такого решения значительно выше сложности решения задачи оптимального управления.

Решение задачи синтеза на основе принципа максимума Понтрягина требует большого числа аналитических преобразований и необходимости решения систем дифференциальных уравнений. Без специальных преобразований найденное решение будет неприменимым для случаев особого управления.

При использовании метода АКОР для решения задачи синтеза системы управления нелинейными объектами, сложности, связанные с необходимостью решения уравнения Гамильтона-Якоби-Беллмана, возрастают с увеличением порядка объекта управления. Таким образом, метод АКОР применяется только для линейных систем.

Синтез системы управления на основе функций Ляпунова также ограничен линейными системами невысокого порядка. Аналогично предыдущим

методам, сложности при работе с системами высокого порядка нарастают лавинообразно.

В методе бэкстеппинга удается достичь устойчивости интегратора путём добавления обратной связи для каждого дифференциального уравнения, описывающего объект управления. Это позволяет избежать трудностей с линеаризацией объекта, но одновременно требует нестандартных подходов при выборе функций Ляпунова и коэффициентов обратной связи. Данные сложности возлагаются на разработчика и не всегда могут быть разрешены.

Метод АКАР для решения задачи синтеза системы управления наиболее эффективный из рассмотренных аналитических методов. Он применим для класса нелинейных систем и заключается в поиске притягивающих многообразий. При этом от разработчика требуется осуществить выбор функции преобразования. Ввиду отсутствия универсальных подходов, такой выбор в большинстве случаев осуществляется на основе опыта и интуиции самого разработчика.

Обзор численных методов синтеза на основе структурно-параметрической многокритериальной оптимизации методами символьной регрессии показал их применимость для гораздо более широкого класса прикладных задач. Применение методов символьной регрессии впервые позволило реализовывать поиск на нечисловом пространстве математических функций и таким образом осуществлять одновременный поиск структуры и оптимальных значений параметров функции управления. Однако рассмотренные подходы также содержат ряд недостатков, основным из которых является невозможность оценить близость найденного решения к оптимальному.

Глава 2. Обзор методов символьной регрессии для синтеза математических выражений

2.1 Методы символьной регрессии для синтеза математических выражений

Как следует из названия, методы символьной регрессии предназначены для поиска взаимосвязи между символьным выражением и набором независимых переменных. При этом в качестве символьного выражения могут выступать как числовые, так и нечисловые многомерные структуры, такие как: компьютерные программы, электрические цепи, химические формулы, математические выражения и др.

Задача синтеза управления подразумевает поиск математического выражения функции управления. Таким образом, данную задачу можно решать путем применения методов символьной регрессии. Искомое символьное выражение при этом будет представлять собой математическое выражение в виде суперпозиции заданных функций и арифметических операций над ними.

Задачи поиска математического выражения решались задолго до появления методов символьной регрессии. Однако в таких задачах структуру математического выражения обычно задавал сам исследователь с точностью до некоторого числа неизвестных параметров. Далее использовался один из оптимизационных алгоритмов для поиска оптимальных значений параметров. Вид выражения и число параметров выбирались на основе собственного опыта и предварительного исследования задачи. Другими словами, если исследователь обладает знаниями о форме функциональной связи между входами и выходами у искомого выражения, то классические методы безусловной оптимизации позволят найти такие значения параметров, которые делают функциональную зависимость в предполагаемом выражении более точной. В случае, когда исследователь не обладает информацией о форме функциональной связи, методы оптимизации окажутся бессильны.

Отличительной особенностью методов символьной регрессии является то, что они позволяют одновременно искать и структуру математического выражения, и значения его параметров. Они не требуют обязательного указания

формы функциональной связи. При использовании методов символьной регрессии предварительный анализ задачи исследователем может положительно влиять на скорость поиска, но не на его конечный результат.

Все методы символьной регрессии осуществляют поиск на нечисловом пространстве кодов. Понятие кодирования является одним из основополагающих для данного класса методов. Под кодированием подразумевается обратимая операция записи числовой или нечисловой структуры в виде кода, специфичного для каждого конкретного метода символьной регрессии. Так в методе генетического программирования, который является первым методом из класса методов символьной регрессии, автором Джоном Козой было предложено использовать кодирование в виде префиксной польской записи [115]. Далее к полученной закодированной записи применялись операции скрещивания и мутации генетического алгоритма. Выбор генетического алгоритма для осуществления поиска оптимального кода не был случаен. Операции скрещивания и мутации не используют арифметические операции, а, следовательно, применимы для поиска на нечисловых пространствах. Другие известные методы символьной регрессии для поиска также используют операции на основе модификаций генетического алгоритма.

Таким образом, метод символьной регрессии в общем виде состоит из алгоритма кодирования символьного выражения и алгоритма поиска на пространстве кодов. Различия в реализации каждой из двух частей формируют отличительные особенности каждого метода из класса.

При решении задачи синтеза системы управления с помощью методов символьной регрессии кодированию подвергается математическое выражение многомерной функции управления. Сложные математические выражения могут быть получены с помощью композиции функций из множества простых функций от одного или нескольких аргументов. В качестве аргументов могут использоваться переменные и параметры математического выражения или функции без аргументов.

В настоящее время известно более 10 различных методов символьной регрессии [23]. Помимо уже упомянутого метода генетического программирования также следует отметить метод декартового генетического программирования [124], метод грамматической эволюции [136], метод индуктивного программирования [129], метод аналитического программирования [154], метод сетевого оператора [96] и метод матриц разбора [122]. Все перечисленные ме-

тоды осуществляют поиск оптимальной структуры символьного выражения на множестве нечисловых кодов. При таком поиске арифметические операции не могут быть применены, а, следовательно, нет возможности применять известные численные методы безусловной оптимизации, основанные на арифметических преобразованиях текущего решения. В генетическом алгоритме эволюционные преобразования, основанные на операциях скрещивания и мутации, не используют арифметические действия. Таким образом, генетический алгоритм и его модификации являются единственными на текущий момент инструментами реализации поиска оптимальной структуры выражения на нечисловом пространстве кодов в методах символьной регрессии.

2.2 Обзор методов символьной регрессии

2.2.1 Метод генетического программирования

Идея использования эволюционных алгоритмов для поиска оптимального символьного представления выражения принадлежит профессору Стэндфордского университета Джону Коза [120]. В основе метода лежит генетический алгоритм, осуществляющий операции скрещивания и мутации на строках символов. Данный подход получил название метод генетического программирования [102; 117].

В настоящее время генетическое программирование является самым известным методом из класса методов символьной регрессии. Известно множество успешных реализаций этого метода для решения разнообразных задач синтеза оптимальных структур, например, решение задачи синтеза сложной электрической схемы [118].

Изначально метод генетического программирования задумывался для решения задач автоматического составления компьютерных программ на языке программирования LISP, в котором выражения представляются в виде префиксной записи [116]. При этом каждому символу или коду соответствует определенная функция или операция из языка программирования LISP.

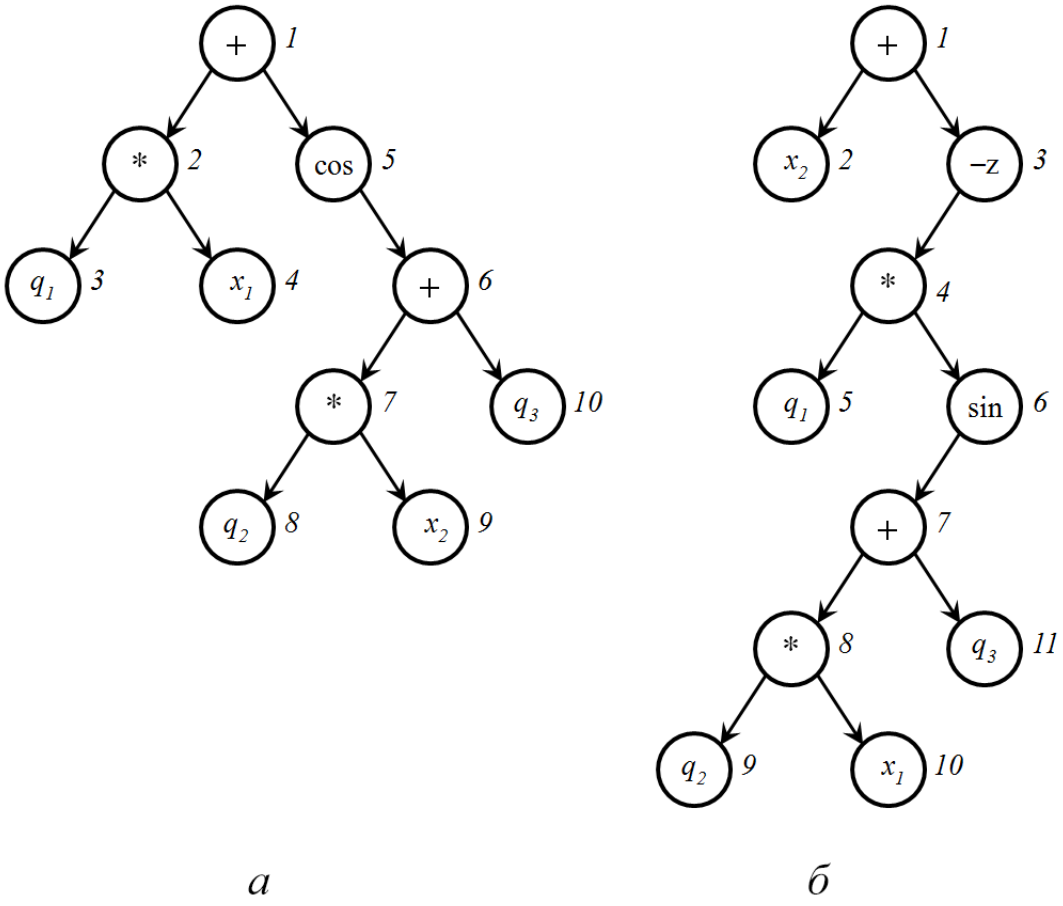
Те же принципы кодирования применимы и для представления математического выражения в виде кода. Здесь каждому символу ставится в соответствие некоторая математическая функция или арифметическая операция. Математическая функция дополнительно характеризуется числом её параметров. Переменные и параметры математического выражения можно представить, как функции без аргументов. Вид математического выражения и соответствие между функциями и их аргументами определяется порядком следования символов в закодированной строке.

Визуально принципы кодирования и поиска нового выражения методом генетического программирования часто представляются в виде графа. Для примера рассмотрим два математических выражения:

$$f_1(\mathbf{x}) = q_1 x_1 + \cos(q_2 x_2 + q_3), \tag{2.1}$$

$$f_2(\mathbf{x}) = x_2 - q_1 \sin(q_2 x_1 + q_3). \tag{2.2}$$

Их представление в виде графа изображено на Рис. 2.1.



а *б*

Рисунок 2.1 — Представление математических выражений в виде графа: *а* — выражение (2.1); *б* — выражение (2.2).

При выполнении генетической операции скрещивания над двумя представлениями математических выражений (2.1) и (2.2), случайным образом определяются номера узлов, для которых будет произведен обмен вместе со всеми дочерними элементами. Результат применения операции скрещивания представлен на Рис. 2.2.

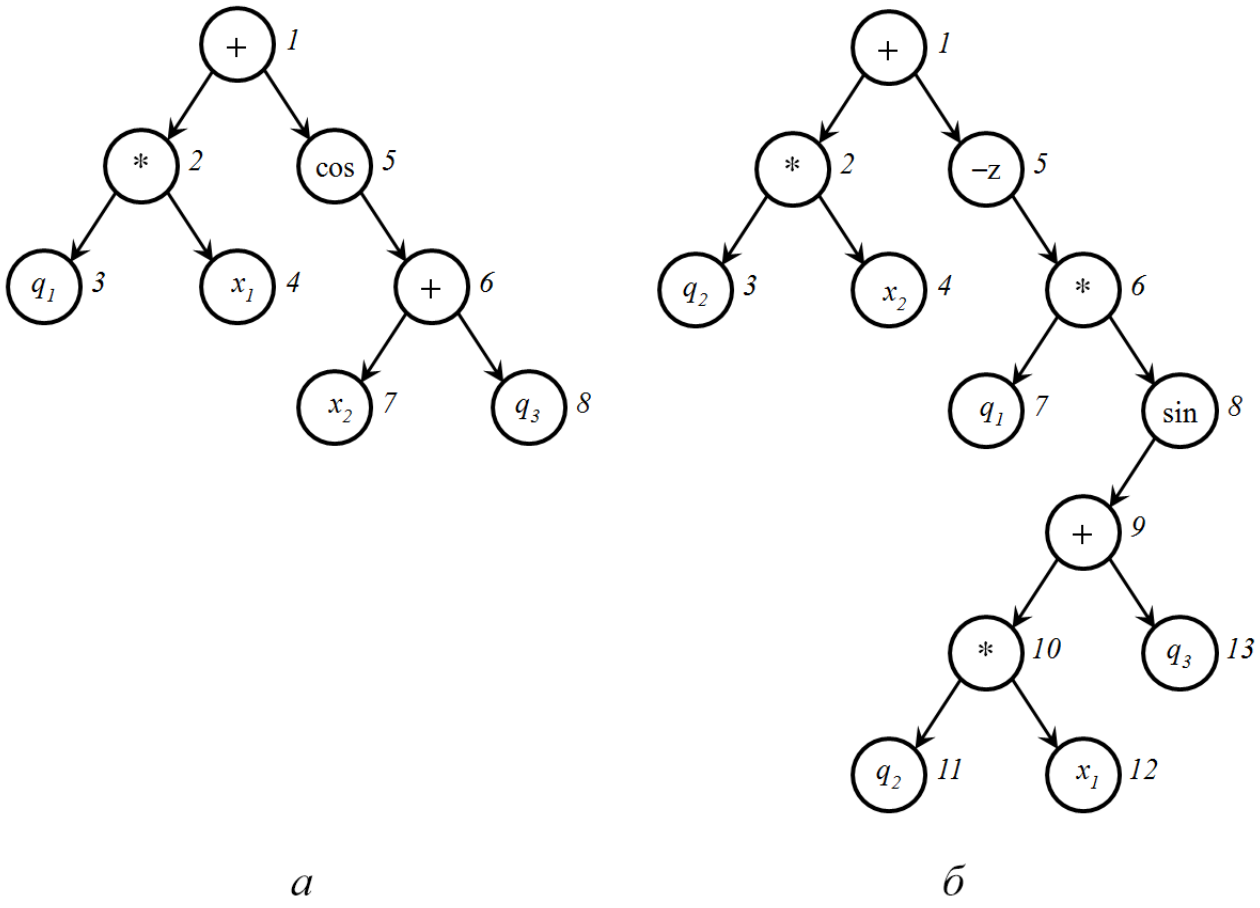


Рисунок 2.2 — Результат выполнения операции скрещивания: *а* — выражение (2.3); *б* — выражение (2.4).

Точкой скрещивания для выражения (2.1) стал узел 7, а для выражения (2.2) — узел 2. В результате применения операции скрещивания были получены следующие математические выражения:

$$f_1'(\mathbf{x}) = q_1 x_1 + \cos(x_2 + q_3), \tag{2.3}$$

$$f_2'(\mathbf{x}) = q_2 x_2 - q_1 \sin(q_2 x_1 + q_3). \tag{2.4}$$

Далее путем вычисления значения определенного задачей критерия качества для каждого нового выражения можно оценить успешность применения генетических операций, отобрать лучшие решения и продолжить описанные вычисления до достижения заданного минимального значения критерия качества.

Для представления математического выражения в памяти компьютера удобно использовать двухкомпонентные целочисленные векторы. Такой вектор описывает кодируемую элементарную функцию. При этом первая компонента вектора определяет количество аргументов кодируемой функции, а вторая компонента указывает на номер этой функции во множестве всех используемых при поиске функций

$$\mathrm{F} = \{\mathrm{F}_0, \ldots, \mathrm{F}_n\},$$

где

$$\mathrm{F}_i = \left(f_{i,1}\left(z_1, \ldots, z_i\right), \ldots, f_{i,m_i}\left(z_1, \ldots, z_i\right)\right), \;\; i = \overline{0,n},$$

$f_{i,j}\left(z_1, \ldots, z_i\right)$ — функция под номером j с количеством аргументов i, $i = \overline{0,n}$. Код функции имеет вид

$$\mathbf{s} = \begin{bmatrix} s_1 & s_2 \end{bmatrix}^T,$$

где s_1 — количество аргументов функции, а s_2 — номер элементарной функции во множестве F_{s_1}.

Математическое выражение в закодированном виде будет представлять собой упорядоченное множество векторов кодов функций.

В качестве примера разберем представление в закодированном виде выражения (2.1). Для этого рассмотрим следующие множества функций:

$$\begin{gathered} \mathrm{F}_0 = (q_1,\, q_2,\, q_3,\, x_1,\, x_2), \\ \mathrm{F}_1 = (-z,\, \cos(z),\, \sin(z)), \\ \mathrm{F}_2 = (z_1 + z_2,\, z_1 z_2). \end{gathered} \tag{2.5}$$

Множества функций (2.5), записанные в кодах, будут иметь вид

$$\begin{gathered} \mathrm{F}_0 = \left(\begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 3 \end{bmatrix}, \begin{bmatrix} 0 \\ 4 \end{bmatrix}, \begin{bmatrix} 0 \\ 5 \end{bmatrix}\right), \\ \mathrm{F}_1 = \left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ 3 \end{bmatrix}\right), \\ \mathrm{F}_2 = \left(\begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix}\right). \end{gathered} \tag{2.6}$$

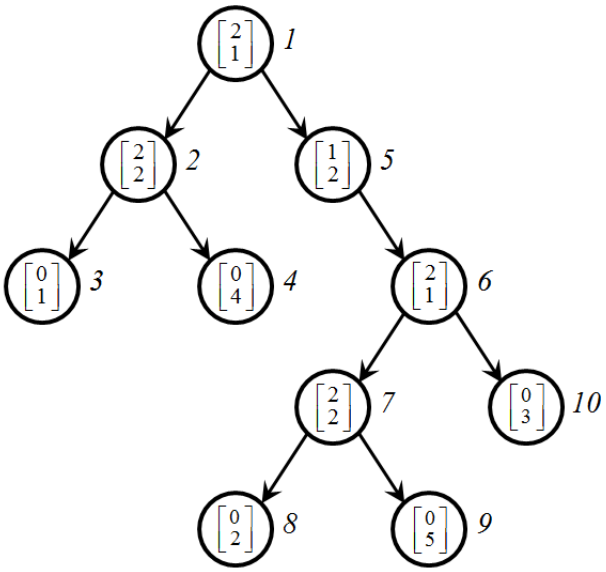


Рисунок 2.3 — Граф вычислений математического выражения (2.1) в кодах функций.

Заменим символьное представление функции в графе на Рис. 2.1(а) соответствующими векторами кода функции (2.6). Граф вычислений в кодах функций представлен на Рис. 2.3.

Из представленного на Рис. 2.3 графа составим закодированное представление математического выражения (2.1) на основе следующих правил:

- Вектор кода первого аргумента функции следует сразу после вектора кода функции;
- Вектора кодов второго и последующих аргументов функции следуют после вектора кода функции с нулевым количеством аргументов;
- Каждый вектор кода аргумента относится к ближайшей слева функции с недостающим числом аргументов.

В итоге получаем следующую запись математического выражения (2.1) в закодированном виде:

$$S_1 = \left(\begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 4 \end{bmatrix}, \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 5 \end{bmatrix}, \begin{bmatrix} 0 \\ 3 \end{bmatrix} \right).$$

Аналогичные действия, выполненные для математического выражения (2.2), позволяют получить следующую закодированную строку:

$$S_2 = \left(\begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 5 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 4 \end{bmatrix}, \begin{bmatrix} 0 \\ 3 \end{bmatrix} \right).$$

Для обеспечения корректности записи математического выражения в закодированном виде необходимо ввести понятие индекса символа математического выражения. Индекс j-го символа указывает на минимальное число недостающих символов справа от него и вычисляется по формуле

$$T\left(j\right) = 1 - j + \sum_{i=1}^{j} s_1^j.$$

Математическое выражение записано в закодированном виде корректно, если выполняются следующие условия

$$T\left(j\right) > 0,\, j = \overline{1,K-1}, \tag{2.7}$$

$$T\left(K\right) = 0. \tag{2.8}$$

где K — количество символов в кодируемом математическом выражении.

Введем понятие записи кода подвыражения математического выражения

$$\mathrm{S}\left(m\right) = \left(\mathbf{s}^m, \ldots, \mathbf{s}^{m+k}\right) \subseteq \mathrm{S},$$

где m — позиция кода символа $\mathbf{s}^m$, с которого начинается подвыражение. На основе использования подвыражений в методе генетического программирования осуществляется процедура скрещивания. Так как любая запись кода подвыражения математического выражения сама является записью математического выражения, для её корректности также необходимо выполнение условий (2.7) и (2.8).

В методе генетического программирования для выполнения процедуры скрещивания двух отобранных решений необходимо выбрать две точки скрещивания. Операция скрещивания заключается в обмене подвыражений у пары выбранных выражений. В рассмотренном выше примере точкой скрещивания для выражения (2.1) был символ 7, а для выражения (2.2) — символ 2. Обеспечив выполнение условий (2.7) и (2.8), определим подвыражения, коды которых начинаются в данных точках

$$\mathrm{S}_1\left(7\right) = \left(\begin{bmatrix}2\\2\end{bmatrix}, \begin{bmatrix}0\\2\end{bmatrix}, \begin{bmatrix}0\\5\end{bmatrix}\right),$$

$$\mathrm{S}_2\left(2\right) = \left(\begin{bmatrix}0\\5\end{bmatrix}\right).$$

После выполнения операции скрещивания получаем новые выражения:

$$S_1' = \left(\begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 4 \end{bmatrix}, \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 5 \end{bmatrix}, \begin{bmatrix} 0 \\ 3 \end{bmatrix} \right), \tag{2.9}$$

$$S_2' = \left(\begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 5 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 0 \\ 4 \end{bmatrix}, \begin{bmatrix} 0 \\ 3 \end{bmatrix} \right). \tag{2.10}$$

Математические выражения в закодированном виде (2.9) и (2.10) соответствуют математическим выражениям (2.3) и (2.4).

Особенностью метода генетического программирования является нефиксированная длина записи математических выражений и их подвыражений в закодированном виде. Данную особенность относят к недостаткам метода, так как это существенно осложняет вычисления с помощью данного метода. Также при поиске подвыражений необходимо учитывать их корректность в соответствии с условиями (2.7) и (2.8).

2.2.2 Метод декартового генетического программирования

Описанные в предыдущем разделе недостатки метода генетического программирования послужили толчком к созданию более простых и эффективных методов символьной регрессии. Так в 1999 году был предложен метод декартового генетического программирования [124; 125]. Преимущество данного метода в фиксированной длине кода математических выражений и в использовании отдельного целочисленного вектора для описания вызова каждой функции. Это позволяет значительно упростить алгоритмическую сложность метода.

По сути, декартово генетическое программирование является усовершенствованной модификацией метода генетического программирования. Своё название данный метод получил в виду того, что закодированное выражение в данном методе представляет собой двухмерную сетку вычислительных узлов [123].

Кодирование математического выражения в методе декартового генетического программирования осуществляется путем использования целочисленного вектора, описывающего коды функции, коды её аргументов, а также номера используемых параметров и переменной, куда должен быть записан результат

вычисления данной функции. Таким образом, элементарный код математического выражения представляет собой вектор из $M+1$ компонент, где M — это максимальное число аргументов используемых функций

$$\mathbf{d} = \left[\begin{array}{ccc} d_1 & \ldots & d_{M+1} \end{array} \right]^T. \tag{2.11}$$

В отличие от метода генетического программирования, в методе декартового генетического программирования все элементарные базовые функции вне зависимости от числа аргументов записываются в одно объединенное множество

$$\begin{aligned} \mathrm{F} = \Bigg\{ & \underbrace{f_1(z), \ldots, f_{n_1}(z)}_{n_1}, \ \underbrace{f_{n_1+1}(z_1,z_2), \ldots, f_{n_1+n_2}(z_1,z_2)}_{n_2}, \ldots, \\ & \underbrace{f_{\sum_{i=1}^{M-1} n_i+1}(z_1,\ldots,z_M), \ldots, f_{\sum_{i=1}^{M-1} n_i+n_M}(z_1,\ldots,z_M)}_{n_M} \Bigg\}. \end{aligned} \tag{2.12}$$

где n_i — количество функций от i аргументов, $i = \overline{1,M}$.

Все параметры и переменные, а также дополнительные переменные для хранения результатов промежуточных вычислений также записываются в одно множество

$$\begin{aligned} \mathrm{R} = \Bigg\{ & \underbrace{r_1 = q_1, \ldots, r_p = q_p}_{p}, \ \underbrace{r_{p+1} = x_1, \ldots, r_{p+n} = x_n}_{n}, \\ & \underbrace{r_{p+n+1}, \ldots, r_{p+n+l}}_{l} \Bigg\}. \end{aligned} \tag{2.13}$$

где p — число параметров, n — число переменных, l — число дополнительных переменных.

В элементарном коде математического выражения (2.11) d_1 — это номер функции из множества (2.12), $d_1 \in \left\{1, \sum_{i=1}^{M} n_i\right\}$, d_j — это номер элемента из множества (2.13), $j = \overline{2,\, M+1}$. Если функция с номером d_1 имеет $m < M$ аргументов, то оставшиеся аргументы в векторе (2.11) не используются, $d_j = 0$, $j = \overline{m+2,\, M+1}$.

Таким образом, элементарный код математического выражения (2.11) позволяет определить вызов функции с соответствующими аргументами

$$f_{d_1} = (r_{d_2}, \ldots, r_{d_m}).$$

Закодированное математическое выражение представляет собой набор векторов (2.11)

$$\mathrm{D} = \left(\mathbf{d}^1, \ldots, \mathbf{d}^l\right). \tag{2.14}$$

При поиске оптимального математического выражения методом декартового генетического программирования генерируется множество возможных решений заданного размера H, каждый элемент которого представляет собой возможное решение в закодированном виде (2.14). В процессе поиска для отобранных возможных решений выполняются генетические операции скрещивания и мутации. Для выполнения этих операций не требуется, чтобы все закодированные математические выражения имели одинаковую длину кода. Такое свойство делает соответствующие операции проще и ставится авторами декартово генетического программирования как преимущество их метода [145].

Изначально определить число вызовов функций в искомом математическом выражении не представляется возможным. Поэтому, для обеспечения одинаковой длины кода математического выражения, количество вызовов функций, а, следовательно, и число дополнительных переменных для хранения промежуточных результатов l берется с избытком. При этом часть дополнительных переменных с результатами выполнения избыточных вызовов функций может нигде не использоваться.

Для выполнения операции скрещивания для двух отобранных наборов векторов

$$\mathrm{D}_i = \left(\mathbf{d}^{i,1}, \ldots, \mathbf{d}^{i,l}\right),$$
$$\mathrm{D}_j = \left(\mathbf{d}^{j,1}, \ldots, \mathbf{d}^{j,l}\right)$$

случайным образом определяется точка скрещивания k, $k \in \{1, \ldots, l\}$, $i, j \in \{1, \ldots, H\}$, $i \neq j$, H — размер множества возможных решений. Правые части наборов векторов, начиная от точки скрещивания, обмениваются местами, в результате чего получаются два новых возможных решения

$$\tilde{\mathrm{D}}_i = \left(\mathbf{d}^{i,1}, \ldots, \mathbf{d}^{i,k-1}, \mathbf{d}^{j,k}, \ldots, \mathbf{d}^{j,l}\right),$$

$$\tilde{\mathrm{D}}_j = \left(\mathbf{d}^{j,1}, \ldots, \mathbf{d}^{j,k-1}, \mathbf{d}^{i,k}, \ldots, \mathbf{d}^{i,l}\right).$$

Для выполнения операции мутации в отобранном возможном решении случайным образом определяется точка мутации α, $\alpha \in \{1, \ldots, l\}$. Вектор кода

элементарного математического выражения $\mathbf{d}^{\alpha}$ на позиции α отобранного возможного решения заменяется на новый вектор $\tilde{\mathbf{d}}^{\alpha}$, сгенерированный случайным образом.

Далее для полученных в результате применения операций скрещивания и мутации новых возможных решений вычисляется значение критерия качества и оценивается успешность поиска. Если заданный критерий окончания поиска не достигнут, то производится отбор лучших возможных решений, для которых продолжаются описанные выше вычисления.

В качестве примера рассмотрим принципы кодирования и поиска нового выражения методом декартового генетического программирования для математических выражений (2.1) и (2.2). Множество элементарных базовых функций (2.12) для данных выражений будет иметь вид

$$\begin{aligned} \mathrm{F} = \ \{&f_1(z) = -z,\ f_2(z) = \cos(z),\ f_3(z) = \sin(z),\\ &f_4(z_1,z_2) = z_1 + z_2,\ f_5(z_1,z_2) = z_1 z_2\}. \end{aligned} \tag{2.15}$$

Для описания математического выражения используются функции с максимальным числом $M = 2$ аргументов. Таким образом, вектор элементарного кода математического выражения будет состоять из $M + 1 = 3$ компонент.

Пусть искомое математическое выражение состоит из максимум $l = 7$ элементарных базовых функций. Тогда набор векторов (2.14) каждого возможного решения будет состоять из $l = 7$ элементов, а множество параметров и переменных (2.13) будет иметь вид:

$$\begin{aligned} \mathrm{R} = \ \{&r_1 = q_1,\ r_2 = q_2,\ r_3 = q_3,\ r_4 = x_1,\ r_5 = x_2,\\ &r_6 = \mathbf{d}_1,\ r_7 = \mathbf{d}_2,\ r_8 = \mathbf{d}_3,\ r_9 = \mathbf{d}_4,\ r_{10} = \mathbf{d}_5,\ r_{11} = \mathbf{d}_6,\ r_{12} = \mathbf{d}_7\}. \end{aligned} \tag{2.16}$$

С учетом (2.15) и (2.16) набор векторов (2.14) для математических выражений (2.1) и (2.2) будет иметь вид соответственно

$$\begin{aligned} \mathrm{D}_1 = \ \Big(&\mathbf{d}^1 = \begin{bmatrix} 5 & 2 & 5 \end{bmatrix}^T,\ \mathbf{d}^2 = \begin{bmatrix} 4 & 6 & 3 \end{bmatrix}^T,\ \mathbf{d}^3 = \begin{bmatrix} 2 & 7 & 0 \end{bmatrix}^T,\\ &\mathbf{d}^4 = \begin{bmatrix} 5 & 3 & 5 \end{bmatrix}^T,\ \mathbf{d}^5 = \begin{bmatrix} 5 & 1 & 4 \end{bmatrix}^T,\ \mathbf{d}^6 = \begin{bmatrix} 3 & 3 & 0 \end{bmatrix}^T,\\ &\mathbf{d}^7 = \begin{bmatrix} 4 & 8 & 10 \end{bmatrix}^T\Big), \end{aligned} \tag{2.17}$$

$$\mathrm{D}_2 = \Big(\mathbf{d}^1 = \begin{bmatrix}5 & 2 & 4\end{bmatrix}^T, \mathbf{d}^2 = \begin{bmatrix}4 & 6 & 3\end{bmatrix}^T, \mathbf{d}^3 = \begin{bmatrix}3 & 7 & 0\end{bmatrix}^T,$$
$$\mathbf{d}^4 = \begin{bmatrix}5 & 8 & 1\end{bmatrix}^T, \mathbf{d}^5 = \begin{bmatrix}1 & 9 & 0\end{bmatrix}^T, \mathbf{d}^6 = \begin{bmatrix}4 & 1 & 2\end{bmatrix}^T, \quad (2.18)$$
$$\mathbf{d}^7 = \begin{bmatrix}4 & 10 & 5\end{bmatrix}^T\Big).$$

Результаты выполнения элементарных базовых функций, описываемых векторами $\mathbf{d}^4 = \begin{bmatrix}5 & 3 & 5\end{bmatrix}^T$, $\mathbf{d}^6 = \begin{bmatrix}3 & 3 & 0\end{bmatrix}^T$ набора D_1 (2.17) и $\mathbf{d}^6 = \begin{bmatrix}4 & 1 & 2\end{bmatrix}^T$ набора D_2 (2.18) не используются в дальнейших вычислениях математического выражения. Значение их компонент не влияет на итоговое математическое выражение, а их наличие в наборах D_1 и D_2 обусловлено необходимостью выполнения условия одинаковой длины кода математического выражения. Однако данные векторы могут участвовать в операциях скрещивания и мутации. В векторах наборов D_1 и D_2, в которых выполняется базовая функция от одного аргумента в качестве компоненты d_3, ассоциирующейся со вторым аргументом, записано значение 0, $d_3 = 0$.

Случайным образом выберем точку скрещивания k. Пусть $k = 4$, тогда после обмена части наборов векторов правее точки скрещивания получим следующие новые возможные решения

$$\tilde{\mathrm{D}}_1 = \Big(\mathbf{d}^1 = \begin{bmatrix}5 & 2 & 5\end{bmatrix}^T, \mathbf{d}^2 = \begin{bmatrix}4 & 6 & 3\end{bmatrix}^T, \mathbf{d}^3 = \begin{bmatrix}2 & 7 & 0\end{bmatrix}^T,$$
$$\mathbf{d}^4 = \begin{bmatrix}5 & 8 & 1\end{bmatrix}^T, \mathbf{d}^5 = \begin{bmatrix}1 & 9 & 0\end{bmatrix}^T, \mathbf{d}^6 = \begin{bmatrix}4 & 1 & 2\end{bmatrix}^T, \quad (2.19)$$
$$\mathbf{d}^7 = \begin{bmatrix}4 & 10 & 5\end{bmatrix}^T\Big),$$

$$\tilde{\mathrm{D}}_2 = \Big(\mathbf{d}^1 = \begin{bmatrix}5 & 2 & 4\end{bmatrix}^T, \mathbf{d}^2 = \begin{bmatrix}4 & 6 & 3\end{bmatrix}^T, \mathbf{d}^3 = \begin{bmatrix}3 & 7 & 0\end{bmatrix}^T,$$
$$\mathbf{d}^4 = \begin{bmatrix}5 & 3 & 5\end{bmatrix}^T, \mathbf{d}^5 = \begin{bmatrix}5 & 1 & 4\end{bmatrix}^T, \mathbf{d}^6 = \begin{bmatrix}3 & 3 & 0\end{bmatrix}^T, \quad (2.20)$$
$$\mathbf{d}^7 = \begin{bmatrix}4 & 8 & 10\end{bmatrix}^T\Big).$$

Полученные в результате операции скрещивания закодированные выражения (2.19) и (2.20) соответствуют следующим математическим выражениям соответственно

$$f_1' = x_2 - q_1\cos(q_2x_2 + q_3),$$
$$f_2' = q_1x_1 + \sin(q_2x_1 + q_3).$$

Для выполнения операции мутации выберем выражение (2.19) и случайным образом определим точку мутации α. Пусть $\alpha = 3$, тогда заменим вектор $\mathbf{d}^3 = \begin{bmatrix} 2 & 7 & 0 \end{bmatrix}^T$ в наборе $\tilde{\mathrm{D}}_1$ на другой вектор, сгенерированный случайно, например $\tilde{\mathbf{d}}^3 = \begin{bmatrix} 5 & 7 & 4 \end{bmatrix}^T$. В итоге получим закодированное выражение

$$\begin{aligned} \tilde{\mathrm{D}}_1 = \Big(& \mathbf{d}^1 = \begin{bmatrix} 5 & 2 & 5 \end{bmatrix}^T, \mathbf{d}^2 = \begin{bmatrix} 4 & 6 & 3 \end{bmatrix}^T, \tilde{\mathbf{d}}^3 = \begin{bmatrix} 5 & 7 & 2 \end{bmatrix}^T, \\ & \mathbf{d}^4 = \begin{bmatrix} 5 & 8 & 1 \end{bmatrix}^T, \mathbf{d}^5 = \begin{bmatrix} 1 & 9 & 0 \end{bmatrix}^T, \mathbf{d}^6 = \begin{bmatrix} 4 & 1 & 2 \end{bmatrix}^T, \\ & \mathbf{d}^7 = \begin{bmatrix} 4 & 10 & 5 \end{bmatrix}^T \Big), \end{aligned}$$

которому соответствует следующее математическое выражение

$$f_1' = x_2 - q_1 x_1 (q_2 x_2 + q_3).$$

Программная реализация алгоритма метода декартового генетического программирования значительно проще реализации алгоритма генетического программирования. Достигается это за счет осуществления поиска оптимальной структуры математического выражения на наборах кодов одинаковой длины. Реализация процедуры скрещивания для двух наборов кодов одинаковой длины также значительно проще. Она не требует дополнительных проверок корректности полученных новых возможных решений.

2.2.3 Метод грамматической эволюции

Метод грамматической эволюции был представлен в 1998 году коллективом авторов во главе с Майклом О'Нейлом [130; 136]. По словам авторов, данный метод может использоваться для широкого класса задач символьной регрессии, в том числе и для задач автоматического написания программного кода на любом языке программирования [131].

Символьная запись выражения в методе грамматической эволюции приводится в виде системы продукционных правил в форме Бэкуса-Наура. Кроме множества продукционных правил также задается множество терминальных символов, множество нетерминальных символов и множество стартовых символов. Для кодирования символьного выражения используется упорядоченное

множество целых чисел — хромосом

$$\mathrm{C} = (c_1, \ldots, c_K)\,. \tag{2.21}$$

Для записи хромосом, как правило, используется двоичное представление числа, однако запись в десятичной системе счисления также возможна. Для обеспечения одинаковой длины кода у различных возможных решений длина каждой хромосомы и их количество в коде выражения фиксируется. При этом для поиска более сложных выражений число хромосом можно взять с избытком. Каждая хромосома указывает на номер элемента из соответствующего заданного множества продукционных правил.

Поиск оптимального выражения осуществляется с помощью генетического алгоритма. Операция скрещивания осуществляется для двух отобранных возможных решений путем определения случайной точки скрещивания и обмена части решений правее данной точки. Операция скрещивания не приводит к изменению длины кода выражения, для её выполнения не требуется выполнение каких-либо дополнительных условий, а после её выполнения не требуется проверка корректности получившегося выражения.

После операции скрещивания отобранных решений производится оценка качества поиска посредством вычисления значения критерия качества для новых возможных решений, полученных в результате применения генетических операций. Если заданный критерий окончания поиска не достигнут, то производится отбор лучших возможных решений, для которых продолжаются описанные выше вычисления.

В качестве примера рассмотрим принципы кодирования и поиска нового выражения методом грамматической эволюции для математических выражений (2.1) и (2.2). Закодированное символьное выражение для каждого математического выражения будет иметь вид множества хромосом (2.21). Возьмем число хромосом $K = 20$. Для описания математических выражений в форме Бэкуса-Наура необходимо определить множества символов. Представим их в виде таблицы продукционных правил (Таблица 1).

Стартовый символ для математического выражения имеет вид

$$f_1(\mathbf{x}) = \text{<expr>}.$$

Далее каждая хромосома в коде выражения будет указывать на номер варианта из соответствующей строки таблицы продукционных правил (Таблица

Таблица 1 — Таблица продукционных правил

Правило	№ варианта	Вариант
<expr>	0	<expr><op><expr>
	1	<function>(<expr>)
	2	<var>
<op>	0	+
	1	−
	2	∗
<function>	0	cos
	1	sin
<var>	0	q_1
	1	q_2
	2	q_3
	3	x_1
	4	x_2

1). Вычисление кода математического выражения (2.1) будет состоять из следующих шагов:

$c_1 = 0 \Rightarrow f_1(\mathbf{x}) =$<expr><op><expr>

$c_2 = 0 \Rightarrow f_1(\mathbf{x}) =$<expr><op><expr><op><expr>

$c_3 = 2 \Rightarrow f_1(\mathbf{x}) =$<var><op><expr><op><expr>

$c_4 = 0 \Rightarrow f_1(\mathbf{x}) = q_1$ <op><expr><op><expr>

$c_5 = 2 \Rightarrow f_1(\mathbf{x}) = q_1*$ <expr><op><expr>

$c_6 = 2 \Rightarrow f_1(\mathbf{x}) = q_1*$ <var><op><expr>

$c_7 = 3 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1$ <op><expr>

$c_8 = 0 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1+$ <expr>

$c_9 = 1 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1+$ <function> (<expr>)

$c_{10} = 0 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1 + \cos($<expr>$)$

$c_{11} = 0 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1 + \cos($<expr><op><expr>$)$

$c_{12} = 0 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1 + \cos($<expr><op><expr><op><expr>$)$

$c_{13} = 2 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1 + \cos($<var><op><expr><op><expr>$)$

$c_{14} = 1 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1 + \cos(q_2$ <op><expr><op><expr>$)$

$c_{15} = 2 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1 + \cos(q_2 * \text{<expr><op><expr>})$
$c_{16} = 2 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1 + \cos(q_2 * \text{<var><op><expr>})$
$c_{17} = 4 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1 + \cos(q_2 * x_2 \text{ <op><expr>})$
$c_{18} = 0 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1 + \cos(q_2 * x_2 + \text{ <expr>})$
$c_{19} = 2 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1 + \cos(q_2 * x_2 + \text{ <var>})$
$c_{20} = 2 \Rightarrow f_1(\mathbf{x}) = q_1 * x_1 + \cos(q_2 * x_2 + q_3)$

Таким образом, код математического выражения (2.1) будет иметь вид

$$\mathrm{C}_1 = (0, 0, 2, 0, 2, 2, 3, 0, 1, 0, 0, 0, 2, 1, 2, 2, 4, 0, 2, 2)\,.$$

Вычисленный аналогичным образом код математического выражения (2.2) будет иметь вид

$$\mathrm{C}_2 = (0, 2, 4, 1, 0, 2, 0, 2, 1, 1, 0, 0, 2, 1, 2, 2, 3, 0, 2, 2)\,.$$

Случайным образом выберем точку скрещивания k. Пусть $k = 8$, производим обмен части наборов векторов правее точки скрещивания. В результате операции скрещивания получаем следующие новые возможные решения

$$\mathrm{C}'_1 = (0, 0, 2, 0, 2, 2, 3, 2, 1, 1, 0, 0, 2, 1, 2, 2, 3, 0, 2, 2)\,, \qquad (2.22)$$

$$\mathrm{C}'_2 = (0, 2, 4, 1, 0, 2, 0, 0, 1, 0, 0, 0, 2, 1, 2, 2, 4, 0, 2, 2)\,. \qquad (2.23)$$

Используя таблицу продукционных правил (Таблица 1), получим математический вид закодированного выражения (2.22)
$c_1 = 0 \Rightarrow f'_1(\mathbf{x}) = \text{<expr><op><expr>}$
$c_2 = 0 \Rightarrow f'_1(\mathbf{x}) = \text{<expr><op><expr><op><expr>}$
$c_3 = 2 \Rightarrow f'_1(\mathbf{x}) = \text{<var><op><expr><op><expr>}$
$c_4 = 0 \Rightarrow f'_1(\mathbf{x}) = q_1 \text{ <op><expr><op><expr>}$
$c_5 = 2 \Rightarrow f'_1(\mathbf{x}) = q_1 * \text{ <expr><op><expr>}$
$c_6 = 2 \Rightarrow f'_1(\mathbf{x}) = q_1 * \text{ <var><op><expr>}$
$c_7 = 3 \Rightarrow f'_1(\mathbf{x}) = q_1 * x_1 \text{ <op><expr>}$
$c_8 = 2 \Rightarrow f'_1(\mathbf{x}) = q_1 * x_1 * \text{ <expr>}$
$c_9 = 1 \Rightarrow f'_1(\mathbf{x}) = q_1 * x_1 * \text{ <function> (<expr>)}$
$c_{10} = 1 \Rightarrow f'_1(\mathbf{x}) = q_1 * x_1 * \sin(\text{<expr>})$
$c_{11} = 0 \Rightarrow f'_1(\mathbf{x}) = q_1 * x_1 * \sin(\text{<expr><op><expr>})$
$c_{12} = 0 \Rightarrow f'_1(\mathbf{x}) = q_1 * x_1 * \sin(\text{<expr><op><expr><op><expr>})$
$c_{13} = 2 \Rightarrow f'_1(\mathbf{x}) = q_1 * x_1 * \sin(\text{<var><op><expr><op><expr>})$

$c_{14} = 1 \Rightarrow f_1'(\mathbf{x}) = q_1 * x_1 * \sin(q_2$ <op><expr><op><expr>)
$c_{15} = 2 \Rightarrow f_1'(\mathbf{x}) = q_1 * x_1 * \sin(q_2*$ <expr><op><expr>)
$c_{16} = 2 \Rightarrow f_1'(\mathbf{x}) = q_1 * x_1 * \sin(q_2*$ <var><op><expr>)
$c_{17} = 3 \Rightarrow f_1'(\mathbf{x}) = q_1 * x_1 * \sin(q_2 * x_1$ <op><expr>)
$c_{18} = 0 \Rightarrow f_1'(\mathbf{x}) = q_1 * x_1 * \sin(q_2 * x_1+$ <expr>)
$c_{19} = 2 \Rightarrow f_1'(\mathbf{x}) = q_1 * x_1 * \sin(q_2 * x_1+$ <var>)
$c_{20} = 2 \Rightarrow f_1'(\mathbf{x}) = q_1 * x_1 * \sin(q_2 * x_1 + q_3)$

Таким образом, коду выражения (2.22) соответствует математическое выражение

$$f_1'(\mathbf{x}) = q_1 x_1 \sin(q_2 x_1 + q_3).$$

Аналогичным образом для закодированного выражения (2.23) получаем математическое выражение

$$f_2'(\mathbf{x}) = x_2 - q_1 + \cos(q_2 x_2 + q_3).$$

При сравнении с методом генетического программирования метод грамматической эволюции обладает явными преимуществами. Основными преимуществами можно назвать одинаковую длину кода всех возможных решений и кодирование символьного представления выражения с помощью целых десятичных или двоичных чисел. Также к преимуществам метода можно отнести более простую реализацию операции скрещивания и отсутствие необходимости проверки новых решений на корректность. Среди недостатков метода грамматической эволюции, как и других рассмотренных выше методов, следует отметить то, что операции скрещивания и мутации могут приводить к существенным изменениям символьного выражения. Это обусловлено тем, что близкие по значению хромосомы не всегда соответствуют близким по значению символам.

2.2.4 Принцип малых вариаций базисного решения

При решении задачи синтеза системы управления с помощью методов символьной регрессии искомая функция управления будет представлять собой оптимальное выражение в виде набора символов или их кодов. Поиск оптимального решения на пространстве нечисловых символов или кодов осложняется

отсутствием связи между близостью элементов пространства поиска и близостью их числовых оценок по заданному критерию качества. В первую очередь отсутствие евклидовой метрики оценки расстояния между возможными решениями существенно усложняет процедуру целенаправленного поиска, особенно в области уже найденного возможного решения. Введение такой метрики для элементов пространства символов или кодов, вообще говоря, невозможно.

Для преодоления указанного недостатка профессором А.И. Дивеевым была предложена модификация алгоритма поиска на пространстве нечисловых символов, основанная на принципе малых вариаций базисного решения [94]. В основе предложенного метода лежит понятие малой вариации, которая изменяет код текущего возможного решения, тем самым приводя к коду нового возможного решения. При этом принцип малых вариаций не меняет методику кодирования и декодирования соответствующего метода символьной регрессии. Однако вместо множества возможных решений берется всего одно возможное решение, которое называется базисным. Далее строится множество наборов малых вариаций. Каждый набор из множества позволяет получить из базисного решения новое возможное решение. Поиск оптимального решения осуществляется на наборах малых вариаций. Предусмотренные соответствующим методом символьной регрессии генетические операции скрещивания и мутации применяются к наборам малых вариаций, в результате чего получаются новые наборы с измененным порядком и самими вариациями. Множество полученных в результате генетических операций новых наборов вариаций после применения к базисному решению дают множество возможных решений. Некоторые из этих возможных решений могут оказаться лучше, чем базисное решение по оценке критерия качества. С определенной периодичностью следует заменять базисное решение. В качестве нового базисного решения берется лучшее на текущий момент поиска возможное решение. При использовании принципа малых вариаций выбор начального базисного решения, основанный на предварительном анализе решаемой задачи, может оказывать положительное влияние на скорость нахождения оптимального решения.

Применение принципа малых вариаций базисного решения позволяет определить численную метрику расстояния между двумя возможными решениями, а также определить понятие поиска в окрестности выбранного решения. Для пояснения введенных понятий следует рассмотреть поиск оптимального

решения с помощью принципа малых вариаций базисного решения более подробно.

В общем виде код символьного выражения можно представить в виде последовательности кодов символов заданной длины n

$$\mathbf{y} = (y_1, y_2, \ldots, y_n), \tag{2.24}$$

где $y_i \in \mathrm{A}$, $i = \overline{1,n}$, A — конечный набор кодов базовых символов

$$\mathrm{A} = \{0, a_1, a_2, \ldots, a_L\}, \tag{2.25}$$

нулевой символ 0 обозначает отсутствие элемента.

При построении кода выражения (2.24) из набора кодов базовых символов (2.25) определяется его допустимость на основе конечного числа функций оценки правильности кода выражения

$$\theta_j(\mathbf{y}) \leqslant 0, \;\; j = \overline{1,l}. \tag{2.26}$$

Из всех возможных кодов символьных выражений (2.24), удовлетворяющих требованиям (2.26), определяем множество допустимых кодов символьных выражений заданной длины n

$$\mathrm{Y} = \left\{\mathbf{y}^1, \mathbf{y}^2, \ldots\right\}. \tag{2.27}$$

Любые два символьных выражения из множества допустимых кодов (2.27) отличаются между собой кодом одного или более базовых символов.

Согласно принципу малых вариаций базисного решения, под элементарной вариацией кода символьного выражения понимается замена кода одного символа данного выражения на символ из множества базовых символов (2.25). При этом такая замена может привести к недопустимому коду выражения в соответствии с условиями (2.26). Для получения допустимого кода символьного выражения одной элементарной вариации может быть недостаточно.

Минимальный набор элементарных вариаций, позволяющий получить из одного допустимого решения новое решение, удовлетворяющее условиям (2.26), называется малой вариацией $\delta(\mathbf{y})$. Малая вариация может состоять из одной или нескольких элементарных вариаций.

Для множества допустимых кодов символьных выражений (2.27) можно определить конечное множество малых вариаций

$$\Omega(\mathrm{Y}) = \{\delta_1(\mathbf{y}), \ldots, \delta_M(\mathbf{y})\}, \tag{2.28}$$

обладающее свойством полноты, то есть допускающее возможность для любых двух кодов выражений из множества (2.27) найти конечное число малых вариаций, позволяющих из одного кода выражения получить второй код выражения

$$\forall \mathbf{y}^i,\mathbf{y}^j \in \mathrm{Y},\, \mathbf{y}^i = \delta_{k_1}(\ldots \delta_{k_d}(\mathbf{y}^j)),$$

где d — минимальное число малых вариаций для получения $\mathbf{y}^j$ из $\mathbf{y}^i$.

Минимальное число d малых вариаций, необходимых для получения из одного допустимого кода символьного выражения другого допустимого кода символьного выражения, называется расстоянием между двумя кодами символьных выражений

$$\forall \mathbf{y}^i,\mathbf{y}^j \in \mathrm{Y},\ \left\|\mathbf{y}^i - \mathbf{y}^j\right\|_\Omega = d, \text{ если } d = \min_r \left\{\mathbf{y}^i = \delta_{k_1}(\ldots \delta_{k_d}(\mathbf{y}^j))\right\}, \qquad (2.29)$$

где $\delta_{k_p}(\mathbf{y}) \in \Omega(\mathrm{Y}),\ p = \overline{1,r}$.

Окрестностью $\Delta(\mathbf{y})$ кода символьного выражения $\mathbf{y}$ называется подмножество кодов символьных выражений, расположенных на расстоянии Δ от кода символьного выражения $\mathbf{y}$

$$\forall \mathbf{y}^i \in \Delta(\mathbf{y}),\ \left\|\mathbf{y}^i - \mathbf{y}\right\|_\Omega = \Delta,\ \Delta(\mathbf{y}) \subseteq \mathrm{Y}.$$

Для организации целенаправленного поиска на множестве кодов символьных выражений с использованием принципа малых вариаций необходимо ввести понятие вектора вариаций $\mathbf{w}$, описывающего малую вариацию $\delta(\mathbf{y})$

$$\mathbf{w} = \begin{bmatrix} w_1 & w_2 & \ldots & w_r \end{bmatrix}^T, \qquad (2.30)$$

где w_1 — номер малой вариации во множестве (2.28), w_2 — номер изменяемого символа в коде символьного выражения (2.24), w_i, $i = \overline{3,r}$ — другая информация для выполнения малой вариации $\delta(\mathbf{y})$, если таковая требуется.

Для определения действия вектора вариаций на любое символьное выражение $\mathbf{y}$ из множества (2.27) используется обозначение

$$\mathbf{y}^i = \mathbf{w}^l \circ \mathbf{y}^j,$$

где $\mathbf{w}^l$ — вектор вариаций, описывающий малую вариацию $\delta_l(\mathbf{y})$, $\delta_l(\mathbf{y}^j) = \mathbf{y}^i$.

Получить код символьного выражения $\mathbf{y}^i$ из Δ-окрестности символьного выражения $\mathbf{y}^j$ можно путем применения не более Δ векторов вариаций

$$\forall \mathbf{y}^i \in \Delta\left(\mathbf{y}^j\right),\ \mathbf{y}^i = \mathbf{w}^k \circ \ldots \circ \mathbf{w}^1 \circ \mathbf{y}^j,\ k \leqslant \Delta.$$

Таким образом, чтобы получить любое символьное выражение $\mathbf{y}'$ в окрестности на заданном расстоянии d от символьного выражения $\mathbf{y}$, $\mathbf{y}' \in d(\mathbf{y})$, можно использовать набор векторов вариаций W из d векторов

$$\mathrm{W} = \left(\mathbf{w}^1, \ldots, \mathbf{w}^d\right). \tag{2.31}$$

При поиске оптимального символьного выражения методом символьной регрессии с использованием принципа малых вариаций выбирается одно начальное возможное решение $\mathbf{y}^0$, называемое базисным. Выбор базисного решения можно осуществлять на основе предварительного анализа решаемой задачи или случайным образом. Остальные коды возможных решений заменяются наборами малых вариаций базисного решения. Целенаправленный поиск с применением генетических операций скрещивания и мутации осуществляется на множестве наборов малых вариаций базисного решения. В терминологии генетического алгоритма такое множество называется популяцией.

Для упрощения процедуры применения генетических операций число векторов вариаций в каждом наборе популяции должно быть одинаковым. Если для получения нового символьного выражения $\mathbf{y}'$ в d-окрестности символьного выражения $\mathbf{y}$ требуется $k < d$ малых вариаций, то для соблюдения количества векторов вариаций в наборе необходимо добавить $l = d - k$ нулевых векторов вариации $\mathbf{w}^0$. Нулевой вектор вариации описывает невыполняемую малую вариацию

$$\forall \mathbf{y} \in \mathrm{Y},\ \mathbf{w}^0 \circ \mathbf{y} = \mathbf{y}.$$

Для организации поиска с использованием принципа малых вариаций задается начальное базисное решение $\mathbf{y}^0$ и генерируется популяция наборов векторов вариаций

$$\mathrm{W}^i = \left(\mathbf{w}^{i,1}, \ldots, \mathbf{w}^{i,d}\right),\ i = \overline{1,H}, \tag{2.32}$$

где d — заданное количество векторов вариаций в одном наборе, H — заданное значение размера популяции.

Популяция (2.32) дополняется набором из нулевых векторов вариаций

$$\mathrm{W}^0 = \left(\mathbf{w}^{0,1}, \ldots, \mathbf{w}^{0,d}\right).$$

Применение любого набора W^i, $i = \overline{0,H}$ из популяции к базисному решению $\mathbf{y}^0$ позволяет получить новое возможное решение $\mathbf{y}^i$ в d-окрестности

базисного решения $\mathbf{y}^0$

$$\mathrm{W}^i \circ \mathbf{y}^0 = \mathbf{y}^i,\ i = \overline{0,H},$$

где $\mathbf{y}^i \in d\left(\mathbf{y}^0\right) \subseteq \mathrm{Y}$.

Оценка нового возможного решения $\mathbf{y}^i$ является одновременно и оценкой соответствующего набора векторов вариаций W^i, $i = \overline{0,H}$. Применение операций генетического алгоритма к наборам векторов вариаций W^i с учетом их оценок обеспечивает поиск наилучшего решения в d-окрестности базисного решения $\mathbf{y}^0$.

Для выполнения операции скрещивания по известным правилам генетических алгоритмов из популяции отбираются пары наборов векторов вариаций

$$\mathrm{W}^i = \left(\mathbf{w}^{i,1}, \ldots, \mathbf{w}^{i,d}\right), \tag{2.33}$$

$$\mathrm{W}^j = \left(\mathbf{w}^{j,1}, \ldots, \mathbf{w}^{j,d}\right), \tag{2.34}$$

где $i,j \in \{1, \ldots, H\}$, $i \neq j$.

Отобранным наборам векторов вариаций (2.33) и (2.34) соответствуют следующие возможные решения в символьном виде

$$\mathbf{y}^i = \mathrm{W}^i \circ \mathbf{y}^0,$$

$$\mathbf{y}^j = \mathrm{W}^j \circ \mathbf{y}^0.$$

Для отобранной пары наборов векторов вариаций случайным образом определяется точка скрещивания k, $k \in \{1, \ldots, d\}$. Правые части наборов векторов, начиная от точки скрещивания, обмениваются местами, в результате чего получаются два новых набора векторов вариаций

$$\tilde{\mathrm{W}}^i = \left(\mathbf{w}^{i,1}, \ldots, \mathbf{w}^{i,k-1}, \mathbf{w}^{j,k}, \ldots, \mathbf{w}^{j,d}\right), \tag{2.35}$$

$$\tilde{\mathrm{W}}^j = \left(\mathbf{w}^{j,1}, \ldots, \mathbf{w}^{j,k-1}, \mathbf{w}^{i,k}, \ldots, \mathbf{w}^{i,d}\right). \tag{2.36}$$

Для выполнения операции мутации в отобранном наборе векторов вариаций случайным образом определяется точка мутации α, $\alpha \in \{1, \ldots, d\}$. Вектор вариаций $\mathbf{w}^\alpha$ на позиции α отобранного набора векторов вариаций заменяется на новый вектор $\tilde{\mathbf{w}}^\alpha$, выбранный случайным образом из множества малых вариаций (2.28).

Наборам векторов вариаций (2.35) и (2.36), полученным в результате применения генетических операций, соответствуют следующие возможные решения в символьном виде

$$\tilde{\mathbf{y}}^i = \tilde{\mathrm{W}}^i \circ \mathbf{y}^0,$$

$$\tilde{\mathbf{y}}^j = \tilde{\mathrm{W}}^j \circ \mathbf{y}^0.$$

По оценкам новых возможных решений строится оценка соответствующих наборов векторов вариаций, с помощью которых они получены из базисного решения, а также определяется их включение в популяцию в соответствии с правилами генетического алгоритма.

После выполнения определенного количества поисковых итераций базисное решение целесообразно заменить на новое, выбранное из лучших по значению оценки наборов векторов вариаций. При обновлении базисного решения все наборы векторов вариаций генерируются заново. Поиск продолжается до достижения заданного критерия окончания или достижения максимального числа поисковых итераций.

Использование при поиске оптимальной структуры символьного выражения с помощью методов символьной регрессии принципа малых вариаций базисного решения обладает существенными преимуществами перед классическими методами, рассмотренными выше. Применение принципа малых вариаций не меняет методов кодирования и декодирования символьного выражения соответствующих методов, однако позволяет осуществлять поиск на кодах одинаковой длины даже в тех методах, классические реализации которых этого не предусматривают. Процедуры поиска методами, использующими принцип малых вариаций, эффективнее по времени поиска и использованию компьютерной памяти при сравнении с аналогичными методами, не использующими данный принцип [22; 107]. Главным преимуществом использования принципа малых вариаций базисного решения является то, что его применение позволяет задать евклидову метрику расстояния между двумя возможными решениями. Наличие такой метрики позволяет эффективно осуществлять целенаправленный поиск оптимального решения.

Принцип малых вариаций базисного решения послужил основой для создания эффективных модификаций некоторых методов символьной регрессии,

среди которых следует отметить метод вариационного генетического программирования [144], метод вариационного аналитического программирования [143], метод вариационной грамматической эволюции [22].

2.2.5 Метод сетевого оператора

Метод сетевого оператора был предложен профессором А.И. Дивеевым в 2008 году [21; 96]. В отличие от метода генетического программирования, который создавался в первую очередь для решения задач автоматического создания программ на языке LISP, метод сетевого оператора создавался для решения задач синтеза системы управления [22]. Таким образом, недостатки метода генетического программирования и некоторых других методов символьной регрессии были учтены и устранены в методе сетевого оператора ещё на этапе его создания.

Для кодирования символьного выражения методом сетевого оператора используется ограниченный набор функций с одним или двумя аргументами. Выражения с большим числом аргументов кодируются с помощью комбинации функций с двумя аргументами. Для этого функции с двумя аргументами должны обладать свойствами коммутативности и ассоциативности, а также иметь единичный элемент. Аргументами функций могут являться значения других функций или переменные и параметры символьного выражения. Из функций с одним и двумя аргументами и из всех переменных и параметров символьного выражения формируются три упорядоченных множества:

- множество переменных и параметров F_0;
- множество функций с одним аргументом F_1;
- множество функций с двумя аргументами F_2.

Символьное выражение в методе сетевого оператора представляется в виде ориентированного графа. Узлы-источники ориентированного графа ассоциируются с переменными и параметрами символьного выражения. Остальные узлы графа ассоциируются с функциями с двумя аргументами. Дуги графа ассоциируются с функциями с одним аргументом.

Для ориентированного графа без циклов возможно произвести нумерацию узлов в соответствии с условиями топологической сортировки. Далее строится

матрица смежности графа, имеющая верхнетреугольный вид

$$\mathbf{A} = [a_{i,j}]\,, \tag{2.37}$$

где

$$a_{i,j} = \begin{cases} 1, \text{ если узлы с номерами } i \text{ и } j \text{ соединены дугой} \\ 0 \text{ — иначе} \end{cases}, \; i,j = \overline{1,L},$$

L — число узлов графа. Единичные элементы полученной матрицы смежности (2.37) соответствуют дугам исходного графа, поэтому они заменяются на номер связанной с этой дугой функции с одним аргументом из множества F_1. На главной диагонали за исключением строк, соответствующих узлам-источникам, ставится номер функции с двумя аргументами из F_2, соответствующий узлу в текущей строке.

Полученная верхнетреугольная матрица вида

$$\mathbf{\Psi} = [\psi_{i,j}]\,, \; i,j = \overline{1,L},$$

называется матрицей сетевого оператора. Матрица сетевого оператора позволяет однозначно представить символьное выражение в закодированном виде. В компьютерной памяти закодированное символьное выражение также представляется в виде матрицы. Элементы матрицы содержат числа, указывающие на номера функций с одним или двумя аргументами из соответствующих множеств.

В случае если кодированию подвергается математическое выражение, то для вычисления значения этого выражения достаточно определить матрицу сетевого оператора и множества аргументов и функций F_0, F_1 и F_2. При этом, для вычисления значения выражения не требуется лексического анализа математического выражения.

Для хранения промежуточных и итоговых результатов вычисления математического выражения необходимо определить вектор узлов

$$\mathbf{z} = \left[\begin{array}{ccc} z_1 & \ldots & z_L \end{array} \right]^T. \tag{2.38}$$

Компоненты вектора узлов (2.38) соответствуют узлу графа сетевого оператора. Каждая компонента вектора узлов инициализируется начальным значением. Для компонент, соответствующих узлам-источникам, — это значение связанной с этим узлом переменной или параметра. Для остальных

компонент — единичный элемент связанной с данным узлом функции от двух аргументов

$$z_i^{(0)} = \begin{cases} f_{0,i}, \text{ если } i \in \{1,\ |\mathrm{F}_0|\} \\ e_{\psi_{i,i}} \text{ — иначе} \end{cases}, \ f_{0,i} \in \mathrm{F}_0,\ i = \overline{1,L}. \tag{2.39}$$

Для вычисления значения математического выражения достаточно одной итерации прохода про строкам матрицы сетевого оператора. В цикле прохода по строкам матрицы при нахождении ненулевого элемента вычисляется новое значение компоненты вектора узлов (2.38), соответствующего столбцу текущего элемента

$$z_j^{(i)} = \begin{cases} f_{2,\psi_{j,j}}\left(z_j^{(i-1)}, f_{1,\psi_{i,j}}\left(z_i^{(i-1)}\right)\right), \text{ если } \psi_{i,j} \neq 0 \\ z_j^{(i-1)} \text{ — иначе} \end{cases}, \tag{2.40}$$

$i = \overline{1, L-1},\ j = \overline{i+1, L}$.

Вычисленное значение математического выражения будет записано в последнюю компоненту вектора узлов (2.38)

$$f(\mathbf{x}) = z_L^{(L-1)}. \tag{2.41}$$

Поиск оптимального кода символьного выражения в форме матрицы сетевого оператора осуществляется с помощью генетического алгоритма на множестве малых вариаций предварительно заданного базисного решения. Использование принципа малых вариаций позволяет определить понятия близости двух возможных решений в форме матриц сетевого оператора, а также задать ограниченную область поиска. Начальное базисное решение задается в виде матрицы базисного сетевого оператора. Там, где это возможно, выбор базисного решения целесообразно производить на основе знаний и опыта исследователя. Малые вариации, применяемые к матрице базисного сетевого оператора, позволяют получать новые возможные решения в форме матрицы сетевого оператора на заданном расстоянии от базисного.

При выполнении малых вариаций необходимо проверять новые возможные решения в форме матрицы сетевого оператора на предмет корректности кода. Код матрицы сетевого оператора считается корректным, если в нем отсутствуют строки и столбцы со всеми нулевыми недиагональными элементами за исключением столбцов, соответствующих узлам-источникам и последней строки матрицы. Условие корректности матрицы сетевого оператора можно

записать следующим образом

$$\forall i \; \exists \psi_{i,j} \neq 0, \;\; i = \overline{1,\, L-1}, \;\; j = \overline{i+1,\, L}, \tag{2.42}$$

$$\forall j \; \exists \psi_{i,j} \neq 0, \;\; j = \overline{|\mathrm{F}_0|+1,\, L}, \;\; i = \overline{1,\, j-1}. \tag{2.43}$$

Метод сетевого оператора предполагает включение в алгоритм процедуры отдельного поиска оптимальных значений используемых параметров. Это является существенным преимуществом данного метода над другими методами из класса методов символьной регрессии, в которых изменение значений параметров осуществляется, как правило, за счёт их нелинейного преобразования. Таким образом, поиск оптимальной структуры символьного выражения осуществляется параллельно с поиском оптимальных значений используемых в выражении параметров.

В качестве примера рассмотрим представление математического выражения (2.1) в закодированном виде в форме матрицы сетевого оператора и последующее применение малых вариаций к получившейся матрице.

Определим множество переменных и параметров

$$\mathrm{F}_0 = \left(f_{0,1} = x_1,\, f_{0,2} = x_2,\, f_{0,3} = q_1,\, f_{0,4} = q_2,\, f_{0,5} = q_3\right), \tag{2.44}$$

множество функций с одним аргументом

$$\mathrm{F}_1 = \left(f_{1,1}(z) = z,\, f_{1,2}(z) = -z,\, f_{1,3}(z) = \sin(z),\, f_{1,4}(z) = \cos(z)\right), \tag{2.45}$$

множество функций с двумя аргументами

$$\mathrm{F}_2 = \left(f_{2,1}(z_1,z_2) = z_1 + z_2,\, f_{2,2}(z_1,z_2) = z_1 z_2\right). \tag{2.46}$$

Множество функций с одним аргументом должно содержать тождественную функцию $f_{1,1}(z) = z$. Функции с двумя аргументами множества (2.46) обладают свойствами коммутативности и ассоциативности и имеют единичный элемент. Для функции $f_{2,1}(z_1,z_2) = z_1 + z_2$ единичный элемент $e_1 = 0$, для функции $f_{2,2}(z_1,z_2) = z_1 z_2$ единичный элемент $e_2 = 1$.

Используя номера функций, переменных и параметров из множеств (2.44) — (2.46), формируем ориентированный граф. Узлы-источники графа будут содержать номера переменных и параметров из множества (2.44), остальные узлы — номера функций с двумя аргументами из множества (2.46), а дуги графа — номера функций с одним аргументом из множества (2.45). Граф сетевого оператора для математического выражения (2.1) представлен на Рис. 2.4.

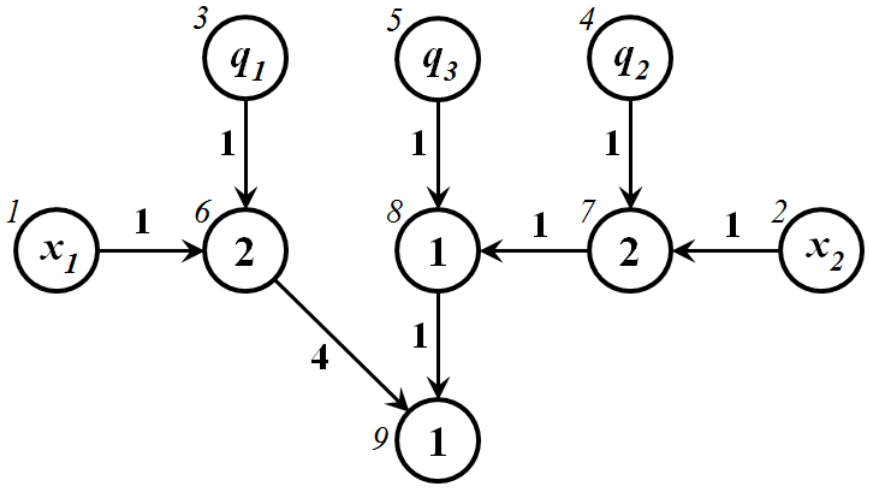


Рисунок 2.4 — Граф сетевого оператора для математического выражения (2.1).

Узлы графа на Рис. 2.4 пронумерованы в соответствии с условиями топологической сортировки. Число узлов полученного графа $L = 9$. Применяя выражение (2.37), строится матрица смежности графа

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{2.47}$$

Единичные элементы полученной матрицы смежности графа (2.47) заменяются на соответствующие номера функций с одним аргументом из множества (2.45). На главной диагонали ставятся соответствующие узлам номера функций с двумя аргументами из множества (2.46). Полученная в результате описанных преобразований матрица является кодом математического выраже-

ния (2.1) в форме матрицы сетевого оператора

$$\mathbf{\Psi}_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.48}$$

Вычисление значения математического выражения не требует обратного декодирования матрицы сетевого оператора. Для вычисления значения выражения кроме самой матрицы сетевого оператора необходима информация о множествах функций, переменных и параметров (2.44) — (2.46), а также определение вектора узлов (2.38) в соответствии с выражением (2.39). Вектор узлов хранит числовые значения переменных и параметров и промежуточные значения вычисления математического выражения. Для выражения (2.1) и его закодированной формы (2.48) вектор узлов имеет вид

$$\mathbf{z}^0 = \begin{bmatrix} x_1 & x_2 & q_1 & q_2 & q_3 & 1 & 1 & 0 & 0 \end{bmatrix}^T.$$

Для поиска структуры оптимального математического выражения в форме сетевого оператора используется принцип малых вариаций базисного решения. В данном примере начальным базисным решением будет сетевой оператор (2.48). Определим вид вектора малой вариации (2.30), применяемый к базисному решению во время поиска

$$\mathbf{w} = \begin{bmatrix} w_1 & w_2 & w_3 & w_4 \end{bmatrix}^T, \tag{2.49}$$

где $w_1 \in \{0, 1, 2, 3\}$ — номер типа малой вариации, $w_2 \in \{0, \ldots, L\}$ — номер строки элемента матрицы, к которому применяется малая вариация, $w_3 \in \{|\mathrm{F}_0| + 1, \ldots, L\}$ — номер столбца элемента матрицы, к которому применяется малая вариация, w_4 — номер функции с одним или двумя аргументами из множеств (2.45) и (2.46).

Для поиска оптимального выражения используются следующие типы малых вариаций:

- $w_1 = 0$ — замена номера функции с одним аргументом;
- $w_1 = 1$ — замена номера функции с двумя аргументами;
- $w_1 = 2$ — удаление функции с одним аргументом;
- $w_1 = 3$ — добавление функции с одним аргументом.

Далее зададим область поиска оптимального решения в окрестности базисного решения. Для этого определим расстояние d (2.29) от базисного решения (2.48). Пусть расстояние $d = 5$. Тогда набор векторов вариаций (2.31) будет состоять из $d = 5$ векторов вариаций (2.49).

Для поиска оптимального выражения на начальном этапе случайным образом генерируется множество заданного размера наборов векторов вариаций (2.31). Отберем случайно два набора из этого множества

$$\mathrm{W}^1 = \Big(\begin{bmatrix} 2 & 3 & 6 & 0 \end{bmatrix}^T, \begin{bmatrix} 3 & 3 & 7 & 1 \end{bmatrix}^T, \begin{bmatrix} 1 & 6 & 6 & 1 \end{bmatrix}^T, \begin{bmatrix} 0 & 7 & 8 & 3 \end{bmatrix}^T, \begin{bmatrix} 0 & 8 & 9 & 2 \end{bmatrix}^T \Big), \tag{2.50}$$

$$\mathrm{W}^2 = \Big(\begin{bmatrix} 2 & 1 & 6 & 0 \end{bmatrix}^T, \begin{bmatrix} 3 & 1 & 7 & 2 \end{bmatrix}^T, \begin{bmatrix} 0 & 5 & 8 & 2 \end{bmatrix}^T, \begin{bmatrix} 1 & 8 & 8 & 2 \end{bmatrix}^T, \begin{bmatrix} 3 & 1 & 9 & 1 \end{bmatrix}^T \Big). \tag{2.51}$$

Набор векторов малых вариаций (2.50), примененный к базисному решению (2.48), позволяет получить новое возможное решение в форме сетевого оператора

$$\boldsymbol{\Psi}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{0} & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & \mathbf{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \mathbf{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.52}$$

В матрице (2.52) полужирным шрифтом выделены элементы, которые подверглись изменению согласно набору малых вариаций (2.50).

Матрица сетевого оператора (2.52) является корректной на основании условия корректности матрицы сетевого оператора (2.42) и (2.43) и соответствует математическому выражению

$$f_1(\mathbf{x}) = x_1 - q_3 - \sin(q_1 q_2 x_2).$$

Набор векторов малых вариаций (2.51), примененный к базисному решению (2.48), позволяет получить новое возможное решение

$$\mathbf{\Psi}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \mathbf{0} & \mathbf{2} & 0 & \mathbf{1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{2} & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.53}$$

Элементы, подвергшиеся изменению в матрице (2.53), выделены полужирным шрифтом.

Матрица сетевого оператора (2.53) является корректной на основании условия корректности матрицы сетевого оператора (2.42) и (2.43) и соответствует математическому выражению

$$f_2(\mathbf{x}) = x_1 + q_1 + \cos(q_2 q_3 x_1 x_2).$$

Выполним операцию скрещивания для отобранных наборов малых вариаций (2.50) и (2.51). Для этого случайным образом определим точку скрещивания k, $k \in \{1, \ldots, d\}$. Пусть $k = 4$, тогда правые части наборов векторов малых вариаций (2.50) и (2.51), начиная от точки скрещивания $k = 4$ и заканчивая последним элементом, обмениваются местами, в результате чего получаются два новых набора векторов малых вариаций

$$\tilde{\mathrm{W}}^1 = \left(\begin{bmatrix} 2 & 3 & 6 & 0 \end{bmatrix}^T, \begin{bmatrix} 3 & 3 & 7 & 1 \end{bmatrix}^T, \begin{bmatrix} 1 & 6 & 6 & 1 \end{bmatrix}^T, \begin{bmatrix} 1 & 8 & 8 & 2 \end{bmatrix}^T, \begin{bmatrix} 3 & 1 & 9 & 1 \end{bmatrix}^T \right), \tag{2.54}$$

$$\tilde{\mathrm{W}}^2 = \left(\begin{bmatrix} 2 & 1 & 6 & 0 \end{bmatrix}^T, \begin{bmatrix} 3 & 1 & 7 & 2 \end{bmatrix}^T, \begin{bmatrix} 0 & 5 & 8 & 2 \end{bmatrix}^T, \begin{bmatrix} 0 & 7 & 8 & 3 \end{bmatrix}^T, \begin{bmatrix} 0 & 8 & 9 & 2 \end{bmatrix}^T \right). \tag{2.55}$$

Набор векторов малых вариаций (2.54), примененный к базисному решению (2.48), позволяет получить возможное решение

$$\tilde{\mathbf{\Psi}}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{0} & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{2} & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.56}$$

Элементы, подвергшиеся изменению в матрице (2.56), выделены полужирным шрифтом.

Матрица сетевого оператора (2.56) является корректной на основании условия корректности матрицы сетевого оператора (2.42) и (2.43) и соответствует математическому выражению

$$\tilde{f}_1(\mathbf{x}) = 2x_1 + \cos(q_1 q_2 q_3 x_2).$$

Набор векторов малых вариаций (2.55), примененный к базисному решению (2.48), позволяет получить новое возможное решение

$$\tilde{\mathbf{\Psi}}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \mathbf{0} & \mathbf{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & \mathbf{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \mathbf{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.57}$$

Элементы, подвергшиеся изменению в матрице (2.57), выделены полужирным шрифтом.

Матрица сетевого оператора (2.57) является корректной на основании условия корректности матрицы сетевого оператора (2.42) и (2.43) и соответствует математическому выражению

$$\tilde{f}_2(\mathbf{x}) = q_1 + q_3 - \sin(-q_2 x_1 x_2).$$

Для набора векторов малых вариаций (2.54) выполним операцию мутации. Для выполнения операции мутации в отобранном наборе векторов вариаций случайным образом определяется точка мутации α, $\alpha \in \{1, \ldots, d\}$. Вектор вариаций $\mathbf{w}^{\alpha}$ на позиции α отобранного набора векторов малых вариаций заменяется на новый вектор $\tilde{\mathbf{w}}^{\alpha}$, выбранный случайным образом из множества малых вариаций (2.28).

Пусть $\alpha = 3$, а новый вектор малых вариаций $\tilde{\mathbf{w}}^{\alpha}$ имеет вид

$$\tilde{\mathbf{w}}^{\alpha} = \begin{bmatrix} 3 & 2 & 8 & 1 \end{bmatrix}^T,$$

тогда набор векторов малых вариаций (2.54) после применения операции мутации будет иметь вид

$$\begin{aligned} \tilde{\mathrm{W}}_1 = \Big(& \begin{bmatrix} 2 & 3 & 6 & 0 \end{bmatrix}^T, \begin{bmatrix} 3 & 3 & 7 & 1 \end{bmatrix}^T, \begin{bmatrix} 3 & 2 & 8 & 1 \end{bmatrix}^T, \\ & \begin{bmatrix} 1 & 8 & 8 & 2 \end{bmatrix}^T, \begin{bmatrix} 3 & 1 & 9 & 1 \end{bmatrix}^T \Big). \end{aligned} \tag{2.58}$$

Полученный в результате операции мутации набор векторов малых вариаций (2.58), примененный к базисному решению (2.48), позволяет получить новое возможное решение

$$\tilde{\boldsymbol{\Psi}}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \mathbf{1} & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{0} & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{2} & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.59}$$

Элементы, подвергшиеся изменению в матрице (2.59), выделены полужирным шрифтом.

Матрица сетевого оператора (2.59) является корректной на основании условия корректности матрицы сетевого оператора (2.42) и (2.43) и соответствует математическому выражению

$$\tilde{f}_1(\mathbf{x}) = 2x_1 + \cos(q_1 q_2 q_3 x_2^2).$$

В результате применения генетических операций над множеством наборов векторов вариаций получаем новые наборы и, соответственно, новые возможные решения в форме сетевого оператора. В процессе поиска оптимального решения может производиться смена базисного решения на новое. Новым базисным решением выбирается лучшее на текущий момент известное решение. Процесс замены базисного решения называется сменой эпохи.

Главным преимуществом метода сетевого оператора является то, что он создавался с целью решения в первую очередь задачи синтеза системы управления. В нем используется принцип малых вариаций, что позволяет определять ограниченную область поиска решения и числовую метрику близости двух возможных решений. На скорость поиска методом сетевого оператора положительно влияет отсутствие необходимости в лексическом анализе математического выражения. Вычислить значение выражения можно по самой матрице сетевого оператора. Применение генетических операций на наборах малых вариаций также упрощает и ускоряет процесс поиска. К недостаткам метода сетевого оператора можно отнести ограниченность набора используемых функций. В методе сетевого оператора применимы только функции с одним и двумя аргументами, причем функции с двумя аргументами должны удовлетворять условиям коммутативности и ассоциативности.

Выводы по Главе 2

Методы символьной регрессии — это численные методы поиска оптимального решения на нечисловом пространстве. Они предоставляют широкие возможности для задач, в которых решением является набор символов, например, программные коды, элементы электрических цепей, математические

функции и т.д. Поэтому данные методы представляют большой интерес в области решения задачи синтеза системы управления, где требуется найти структуру математической функции управления.

В методах символьной регрессии сложные математические выражения представляются в виде суперпозиции кодов заданных функций и арифметических операций над ними. Дальнейший поиск осуществляется на нечисловом пространстве кодов с помощью генетического алгоритма. Кодирование математического выражения является обратимой операцией. По окончанию поиска, найденное решение может быть декодировано и представлено в форме структуры математического выражения и значений его параметров.

Методы символьной регрессии различаются между собой алгоритмами кодирования символьного выражения и алгоритмами поиска на пространстве кодов. При этом все известные методы построены на использовании разновидностей генетического алгоритма для поиска на пространстве кодов.

Первым и наиболее широко известным методом символьной регрессии является метод генетического программирования. В нём закодированное математическое выражение представляется в виде упорядоченного множества векторов кодов функций. Далее для получения новых кодов применяются генетические операции скрещивания и мутации. При этом возникает задача проверки корректности записи нового выражения. Также спецификой метода генетического программирования является нефиксированная длина записи кода математического выражения. Оба этих факта являются существенными недостатками данного метода и усложняют его прикладное применение.

Методы, появившиеся после метода генетического программирования, в большинстве случаев создавались с целью частичной или полной компенсации описанных выше недостатков. Так в методе декартового генетического программирования и методе грамматической эволюции используются коды выражений одинаковой длины. Это в свою очередь значительно упрощает применение операции скрещивания. Однако наиболее совершенным из рассмотренных методов является метод сетевого оператора. Данный метод выделяется на фоне остальных тем, что при его создании учитывалась специфика задачи синтеза системы управления. Помимо фиксированной длины кодов выражений и более простой реализации операции скрещивания, в нём используется принцип малых вариаций базисного решения. Это позволяет определять ограниченную область поиска решения и числовую метрику близости двух возможных решений.

Глава 3. Синтез системы управления на основе аппроксимации множества оптимальных траекторий методами символьной регрессии

В диссертации предлагается новый метод численного решения задачи синтеза системы управления и нахождения многомерной функции управления (1.6) на основе аппроксимации множества предварительно найденных оптимальных траекторий. Для нахождения структуры многомерной функции управления используются методы символьной регрессии, в частности предлагается использовать рассмотренный в разделе 2.2.5 метод сетевого оператора, который более других приспособлен для решения задачи синтеза системы управления.

Следует ещё раз отметить, что применение методов символьной регрессии для решения задачи синтеза системы управления уже было рассмотрено в работах А.И. Дивеева [95; 96]. Представленный в этих работах подход основан на структурно-параметрической многокритериальной оптимизации и действительно позволяет находить структуру математического выражения функции управления в явном виде, однако он не позволяет оценить близость найденного решения к оптимальному. При таком подходе методы символьной регрессии используются впрямую для подбора кода многомерной функции управления. Это позволяет найти наилучшее решение на пространстве кодов всех рассмотренных возможных решений, но метрика близости найденного решения к оптимальному не определена и, соответственно, не может быть оценена.

Предлагаемый в диссертации подход лишен описанного выше недостатка, так как поиск структуры математического выражения методами символьной регрессии осуществляется не впрямую, а путем аппроксимации множества предварительно найденных оптимальных траекторий. Также это позволяет перейти от многокритериальной оптимизации к поиску решения с учётом одного критерия качества. В логике использования методов символьной регрессии в данном случае происходит переход от поиска путём подбора к поиску путём обучения. Найденная таким образом многомерная функция управления будет доставлять управление, близкое к оптимальному, причем его близость к оптимальному решению определяется точностью аппроксимации и размером обучающего множества оптимальных траекторий.

3.1 Решение задачи численного синтеза системы управления на основе аппроксимации оптимальных траекторий

Для численного решения задачи синтеза системы управления разработан новый подход на основе аппроксимации множества оптимальных траекторий. Данный подход предполагает два основных этапа:

1. Поиск множества оптимальных траекторий для заданного множества начальных состояний;
2. Поиск функции управления от координат состояния, аппроксимирующей полученное множество оптимальных траекторий.

На первом этапе решается задача поиска множества оптимальных траекторий. Такое множество может быть получено путем многократного решения задачи оптимального управления для ограниченного числа начальных состояний из непрерывного множества (1.3). Для этого заменим непрерывное множество (1.3) конечным множеством из N элементов

$$\tilde{\mathrm{X}}_0 = \left(\mathbf{x}^{0,1}, \ldots, \mathbf{x}^{0,N}\right). \tag{3.1}$$

Решением каждой отдельно взятой задачи оптимального управления для одного начального состояния из множества (3.1) будет управление как функция от времени $\tilde{\mathbf{u}}^j(t)$, $j \in \{1, \ldots, N\}$, N — размер множества (3.1). При данном управлении перемещение объекта управления из начального состояния в терминальное положение осуществляется по оптимальной траектории, также называемой экстремалью. Множество оптимальных управлений $\tilde{\mathbf{u}}^j(t)$, $j = \overline{1,N}$, полученное путем решения задачи оптимального управления для каждого начального состояния из множества (3.1), даст нам соответствующее множество оптимальных траекторий.

Здесь следует отдельно подчеркнуть разницу между задачей оптимального управления и задачей синтеза системы управления. В случае задачи оптимального управления решением будет функция управления от времени

$$\mathbf{u} = \mathbf{v}(t). \tag{3.2}$$

Данное решение доставляет оптимальную траекторию только для одного начального состояния объекта управления.

В случае задачи синтеза системы управления решение представляет собой функцию управления от вектора компонент состояния объекта (1.6). Данное решение доставляет оптимальную траекторию перемещения объекта для любого начального состояния из множества возможных состояний или его ограниченной области.

Подставив управления вида (3.2) и вида (1.6) в (1.2) получим соответствующие записи математической модели объекта управления

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x},\, \mathbf{v}\left(t\right)\right), \tag{3.3}$$

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x},\, \mathbf{h}\left(\mathbf{x}\right)\right). \tag{3.4}$$

Решения $\mathbf{x}\left(\mathbf{x}^0,t\right)$ систем (3.3) и (3.4), обеспечивающие перемещение объекта управления из одного и того же начального состояния $\mathbf{x}^0 \in \tilde{\mathrm{X}}_0$ из множества (3.1) в терминальное положение $\mathbf{x}^f$ по оптимальной траектории, должны доставлять минимум функционалу качества (1.5)

$$\begin{gathered}\int\limits_0^{t_f} f_0\left(\mathbf{x}\left(\mathbf{x}^0,t\right),\, \mathbf{v}\left(t\right)\right) dt = \int\limits_0^{t_f} f_0\left(\mathbf{x}\left(\mathbf{x}^0,t\right),\, \mathbf{h}\left(\mathbf{x}\left(\mathbf{x}^0,t\right)\right)\right) dt = \\ = \min_{\mathbf{u}\in\tilde{\mathrm{U}}} \int\limits_0^{t_f} f_0\left(\mathbf{x},\mathbf{u}\right) dt.\end{gathered} \tag{3.5}$$

Другими словами, решение системы (3.3) для выбранного начального состояния $\mathbf{x}^0 \in \tilde{\mathrm{X}}_0$ из множества (3.1) должно обеспечивать получение такой же оптимальной траектории, а, следовательно, такого же значения функционала качества, как и решение задачи синтеза системы управления для этого же конкретного начального состояния $\mathbf{x}^0$.

Решив задачу оптимального управления для каждого начального состояния объекта из множества (3.1), мы получим множество функций управления от времени

$$\tilde{\mathrm{V}} = \left(\mathbf{v}^1(t), \ldots, \mathbf{v}^N(t)\right), \tag{3.6}$$

где $\mathbf{v}^j(t)$ — решение задачи оптимального управления для начального условия $\mathbf{x}^{0,j}$, $j = \overline{1,N}$, удовлетворяющее функционалу качества (3.5) и терминальным условиям (1.4).

Для поиска решений задачи оптимального управления (3.6) можно использовать любой из известных методов решения. При этом отдельно возникает

задача выбора наиболее подходящего подхода и метода её решения. Используемый метод должен, с одной стороны, быть универсальным и учитывать специфику прикладных задач оптимального управления, а, с другой стороны, быть достаточно точным для получения репрезентативных результатов. Данному вопросу посвящена Глава 4 настоящей диссертации.

Подставив каждое решение $\mathbf{v}^j(t)$ из множества (3.6) в модель объекта управления (3.3), получим множество пар оптимальных траекторий и программных управлений:

$$\tilde{\mathrm{D}} = \left\{ \left(\tilde{\mathbf{x}}^1(\cdot), \tilde{\mathbf{u}}^1(\cdot) \right), \ldots, \left(\tilde{\mathbf{x}}^N(\cdot), \tilde{\mathbf{u}}^N(\cdot) \right) \right\}, \tag{3.7}$$

где $\tilde{\mathbf{x}}^j(\cdot)$ — частное решение системы (3.3) для начального условия $\mathbf{x}(0) = \mathbf{x}^{0,j}$, $\tilde{\mathbf{u}}^j(\cdot)$ — решение задачи оптимального управления для заданного начального условия и с учетом ограничений на управление $\tilde{\mathbf{u}}^j(t) \in \mathrm{U} \subseteq \mathbb{R}^m \ \forall t \in [0; t_f]$, $j = \overline{1,N}$.

На втором этапе решается задача поиска многомерной функции управления от координат состояния объекта (1.6) путем аппроксимации полученного множества оптимальных траекторий и программных управлений (3.7).

Для численной аппроксимации множества (3.7) вводим дискретизацию по времени. Для этого задаем малое значение $\Delta_s t > 0$ и определяем множество дискретных значений времени для всех найденных решений из множества (3.6)

$$\mathrm{T}_j = (0,\, \Delta_s t,\, 2\Delta_s t, \ldots,\, M_j \Delta_s t),$$

где $M_j = \left\lceil \dfrac{t_f\left(\mathbf{x}^{0,j}\right)}{\Delta_s t} \right\rceil$, $t_f\left(\mathbf{x}^{0,j}\right)$ — время процесса управления для найденного решения задачи оптимального управления из начального состояния $\mathbf{x}^{0,j}$, $j = \overline{1,N}$. Тогда значения векторов состояния и управления для каждого начального состояния в дискретный момент времени $t_j \in \mathrm{T}_j$ можно записать в виде

$$\tilde{\mathbf{x}}^{j,i} = \tilde{\mathbf{x}}^j(t_{j,i}),$$

$$\tilde{\mathbf{u}}^{j,i} = \tilde{\mathbf{u}}^j(t_{j,i}),$$

где $i = \overline{1,M_j}$, $j = \overline{1,N}$.

В итоге получаем множество точек оптимальных траекторий и программных управлений в пространстве $\mathbb{R}^n \times \mathbb{R}^m$

$$\begin{aligned} \mathrm{D} = \ & \left\{ \left(\left(\tilde{\mathbf{x}}^{1,1}, \tilde{\mathbf{u}}^{1,1} \right), \ldots, \left(\tilde{\mathbf{x}}^{1,M_1}, \tilde{\mathbf{u}}^{1,M_1} \right) \right), \ldots, \right. \\ & \left. \left(\left(\tilde{\mathbf{x}}^{N,1}, \tilde{\mathbf{u}}^{N,1} \right), \ldots, \left(\tilde{\mathbf{x}}^{N,M_N}, \tilde{\mathbf{u}}^{N,M_N} \right) \right) \right\}. \end{aligned} \tag{3.8}$$

Решением задачи численного синтеза системы управления будет многомерная функция управления от координат состояния объекта вида (1.6), удовлетворяющая критерию

$$J = \sum_{j=1}^{N} \sum_{i=1}^{M_j} \left\| \tilde{\mathbf{x}}^{j,i} - \mathbf{x}^j(\mathbf{x}^{0,j}, t_{j,i}) \right\| \to \min, \tag{3.9}$$

где $\mathbf{x}^j\left(\mathbf{x}^{0,j}, t_{j,i}\right)$ — частное решение системы $\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}^j(\mathbf{x}^{0,j}, t_{j,i-1}), \mathbf{h}\left(\mathbf{x}^j(\mathbf{x}^{0,j}, t_{j,i-1})\right)\right)$ для начального условия $\mathbf{x}(0) = \mathbf{x}^{0,j}$ в дискретный момент времени $t_{j,i}$, $i = \overline{1,M_j}$, $\mathbf{x}^j(\mathbf{x}^{0,j}, t_{j,0}) = \mathbf{x}^{0,j}$ — состояние объекта управления в начальный момент времени $t_{j,0} = 0$, $j = \overline{1,N}$.

Значение функционала (3.9) определяет близость рассматриваемого решения к оптимальным траекториям, представленным в виде обучающей выборки (3.8). Чем ближе значение функционала (3.9) к нулю, тем точнее выполнена аппроксимация и ближе решение к оптимальному.

Ключевыми параметрами, влияющими на качество решения задачи синтеза системы управления и его близость к оптимальному, являются размер множества оптимальных траекторий N и значение параметра дискретизации оптимальных траекторий по времени $\Delta_s t$. Увеличение размера множества оптимальных траекторий позволит улучшить качество решения для разных начальных состояний, в том числе тех, которые не входили в исходное множество. В свою очередь уменьшение параметра дискретизации приведёт к увеличению числа M_j, $j = \overline{1,N}$ рассматриваемых дискретных значений для каждой оптимальной траектории, что позволит улучшить качество аппроксимации.

При решении прикладных задач критерий (3.9) может быть дополнен функциями поощрений или штрафов, например за нарушение фазовых ограничений или выход из области допустимых состояний объекта.

В качестве иллюстрации случая, когда штраф за нарушение фазовых ограничений необходимо включить в критерий качества обучения, рассмотрим следующий пример (Рис. 3.1). Пусть известна оптимальная траектория перемещения объекта управления из начального состояния $\mathbf{x}^0$ в терминальное состояние $\mathbf{x}^f$ (сплошная черная линия) и существуют два возможных решения задачи синтеза системы управления $\mathbf{h}_1(\mathbf{x})$ и $\mathbf{h}_2(\mathbf{x})$ и траектории, полученные с их помощью (пунктирные черные линии). Пусть для точек на рассматриваемых траекториях верно следующее неравенство $\left\|\tilde{\mathbf{x}} - \mathbf{x}^1\right\| < \left\|\tilde{\mathbf{x}} - \mathbf{x}^2\right\|$. Тогда согласно (3.9) возможное решение $\mathbf{h}_1(\mathbf{x})$ является более качественным. Однако

из Рис. 3.1 видно, что часть траектории, полученной с помощью данного решения нарушает фазовые ограничения (область, ограниченная красной сплошной линией), в том числе и точка $\mathbf{x}^1$, а следовательно возможное решение $\mathbf{h}_1(\mathbf{x})$ должно быть исключено из рассмотрения.

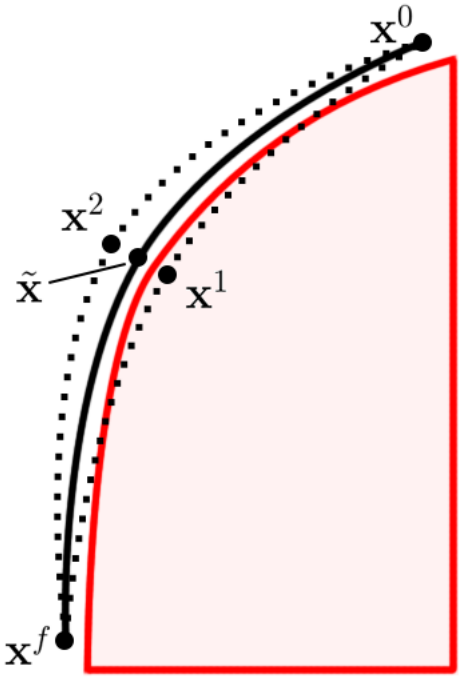


Рисунок 3.1 — Пример. Случай необходимости включения штрафа за нарушение фазовых ограничений в критерий качества обучения.

Для задач с фазовыми ограничениями добавим в критерий (3.9) штраф за их нарушение

$$J = \sum_{j=1}^{N} \sum_{i=1}^{M_j} \left\| \tilde{\mathbf{x}}^{j,i} - \mathbf{x}^j \left(\mathbf{x}^{0,j}, t_{j,i} \right) \right\| + \sum_{k=1}^{z} \alpha_k \vartheta \left(h_k \left(\mathbf{x} \right) \right) h_k \left(\mathbf{x} \right) \to \min, \qquad (3.10)$$

где α_k — заданный штрафной коэффициент, $h_k(\mathbf{x})$ — фазовое ограничение в пространстве вектора состояния объекта, $k = \overline{1,z}$, z — число фазовых ограничений, $\vartheta(h_k(\mathbf{x}))$ — функция Хэвисайда (1.22).

Задача поиска многомерной функции управления (1.6), подстановка которой в математическую модель объекта управления (1.2) для разных начальных состояний из множества (3.1) будет давать частные решения, удовлетворяющие критерию качества (3.10), может быть представлена как задача обучения с подкреплением [67].

Обучение с подкреплением эффективно используется для решения задач поиска оптимального управления [98; 109]. В отличие от обучения с учителем,

при котором осуществляется поиск функциональной зависимости программных управлений $\mathbf{u}^{j,i}$ от значений состояния объекта $\mathbf{x}^{j,i}$, $i = \overline{1,M_j}$, $j = \overline{1,N}$, в обучении с подкреплением присутствует явная обратная связь, которая заключается во влиянии так называемого отклика системы — состояния $\mathbf{x}^{j,i}$ на последующие решения, получаемые функцией управления.

В настоящее время большую популярность и развитие получили искусственные нейронные сети и связанные с ними технологии машинного обучения [104]. В связи с этим довольно часто понятия “обучения” и “обучающей выборки” связывают именно с ними. Однако данные понятия могут относиться не только к применению методов машинного обучения при использовании нейронных сетей, но и к множеству других подходов [133].

В данном случае многослойная искусственная нейронная сеть играет роль универсального аппроксиматора при поиске функциональной зависимости управления от состояния объекта на основе обучающей выборки (3.8) [91]. Помимо нейронных сетей в качестве таких аппроксиматоров могут также использоваться методы на основе дерева решений [100], линейной комбинации радиально-базисных функций [128], локально-взвешенной линейной регрессии [85] и методы символьной регрессии [140].

Выбор подходящего аппроксиматора и его настройка, например, выбор количества и типов базисных функций или количества и размеров слоев искусственной нейронной сети, требует значительных усилий и большого объема экспертных знаний. При этом настроенный аппроксиматор будет представлять собой черный ящик, то есть получить структуру аппроксимирующей функции и провести ее анализ не получится.

Из-за высокой сложности настройки и невозможности получить в явном виде структуру аппроксимирующей функции, подходы на основе нейронных сетей не используются в задачах синтеза системы управления [90]. Напротив, в работе [140] показано, что методы символьной регрессии могут эффективно использоваться для получения аналитического выражения на основе обучающей выборки.

3.2 Постановка задачи численного синтеза системы управления на основе аппроксимации множества оптимальных траекторий методами символьной регрессии

Предлагаемый в диссертации подход численного решения задачи синтеза системы управления и нахождения многомерной функции управления путем аппроксимации множества оптимальных траекторий использует принципы обучения с подкреплением. Ввиду того, что решением задачи синтеза системы управления является математическое выражение функции управления (1.6), для его поиска необходимо осуществить выбор эффективного аппроксиматора, позволяющего получить искомое выражение в явном виде. В качестве такого аппроксиматора в диссертации предлагается использовать методы символьной регрессии.

Сформулируем постановку задачи численного синтеза системы управления на основе аппроксимации множества оптимальных траекторий методами символьной регрессии.

Пусть задача синтеза системы управления в исходном виде задана с помощью выражений (1.2) — (1.6). Произведем замену непрерывного множества начальных состояний (1.3) на конечное множество из N элементов (3.1).

На первом этапе для каждого начального состояния из дискретного множества (3.1) путем решения задачи оптимального управления необходимо найти оптимальную траекторию перехода в терминальное состояние (1.4). Для этого вводим критерий качества

$$J_{ocp}^{j} = \int_{t_0}^{t_f} f_0\left(\mathbf{x},\mathbf{u}\right) dt \to \min\,,\;\; j = \overline{1,N} \tag{3.11}$$

и решаем N задач оптимального управления по переводу объекта управления из состояния $\mathbf{x}^{0,j} \in \tilde{\mathrm{X}}_0$ в состояние $\mathbf{x}^f$. В итоге получаем решение в виде множества функций управления от времени (3.6), доставляющих минимум критерию качества (3.11).

Подставив найденные функции управления в математическую модель объекта (1.2), получим оптимальные траектории, соответствующие каждому начальному состоянию.

Далее необходимо подготовить выборку данных для их последующей аппроксимации. Для этого зададим значение параметра дискретизации $\Delta_s t > 0$

и получим множество точек оптимальных траекторий и программных управлений (3.8).

В соответствии с предложенным методом для аппроксимации используется только информация об оптимальных траекториях. Тогда для последующего поиска математического выражения функции управления достаточно получить множество точек оптимальных траекторий вида

$$\mathrm{D} = \left\{ \left(\tilde{\mathbf{x}}^{1,1},\tilde{\mathbf{x}}^{1,2},\ldots,\tilde{\mathbf{x}}^{1,M_1}\right),\ldots,\left(\tilde{\mathbf{x}}^{N,1},\tilde{\mathbf{x}}^{N,2},\ldots,\tilde{\mathbf{x}}^{N,M_N}\right)\right\}, \tag{3.12}$$

где $\tilde{\mathbf{x}}^{j,i} = \tilde{\mathbf{x}}^{j}\left(t_{j,i}\right)$ — вектор состояния объекта управления в дискретный момент времени $t_{j,i}$ при старте из точки начального состояния $\mathbf{x}^{0,j}$, принадлежащей множеству (3.1), $i = \overline{1,M_j}$, M_j — число точек дискретизации оптимальной траектории из начального состояния $\mathbf{x}^{0,j}$, $j = \overline{1,N}$.

На втором этапе осуществляем поиск структуры и параметров функции управления, аппроксимирующей множество точек оптимальных траекторий (3.12). Для этого введём обозначение записи искомой многомерной функции управления (1.6) в виде кода метода символьной регрессии

$$\mathbf{g}\left(\mathbf{x},\mathbf{s}\right), \tag{3.13}$$

где $\mathbf{s}$ — вектор постоянных параметров, заданной размерности K, $\mathbf{s} = \left[s_1 \ldots s_K\right]^T$.

Решением задачи численного синтеза системы управления на основе аппроксимации множества оптимальных траекторий с помощью методов символьной регрессии будет запись оптимальной структуры функции управления

$$\tilde{\mathbf{u}} = \mathbf{g}^{*}\left(\mathbf{x},\mathbf{s}^{*}\right) \tag{3.14}$$

и оптимальные значения ее параметров $\mathbf{s}^{*}$, доставляющие минимум критерию качества (3.9).

Главным преимуществом предлагаемого метода является возможность в процессе поиска оптимального выражения функции управления в форме (3.14) производить оценку близости текущих возможных решений к оптимальному на основе значения критерия качества (3.9). Также следует отметить, что в приведенной постановке задачи численного синтеза системы управления не требуется производить многокритериальную оптимизацию, что также положительно выделяет предлагаемый подход при сравнении с методом, рассмотренным в разделе 1.3 диссертации.

Эффективность предлагаемого подхода исследовалась на различных прикладных задачах синтеза системы управления. Положительные результаты проведенных исследований отражены в работах [172; 174—176; 179; 180] и подкрепляют доводы, приводимые в диссертации.

Выводы по Главе 3

Аналитические методы решения задачи синтеза системы управления зачастую неприменимы для сложных прикладных задач. В таких случаях решение поставленной задачи следует искать на основе численных подходов. Рассмотренный метод поиска структуры многомерной функции управления на основе аппроксимации множества предварительно найденных оптимальных траекторий является новым численным методом решения задачи синтеза системы управления. Он позволяет находить многомерную функцию управления в аналитическом виде, используя для этого численные методы символьной регрессии.

Отличительной особенностью предложенного метода от других известных подходов численного синтеза системы управления является его соответствие идеологии обучения с подкреплением. В методе предварительно формируется обучающая выборка на основе множества оптимальных траекторий. Для её формирования необходимо получить решения задачи оптимального управления для разных начальных условий. Далее строится критерий качества, позволяющий оценить соответствие текущего решения эталонному из обучающей выборки, и осуществляется поиск многомерной функции управления, доставляющей минимум данному критерию.

Такой подход позволяет использовать показатель качества обучения как оценку близости найденного решения к оптимальному. Также появляется возможность управления качеством обучения с помощью увеличения размера и полноты обучающей выборки. При этом поиск оптимального решения производится на множестве всего одного функционала, а, значит, нет необходимости в поиске и выборе решения на множестве Парето, что свойственно для многокритериальной оптимизации.

Поиск структуры, максимально точно аппроксимирующей обучающую выборку, предлагается осуществлять с помощью методов символьной регрес-

сии. В методах символьной регрессии сложные математические выражения представляются в виде суперпозиции кодов заданных функций и арифметических операций над ними. Дальнейший поиск осуществляется на нечисловом пространстве кодов с помощью генетического алгоритма. Кодирование математического выражения методами символьной регрессии является обратимой операцией. По окончанию поиска, найденное решение может быть декодировано и представлено в форме структуры математического выражения и значений его параметров.

Разработанный метод решения задачи численного синтеза системы управления на основе аппроксимации оптимальных траекторий был протестирован на различных прикладных задачах и показал свою эффективность. Основные результаты проведенных исследований опубликованы в работах [172; 174—176; 179; 180].

Глава 4. Применение эволюционных алгоритмов для численного решения задачи оптимального управления

Бурное развитие сложных технических систем в области самолетостроения, ракетостроения и в других производственных сферах в конце 50-х годов прошлого века привело к необходимости решения большого числа задач оптимального управления. Этим обусловлено создание и исследование в это же время численных методов решения задач оптимального управления [26]. Дополнительным стимулом к развитию численных методов решения задач оптимального управления послужило появление и внедрение первых ЭВМ.

Как правило, при упоминании численных методов решения задач оптимального управления, имеют в виду прямые методы, основанные на преобразовании исходной задачи поиска оптимального управления к задаче конечномерной оптимизации и применении методов нелинейного программирования [59]. Однако, существует и ряд исследований, связанных с непрямыми методами, в которых исходная задача редуцируется на основе принципа максимума Понтрягина к краевой [52], а также связанные с численными методами решения задачи оптимального управления на основе динамического программирования [9; 54].

Наиболее эффективным и активно использующимся по сей день оказался метод, основанный на применении методов нелинейного программирования. Данный метод активно используется для решения широкого круга сложных задач оптимального управления [18]. Он позволяет использовать множество методов безусловной оптимизации, начиная от градиентных методов и заканчивая разнообразными современными алгоритмами.

Так в работах [18; 27; 53] при решении задачи оптимального управления, сведенной к задаче нелинейного программирования, применяются градиентные методы для определения направления поиска. Это накладывает определенные требования к целевой функции, в частности требуется выполнение условий гладкости, выпуклости и унимодальности целевой функции. Проверка выполнения данных условий для прикладных задач оптимального управления может потребовать существенных усилий, которые по трудоёмкости могут превышать усилия по нахождению самого решения.

Для задач оптимального управления с фазовыми ограничениями отсутствие сведений о топологических свойствах целевой функции является характерной особенностью, так как определить свойства целевой функции и гарантировать её выпуклость даже на ограниченной области пространства поиска достаточно сложно. В случаях, когда целевая функция содержит большое количество локальных экстремумов, методы нелинейного программирования, использующие градиентные методы для определения направления поиска, могут застревать в точках локального минимума и седловых точках, что не позволяет получить информацию о глобальном оптимальном решении в пространстве поиска.

Рассматриваемые в данной главе современные поисковые алгоритмы основаны на идеях, имеющих аналогии с приспособленностью и эволюционированием живых организмов. Эволюционные алгоритмы не требуют выполнения свойств выпуклости и других специальных свойств и позволяют эффективно находить приближенное оптимальное решение [33]. До недавнего времени широкому развитию и применению данных алгоритмов препятствовала лишь слабая вычислительная мощность ЭВМ. Современные вычислительные системы позволяют использовать и применять на практике все преимущества эволюционных алгоритмов.

Следует отметить, что большинство из существующих эволюционных алгоритмов хорошо исследованы. Но в основном они исследовались на задачах конечномерной оптимизации, в частности, их оценивали на решениях тестовых функций. Ещё десять лет назад научные работы, посвященные исследованию применения эволюционных алгоритмов в задачах оптимального управления, отсутствовали. Данное направление стало активно развиваться научным коллективом во главе с профессором А.И. Дивеевым начиная с 2014 года. Так в работах [157—161; 165; 167; 169; 178] было проведено масштабное исследование, показавшее высокую эффективность применения эволюционных алгоритмов для решения задачи оптимального управления.

Отличие исследования эволюционных алгоритмов на предмет решения задачи оптимального управления состоит в том, что явно не задана целевая функция на пространстве параметров. Для того, чтобы получить значение целевой функции, сначала требуется проинтегрировать систему дифференциальных уравнений, а потом на основе полученных решений получить значение целевой функции. Это является характерной особенностью не только для эволюцион-

ных алгоритмов, но и для любых других, в том числе и градиентных, при решении задачи оптимального управления путем редукции к задаче нелинейного программирования.

Интегрирование системы дифференциальных уравнений в прикладных задачах может представлять гораздо бóльшую вычислительную сложность, то есть занимать больше вычислительного времени, чем процедуры преобразования искомого вектора на текущей итерации и вычисления фазового состояния объекта управления. Таким образом, более точную оценку сложности алгоритма решения задачи оптимального управления можно получить оценивая число вычислений значений целевой функции, а не по числу произведенных итераций поиска.

Ещё одной характерной особенностью задач оптимального управления, решаемых методами нелинейного программирования, является большая размерность пространства искомого вектора решения. Редукция задачи оптимального управления к задаче нелинейного программирования состоит в дискретизации функций управления по времени. Размерность вектора решения при этом будет равна произведению количества точек дискретизации и размерности вектора управления. При таком подходе, чем больше точек дискретизации, тем точнее численное решение задачи оптимального управления, но тем больше размерность пространства искомого вектора решения.

Здесь следует отметить, что даже для простейшей прикладной задачи с векторами состояния и управления небольшой размерности и достаточно грубой редукции к задаче нелинейного программирования, пространство поиска вектора решения будет иметь весьма большую размерность.

В методах нелинейного программирования, в которых используются градиентные методы для определения направления поиска, число вычислений целевой функции равно размерности искомого вектора решения, а в методах второго порядка — квадрату его размерности. Таким образом, в прикладных задачах оптимального управления, решаемых методами нелинейного программирования с использованием градиентных методов, вычисление значений целевой функции на каждом шаге занимает бóльшую часть вычислительного процесса. Это ещё раз подтверждает справедливость предположения об эффективности оценки вычислительной сложности численных методов решения задач оптимального управления по количеству вычислений целевой функции.

4.1 Постановка задачи оптимального управления с фазовыми ограничениями

В классической постановке задачи оптимального управления объект управления записывается в виде системы дифференциальных уравнений [51]

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},\mathbf{u}),$$

где $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{x} = [x_1 \dots x_n]^T$ — вектор состояния, $\mathbf{u} \in \mathrm{U} \subseteq \mathbb{R}^m$, $\mathbf{u} = [u_1 \dots u_m]^T$ — вектор управления, $\mathbf{f}(\mathbf{x},\mathbf{u}) = [f_1(\mathbf{x},\mathbf{u}) \dots f_n(\mathbf{x},\mathbf{u})]^T$.

Заданы ограничения на управление

$$\mathbf{u}^- \leqslant \mathbf{u} \leqslant \mathbf{u}^+,$$

где $\mathbf{u}^-$, $\mathbf{u}^+$ — заданные постоянные векторы ограничений, $\mathbf{u}^- = \left[u_1^- \dots u_m^-\right]^T$, $\mathbf{u}^+ = \left[u_1^+ \dots u_m^+\right]^T$.

Многомерная функция $\mathbf{f}(\mathbf{x},\mathbf{u})$ определена для любых возможных значений вектора состояний $\mathbf{x}$ и любых возможных значений вектора управления $\mathbf{u}$.

Заданы начальные условия

$$t_0 \geqslant 0,$$

$$\mathbf{x}(t_0) = \mathbf{x}^0,$$

где t_0 — начальное время, $\mathbf{x}^0$ — заданный вектор состояния объекта в начальный момент времени t_0, $\mathbf{x}^0 = \left[x_1^0 \dots x_n^0\right]^T$.

Заданы терминальные условия

$$\mathbf{x}(t_f) = \mathbf{x}^f,$$

где $\mathbf{x}^f$ — заданный вектор терминального состояния, $\mathbf{x}^f = \left[x_1^f \dots x_n^f\right]^T$, t_f — ограниченное время процесса управления

$$t_f = \begin{cases} t, \text{ если } t < t_{max} \text{ и } \left\|\mathbf{x}(t) - \mathbf{x}^f\right\| \leqslant \varepsilon \\ t_{max} \text{ — иначе} \end{cases},$$

где t_{max} — ограничение на время процесса управления, $\varepsilon > 0$ — необходимая точность достижения терминального состояния. Норма разности векторов может выбираться из особенностей задачи. Для модели объекта с соизмеримыми

компонентами вектора состояния целесообразно выбрать Евклидову норму

$$\left\|\mathbf{x}\left(t\right)-\mathbf{x}^{f}\right\|=\sqrt{\sum_{i=1}^{n}\left(x_{i}\left(t\right)-x_{i}^{f}\right)^{2}}.$$

В общем виде функционал качества для задачи перевода объекта управления из состояния $\mathbf{x}^0$ в состояние $\mathbf{x}^f$ будет выглядеть следующим образом

$$J=\int_{t_0}^{t_f}f_0\left(\mathbf{x},\mathbf{u}\right)dt\to\min. \tag{4.1}$$

Частным случаем рассматриваемой задачи является задача об оптимальном быстродействии, то есть задача поиска такого вектора управления $\mathbf{u}$, который обеспечит перевод объекта управления из состояния $\mathbf{x}^0$ в состояние $\mathbf{x}^f$ за минимальное время. В этом случае значение $f_0\left(\mathbf{x},\mathbf{u}\right)=1$, а значение функционала качества (4.1) принимает вид

$$J=t_f\to\min.$$

В прикладных задачах довольно часто приходится сталкиваться с ограничениями на состояние объекта управления. В простейшем случае можно говорить о статических фазовых ограничениях, информация о которых известна изначально

$$h_i\left(\mathbf{x}\right)\leqslant 0,\quad i=\overline{1,z},$$

где z — число фазовых ограничений.

Также в прикладных задачах встречаются динамические фазовые ограничения, которые вычисляются на каждом шаге поиска. Все фазовые ограничения также должны быть учтены при поиске оптимального управления. Тогда функционал качества с учётом фазовых ограничений и значения нормы разности векторов полученного и терминального состояний, представляющих собой ошибку управления, будет иметь вид

$$J=\mu\left\|\mathbf{x}(t_f)-\mathbf{x}^{f}\right\|+\int_{t_0}^{t_f}\left(f_0\left(\mathbf{x},\mathbf{u}\right)+\sum_{i=1}^{z}\alpha_i\vartheta\left(h_i(\mathbf{x})\right)h_i(\mathbf{x})\right)dt\to\min,$$

где μ — весовой коэффициент, $\vartheta\left(h_i\left(\mathbf{x}\right)\right)$ — функция Хэвисайда (1.22), α_i — заданные коэффициенты штрафа, $i=\overline{1,z}$.

Результатом решения задачи оптимального управления является вектор управления с компонентами в форме функций времени $\tilde{\mathbf{u}}(\cdot) = [\tilde{u}_1(\cdot) \dots \tilde{u}_m(\cdot)]^T$, поэтому данная задача относится к классу задач бесконечномерной оптимизации. Далее эта задача может быть решена одним из численных методов решения.

4.2 Численные методы решения задачи оптимального управления

Среди обширного класса численных методов решения задачи оптимального управления выделяют несколько существенно отличающихся друг от друга направлений [53].

Наиболее популярным и эффективным является прямой метод, заключающийся в сведении исходной задачи бесконечномерной оптимизации к задаче конечномерной оптимизации и применении методов нелинейного программирования. При реализации этого подхода применяются различные методы оптимизации, которые исследуются и совершенствуются по сей день.

Альтернативный подход к решению задачи оптимального управления связан с непрямыми методами, в которых с помощью принципа максимума Понтрягина исходная задача редуцируется к краевой.

4.2.1 Непрямые методы решения задачи оптимального управления

Решение задачи оптимального управления на основе принципа максимума Понтрягина

Непрямые методы решения задачи оптимального управления существенным образом основаны на результатах работ Л.С. Понтрягина [51], хорошо известных как «принцип максимума» [26].

При описании принципа максимума Понтрягина для решения задачи оптимального управления в литературе часто приводится аналогия с методами

исследования функций от многих переменных, при котором сначала производится отбор точек, удовлетворяющих необходимым условиям, а далее — их анализ на предмет выполнения достаточных условий [20]. В данном случае принцип максимума выполняет роль необходимого условия оптимальности.

Принцип максимума является обобщением классической вариационной задачи нахождения оптимального управления как функции от времени, обеспечивающего минимум функционалу качества (4.1). При этом поиск оптимального управления осуществляется при условии максимизации функции Гамильтона $H(\boldsymbol{\psi},\mathbf{x},\mathbf{u})$.

Описание принципа максимума Понтрягина приведено в разделе 1.2.2 диссертации.

Важно подчеркнуть, что, при переходе от исходной задачи оптимального управления к краевой задаче, число дифференциальных уравнений возрастает в два раза, а это, в свою очередь, значительно увеличивает сложность вычислений.

Решение задачи оптимального управления на основе метода динамического программирования

В научной литературе [11; 54] также рассматриваются численные методы решения задачи оптимального управления на основе метода динамического программирования. В основе такого подхода лежит принцип оптимальности Р. Беллмана [9].

Описание метода динамического программирования в терминологии теории управления приведено в разделе 1.2.1 диссертации.

Уравнение Беллмана (1.12) может быть использовано для построения вектора оптимального управления $\mathbf{u}$ и вектора оптимального фазового состояния $\mathbf{x}$. В работах В.Г. Болтянского [11; 51] было показано, что в принципе оптимальности Р. Беллмана и принципе максимума Л.С. Понтрягина используются одни и те же необходимые условия оптимальности, сформулированные по разному.

Однако практическое использование принципа максимума Понтрягина и принципа оптимальности Беллмана оказалось весьма затрудненным, ввиду

значительного увеличения вычислительной сложности с ростом размерности системы управления при решении прикладных задач. Этим обусловлена бóльшая популярность прямых методов решения задачи оптимального управления.

4.2.2 Прямые методы решения задачи оптимального управления

Прямые методы решения задачи оптимального управления основаны на идее перехода от исходной задачи к задаче нелинейного программирования и позволяют использовать обширный набор разнообразных методов безусловной оптимизации [18; 25; 59; 76]. Переход от исходной задачи осуществляется путём дискретизации и вводом некоторой вспомогательной функции от многих переменных, изменяемой от шага к шагу [13; 26].

Решением задачи оптимального управления является вектор управления с компонентами в форме кусочно-непрерывных функций времени $\tilde{\mathbf{u}}(\cdot)$, поэтому данная задача относится к классу задач бесконечномерной оптимизации. Для поиска решения задачи оптимального управления прямым подходом с помощью методов нелинейного программирования необходимо аппроксимировать искомые компоненты вектора управления $\tilde{u}_i(\cdot)$, $i = \overline{1,m}$ функциональными зависимостями от конечного числа параметров. Для этой цели часто используют полиномы, ортогональные ряды или кусочно-функциональную аппроксимацию [61]. Эксперименты показали, что среди перечисленных видов аппроксимаций для исследуемой в работе прикладной задачи лучший результат дала кусочно-линейная аппроксимация.

Рассмотрим редукцию задачи оптимального управления к задаче нелинейного программирования с помощью кусочно-линейной аппроксимации.

Задаётся малый интервал $\Delta t > 0$, и определяется количество интервалов

$$M = \left\lceil \frac{t_{max}}{\Delta t} \right\rceil, \tag{4.2}$$

где t_{max} — ограничение на время управления.

Значение управления $\tilde{\mathbf{u}}(t) = [\tilde{u}_1(t) \dots \tilde{u}_m(t)]^T$ в момент времени t будет определяться из соотношения

$$\tilde{u}_j(t) = \begin{cases} u_j^-, \text{ если } q(t,j,i,\Delta t) < u_j^- \\ u_j^+, \text{ если } q(t,j,i,\Delta t) > u_j^+ \\ q(t,j,i,\Delta t) \text{ — иначе} \end{cases},$$

где $q(t,j,i,\Delta t) = q_{(j-1)M+i} + \left(q_{(j-1)M+i+1} - q_{(j-1)M+i}\right) \dfrac{(t-(i-1)\Delta t)}{\Delta t}$, $i\Delta t \leqslant t < (i+1)\Delta t$, $i = \overline{1,M}$, $j = \overline{1,m}$.

В результате поиск управления можно заменить поиском вектора постоянных параметров $\mathbf{q} = [q_1 \dots q_p]^T$, $\mathbf{q} \in \mathbb{R}^p$, где $p = m(M+1)$.

При поиске значения параметров следует ограничить

$$q_i^- \leqslant q_i \leqslant q_i^+, \quad i = \overline{1,p},$$

где q_i^-, q_i^+ — заданные значения ограничений на параметры, $u_j^- \geqslant q_i^-$, $u_j^+ \leqslant q_i^+$, $j = \overline{1,m}$, $i = \overline{(j-1)(M+1)+1, j(M+1)}$. Ограничение на управление не совпадает с ограничениями на значения параметров с целью увеличения Липшицевости искомой функции. Постоянная Липшица определяется по формуле $\left(q_i^+ - q_i^-\right)/\Delta t$.

Поиск вектора постоянных параметров $\mathbf{q}$, являющегося решением редуцированной задачи оптимального управления, реализуется одним из многочисленных методов безусловной оптимизации. От эффективности и сходимости выбранного метода в условиях отсутствия сведений о топологических свойствах целевой функции во многом зависит результат решения поставленной задачи.

4.3 Методы безусловной оптимизации для решения задачи оптимального управления

В настоящее время разработано и исследовано множество разнообразных численных методов для задач безусловной оптимизации [56]. К классическим методам принято относить класс хорошо изученных методов, которые принято разделять на методы нулевого порядка, методы первого порядка, требующие вычисления первой производной, и методы второго порядка, требующие вычисления второй производной.

Среди классических методов нулевого порядка можно выделить метод Нелдера-Мида, метод Хука-Дживса, метод Розенброка [56]. Также к методам нулевого порядка относится обширный класс методов случайного поиска [63]. Методы нулевого порядка хорошо зарекомендовали себя для решения простых задач оптимизации малой размерности, однако не используются на практике для решения задач оптимального управления.

Среди методов первого порядка, требующих вычисления первой производной на каждом шаге поиска, можно выделить метод градиентного спуска с постоянным шагом, метод наискорейшего градиентного спуска, метод покоординатного градиентного спуска, метод Гаусса-Зейделя, метод Флетчера-Ривса [56]. Также заслуживает внимания современный метод Adam [113]. В методах первого порядка значение градиента используется для вычисления направления поиска на каждом шаге алгоритма. Благодаря использованию значения градиента функции в процессе поиска, эффективность методов первого порядка намного выше чем у методов нулевого порядка, но одновременно с этим появляются определенные ограничения на класс минимизируемых функций ввиду необходимости их дифференцируемости на всем множестве поиска.

Среди методов второго порядка, в которых на каждом шаге алгоритма требуется вычисление второй производной, можно выделить хорошо известный метод Ньютона и его модификацию — метод Ньютона-Рафсона, а также метод Марквардта [56; 72]. Методы второго порядка эффективнее и работают намного быстрее методов первого порядка, но накладывают ещё бóльшие ограничения на минимизируемые функции, так как требуют существования второй производной на всем пространстве поиска.

В развитии современных методов безусловной оптимизации можно выделить два направления. Одно направление развития основано на использовании значения градиента для поиска решения на каждом шаге. Такие методы можно отнести к классу градиентных методов. Эти методы нашли широкое применение в задачах обучения нейронных сетей.

Второе направление развития современных методов безусловной оптимизации связано с появлением в 80-х годах прошлого века и последующим интенсивным развитием класса стохастических алгоритмов. Данные алгоритмы основаны на принципах одновременного учёта множества возможных решений оптимизационной задачи и эволюционирования этих решений на каждом шаге алгоритма с целью получения лучшего решения в соответствии с критериями

поиска. Данный класс методов получил название эволюционных или популяционных методов. Первым и самым известным методом данного класса является генетический алгоритм, предложенный Джоном Холландом в 1975 году [106]. Среди широко известных эволюционных алгоритмов также стоит упомянуть метод роя частиц, предложенный в 1995 году и эффективно использующийся по сей момент [112]. Класс эволюционных алгоритмов активно развивается и каждый год пополняется новыми методами.

С бурным ростом числа разнообразных оптимизационных алгоритмов, стала возникать задача их сравнительного анализа с целью оценки их эффективности и производительности. Точный и полный анализ алгоритмов может служить хорошим помощником в выборе определенного метода для решения конкретной прикладной задачи. Число научных работ, посвященных сравнительному анализу оптимизационных методов довольно велико. Так в 60-70-х годах прошлого века большое внимание уделялось выявлению наиболее эффективных подходов среди методов нулевого порядка и градиентных методов [87; 141]. А в настоящее время внимание ученых направлено на поиск универсальных и эффективных методов среди современных метаэвристических алгоритмов, включая эволюционные алгоритмы [88; 127; 135].

Несмотря на то, что для подавляющего большинства оптимизационных алгоритмов существует значительное число сравнительных исследований их эффективности, эти исследования в основном выполнялись с использованием, тестовых функций [108; 137; 148]. Использование тестовых функций позволяет оценить эффективность того или иного алгоритма в условиях мультимодальности и недифференцируемости оптимизируемого функционала. Однако в полной мере сделать вывод о преимуществе определённого алгоритма для решения прикладной задачи по результатам его работы с тестовыми функциями всё-таки невозможно.

Рассматривая широкий перечень разнообразных методов безусловной оптимизации, необходимо учитывать их эффективность именно для решения задачи оптимального управления.

4.3.1 Градиентные методы

К классу градиентных методов относятся как методы первого порядка, так и методы второго порядка. В подавляющем большинстве методов данного класса используется траекторный подход к организации поиска оптимального значения, то есть градиентные методы относятся к классу методов детерминированного поиска. При таком подходе для заданной или выбранной случайно начальной точки с помощью вычислений на основе значения градиента функции строится траектория перемещения в точку искомого оптимального решения.

Градиентные методы наиболее часто предлагают для решения задачи нелинейного программирования. Методы данного класса больше других подвергались теоретическим исследованиям [5; 17; 31]. Приведем основные характеристики наиболее известных градиентных методов: метода наискорейшего градиентного спуска; метода Ньютона-Рафсона; метода Марквардта; метода Adam.

Метод наискорейшего градиентного спуска включает поиск направления по антиградиенту целевой функции в одной точке и одномерный поиск в направлении антиградиента, как правило, прямым методом, например, методом золотого сечения [56]. Использование прямых методов одномерной оптимизации обусловлено тем, что в задачах оптимального управления отсутствует явный вид функциональной зависимости между целевой функцией и компонентами искомых параметров, а, следовательно, возможность использования производной целевой функции по параметрам отсутствует. Данный факт также говорит о слабости классических градиентных методов при их применении для задач оптимального управления, ввиду их использования без обоснования свойств унимодальности целевого функционала. Несмотря на это, в работе Э.Б. Ли и Л. Маркуса по оптимальному управлению [47] метод наискорейшего спуска вставлен в приложение, как один из методов решения задачи оптимального управления.

Метод Ньютона-Рафсона относится к градиентным методам 2-го порядка и является модификацией классического метода Ньютона [56; 72]. Для вычислений используется информация о частных производных целевой функции первого и второго порядка. Методы 2-го порядка и метод Ньютона-Рафсона

в частности обладают более высокой скоростью сходимости, так как позволяют вычислить величину шага в сторону антиградиента по обратной матрице вторых частных производных Гессе. При этом для вычислений необходимо соблюдение условий положительной определенности матрицы Гессе. В тех итерациях, где матрица Гессе не является положительно определенной, вычисления производятся по алгоритму наискорейшего градиентного спуска.

Метод Марквардта также относится к градиентным методам 2-го порядка [56]. Но, в отличие от метода Ньютона-Рафсона, в данном методе для вычислений не требуется положительная определенность матрицы Гессе. На каждой итерации при вычислении нового возможного решения используется специальная переменная μ. При уменьшении значения μ метод Марквардта по своим свойствам приближается к методу Ньютона, а при увеличении μ — к градиентному спуску.

Новый градиентный метод Adam (сокр. от англ. Adaptive Moment Estimation) относится к градиентным методам 1-го порядка. Он был предложен в 2015 году [113] для решения задачи обучения больших искусственных нейронных сетей. Концепция метода требует знания только градиента функции, что положительно сказывается на скорости работы алгоритма и используемой памяти.

4.3.2 Методы случайного поиска

Одним из первых советских учёных, исследовавших и популяризовавших методы случайного поиска, был Л.А. Растригин, который в своих работах [63; 64] представил подробное описание и анализ эффективности нескольких методов из данного класса. Главными преимуществами методов случайного поиска перед классическими градиентными методами считается их быстродействие и простота реализации, малочувствительность к росту размерности оптимизируемой задачи и их применимость к многопараметрическим мультимодальным задачам. В задачах, в которых оптимизируемый функционал обладает свойствами дифференцируемости и унимодальности, методы случайного поиска ожидаемо проигрывают градиентным методам.

Самым простым, но одновременно и самым известным методом из класса методов случайного поиска является метод Монте-Карло [62]. Суть метода в выборе лучшего решения из множества случайно распределенных на пространстве поиска точек. Очевидно, что данный метод не является эффективным для решения задач оптимального управления, хотя, при устремлении размера множества случайных точек к бесконечности, позволит получить глобально оптимальное решение. Современные методы случайного поиска позволяют эффективно находить решение оптимизационных задач за конечное время. Среди наиболее известных и эффективных методов данного класса можно выделить адаптивный метод случайного поиска [56; 63], метод статистического градиента [63], метод случайного поиска с направляющим гиперквадратом [8] и метод наилучшей пробы для многоэкстремальных задач [56; 63].

Адаптивный метод случайного поиска относится к типу направленного случайного поиска [56; 63]. В отличие от ненаправленного случайного поиска, в котором все случайные решения генерируются независимо от результатов предыдущих решений, в направленном случайном поиске полученное на предыдущей поисковой итерации решение используется при формировании нового решения на текущей поисковой итерации. Такой подход позволяет обеспечить более высокую сходимость метода, но в тоже время найденные решения могут оказаться локальными экстремумами.

В методе статистического градиента на каждой итерации берется заданное число m независимых решений на гиперсфере с центром в текущей точке [63]. Для каждого решения вычисляется приращение значения функционала качества. При $m \to \infty$ направление, на котором достигается максимальное уменьшение значения функционала качества, совпадает с направлением антиградиента.

В методе случайного поиска с направляющим гиперквадратом строится гиперквадрат, на начальном этапе совпадающий с допустимой областью поиска [8]. Внутри гиперквадрата берется заданное число m независимых решений. Лучшее из этих решений становится центром нового гиперквадрата меньшего размера.

В методе наилучшей пробы для многоэкстремальных задач из случайно заданного на допустимой области решения осуществляется поиск локального минимума с заданной точностью [56; 63]. Далее берется новое случайное решение и по той же схеме осуществляется поиск, результатом которого может быть

точка локального минимума со значением функции либо больше, чем в предыдущей точке, либо меньше. В последнем случае, новая точка берется за текущее решение, а поиск новых локальных минимумов продолжается до достижения заданного числа итераций.

4.3.3 Эволюционные алгоритмы

В конце XX века появились и стали активно развиваться эволюционные алгоритмы — методы глобальной безусловной оптимизации, относящиеся к классу метаэвристических методов [32; 71]. Приспособленность живых организмов к выживанию и размножению в разных условиях их жизнедеятельности являлась вдохновением для разработчиков при создании алгоритмов, имитирующих соответствующие процессы. Отсюда происходит название класса эволюционных алгоритмов, для которых также известны и другие названия: поведенческие, популяционные или вдохновленные природой [33].

Первым из эволюционных алгоритмов следует считать генетический алгоритм, появившийся в начале 70-х годов прошлого века [106]. Спустя два десятилетия развитие эволюционных алгоритмов вышло на новый виток. Этому способствовала значительно возросшая вычислительная мощность ЭВМ.

В настоящее время известно большое число эффективных и хорошо зарекомендовавших себя алгоритмов данного класса [33]. Среди них можно выделить метод роя частиц [112], предложенный в 1995 году, пчелиный алгоритм [142], алгоритм муравьиной колонии [97], метод дифференциальной эволюции [139], метод инспирированный летучими мышами [147], алгоритм светлячков [149], кукушкин поиск [152], метод серых волков [126].

Следует сказать, что экзотические названия новых поисковых алгоритмов скорее препятствуют их широкому распространению и исследованию в среде прикладных математиков и инженеров-вычислителей. Сложность оценки близости выполняемых операций в данных алгоритмах к реальным биологическим процессам в организованных живых системах, именами которых эти алгоритмы названы, может вызывать недоверие к результатам поиска. Определенное недоверие также может вызывать тот факт, что все эволюционные алгоритмы относятся к классу эвристических недерминированных методов, что означает

отсутствие доказательства их сходимости. Видимо, этим обусловлено большое число научных работ, посвященных экспериментальному доказательству их эффективности [33; 148; 151].

Все эволюционные алгоритмы имеют общие признаки. У эволюционных алгоритмов для поиска используется множество возможных решений вместо одного решения как при классическом траекторном подходе в алгоритмах детерминированного поиска. Данное множество возможных решений в разных публикациях называется роем, стаей, популяцией и т.п. Начальное множество возможных решений в большинстве случаев генерируется случайно. Далее выполняется оценка всех возможных решений по значению целевой функции. После этого на основе полученных оценок выполняется преобразование элементов множества возможных решений с целью улучшения оценок этих элементов. Часто вместо возможных решений с плохими оценками генерируются новые возможные решения. Преобразование множества возможных решений на основе анализа оценок существующих решений называется эволюцией. Процессы генерации возможных решений, их оценивание и преобразование повторяются конечное число итераций. Решение с наилучшей оценкой в последнем множестве возможных решений считается решением задачи оптимизации.

Процедура эволюции множества возможных решений состоит из двух этапов — исследовательского поиска и локального поиска [151]. Данные этапы можно рассматривать как последовательные процессы диверсификации и интенсификации поиска.

Первый этап — исследовательский поиск. На данном этапе алгоритм проводит исследование всей области поиска, генерируя разнообразные возможные решения на значительном расстоянии друг от друга. Эта часть поиска является глобальной в части исследования области возможных решений. Она позволяет значительно уменьшить вероятность застревания в локальных оптимумах и увеличить вероятность нахождения глобального оптимума.

Второй этап — локальный поиск. На данном этапе известная на текущей момент информация об области поиска используется для поиска новых решений, которые будут лучше чем текущие решения. Эта часть поиска хоть и подвержена застреванию в локальных оптимумах, но позволяет обеспечить высокую сходимость алгоритма.

Алгоритмы, в которых исследовательский поиск преобладает над локальным, могут находить решения в области глобального оптимума, но не

обеспечивать необходимого уровня точности. И наоборот, алгоритмы, в которых локальный поиск преобладает над исследовательским, могут обеспечивать высокую точность, но застревать в локальных оптимумах. Таким образом, для обеспечения высокой эффективности алгоритма, необходимо соблюсти баланс между исследовательским и локальным поисками. Нахождение такого баланса до сих пор является отдельной актуальной задачей среди исследователей эволюционных алгоритмов. На выбор соотношения между исследовательским и локальным поиском могут влиять настройки параметров алгоритма, механизм работы самого алгоритма и тип оптимизируемой задачи.

Эволюционная стратегия описывает механизмы модификации всего множества возможных решений в целом, но свойства отдельно взятого элемента этого множества также представляют интерес. Каждое решение из множества возможных решений на каждой поисковой итерации обладает следующими свойствами [33]:

1. Автономность — отдельные решения хотя бы частично независимы друг от друга;
2. Стохастичность — эволюционные преобразования решений множества содержат случайную составляющую;
3. Ограниченность — отдельно взятые решения не предоставляют информации об исследуемой задаче в целом;
4. Децентрализованность — процесс поиска основан на информации о множестве решений, а не одного, пусть даже и лучшего, решения;
5. Коммуникабельность — эволюционные преобразования отдельного решения могут быть частично основаны на информации, полученной от других решений множества.

Отличие эволюционных алгоритмов между собой заключается в различии преобразований возможных решений на этапе эволюции. Следует особо отметить, что, несмотря на то, что эволюционные алгоритмы относятся к классу стохастических методов, случайная составляющая не играет главенствующую роль. При эволюционных преобразованиях основная роль отводится информации об оценках всего множества возможных решений. Случайная составляющая в свою очередь обеспечивает диверсификацию поиска для более высокой вероятности нахождения глобального оптимума решаемой задачи.

В генетическом алгоритме эволюция выполняется с помощью операций скрещивания и мутации кодов возможных решений. Вероятность выполнения

операции скрещивания зависит от близости значений оценок отобранных для скрещивания возможных решений к наилучшей текущей оценке. Полученные после эволюционных преобразований новые возможные решения добавляются к множеству, если их оценки не хуже наихудшей оценки во множестве. В наиболее популярном сегодня алгоритме роя частиц эволюционные преобразования возможного решения выполняются с учетом наилучшего найденного к текущему моменту возможного решения и наилучшего среди некоторого подмножества возможных решений, называемого множеством информаторов.

Обычно эволюционные алгоритмы имеют некоторое число свободных параметров. Решение той или иной прикладной оптимизационной задачи зависит как от правильного выбора значений свободных параметров, так и от правильного выбора используемого метода в целом.

Далее в диссертации рассматриваются следующие классические и современные эволюционные алгоритмы в терминологии, принятой для описания задач оптимального управления: генетический алгоритм [106], метод дифференциальной эволюции [139], метод роя частиц [112], пчелиный алгоритм [142], алгоритм летучих мышей [147], алгоритм серых волков [126].

Во всех эволюционных алгоритмах поиск осуществляется на множестве возможных решений заданного размера H, в котором выполняются определенные эволюционные преобразования заданное W число раз, соответствующее количеству поисковых итераций. Решением поставленной задачи будет наилучшее по значению целевой функции возможное решение в итоговом множестве.

Генетический алгоритм

Начало 70-х годов XX века можно назвать отправной точкой развития эволюционных алгоритмов. В это время американский учёный Джон Холланд с помощниками начали применять операции скрещивания и рекомбинации для исследования адаптивных систем. Результатом данных исследований стало изобретение генетического алгоритма, который был опубликован в работе Дж. Холланда в 1975 году [106]. Главной особенностью алгоритма было то, что он был представлен как математическая абстракция Дарвинской теории эволюции и естественного отбора в природе, реализованная с помощью мате-

матических операторов скрещивания, рекомбинации, мутации и отбора [103]. Генетический алгоритм показал высокую эффективность в решении широкого спектра задач, в том числе сложноформализуемых, мультимодальных и недифференцируемых.

Основу генетического алгоритма составляют три операции: скрещивание, мутация и отбор. Операция скрещивания выполняет роль локальной части поиска, ее задача — улучшение сходимости алгоритма. Операция мутации представляет исследовательскую часть поиска и позволяет увеличить шанс нахождения глобального оптимума в ущерб скорости сходимости алгоритма.

Рассмотрим генетический алгоритм пошагово.

В генетическом алгоритме каждая компонента вектора возможного решения $\mathbf{q}$ кодируется двоичным кодом Грея. Для кодирования одной компоненты решения выделяем C бит под целую часть числа и D бит — под дробную часть. В результате код каждой компоненты вектора возможного решения представляет собой положительное целое число g_i, $i = \overline{1,p}$, в диапазоне от 0 до 2^{C+D}. Всего код одного вектора из p компонент содержит $p\,(C+D)$ бит.

На начальном этапе генерируем множество возможных решений из H случайных двоичных кодов. При кодировании используются коды Грея

$$\mathbf{S} = \{\mathrm{S}_1, \ldots, \mathrm{S}_H\},$$

где $\mathrm{S}_i = \left(s_1^i, \ldots, s_{p(C+D)}^i\right)$, $s_j^i \in \{0,1\}$, $i = \overline{1,H}$, $j = \overline{1,\, p\,(C+D)}$.

Запускаем процесс поиска. Для каждого возможного решения производим вычисление значения функционала. Для этого преобразовываем код Грея возможного решения $\mathrm{S}_i = \left(s_1^i, \ldots, s_{p(C+D)}^i\right)$ в двоичный код $\mathrm{B}_i = \left(b_1^i, \ldots, b_{p(C+D)}^i\right)$, где

$$b_j^i = \begin{cases} s_j^i, \text{ если } (j-1) \bmod (C+D) = 0 \\ s_j^i \oplus s_{j-1}^i \text{ — иначе} \end{cases}, \quad j = \overline{1,\, p\,(C+D)}.$$

Полученный двоичный код переводим в десятичный

$$g_k^i = \sum_{l=1}^{C+D} b_{(C+D)(k-1)+l}^i 2^{C-l}, \quad k = \overline{1,p}.$$

Вычисляем значения компонент вектора возможного решения $\mathbf{q}^i$ по формуле

$$q_k^i = \frac{g_k^i}{2^C}\left(q_k^+ - q_k^-\right) + q_k^-, \quad k = \overline{1,p}.$$

Формируем множество оценок возможных решений по значению функционала

$$\mathrm{F} = \left\{ f_1 = J\left(\mathbf{q}^1\right), \ldots, f_H = J\left(\mathbf{q}^H\right) \right\}.$$

Выполняем одноточечное скрещивание. Случайно отбираем два кода возможных решений

$$\mathrm{S}_\alpha = \left(s_1^\alpha, \ldots, s_{p(C+D)}^\alpha \right), \quad \mathrm{S}_\beta = \left(s_1^\beta, \ldots, s_{p(C+D)}^\beta \right), \quad \alpha,\beta \in \left\{ \overline{1,H} \right\}.$$

Вычисляем вероятность скрещивания по формуле

$$P_c = \max \left\{ \frac{f^-}{f_\alpha}, \frac{f^-}{f_\beta} \right\},$$

где $f^- = \min \{ f_1, \ldots, f_H \}$.

Процедура скрещивания выполняется если $\xi \leqslant P_c$, где ξ — случайное число, $\xi \in [0; 1]$. Точка скрещивания σ определяется случайно, $\sigma \in \left\{ \overline{1,\, p\,(C+D)} \right\}$. Части кодов отобранных возможных решений правее точки скрещивания обмениваются. В результате получаем коды двух новых возможных решений.

$$\mathrm{S}_{H+1} = \left(s_1^\alpha, \ldots, s_{\sigma-1}^\alpha, s_\sigma^\beta, \ldots, s_{p(C+D)}^\beta \right),$$

$$\mathrm{S}_{H+2} = \left(s_1^\beta, \ldots, s_{\sigma-1}^\beta, s_\sigma^\alpha, \ldots, s_{p(C+D)}^\alpha \right).$$

Операция мутации выполняется с заданной величиной вероятности P_m. В коде нового возможного решения S_{H+1} случайным образом определяется точка мутации $\mu \in \left\{ \overline{1,\, p\,(C+D)} \right\}$ и производится инверсия компоненты $s_\mu^{H+1} \in \{0,1\}$. Те же действия повторяем для возможного решения S_{H+2}.

Производим оценку полученных новых возможных решений по значению минимизируемого функционала

$$f_{H+1} = J\left(\mathbf{q}^{H+1}\right), \quad f_{H+2} = J\left(\mathbf{q}^{H+2}\right).$$

Если значение оценки нового возможного решения лучше наихудшей оценки решения из множества всех возможных решений, то наихудшее решение из множества заменяем на новое возможное решение

$$\mathrm{S}_w = \mathrm{S}_{H+i},\ f_w = f_{H+i},\ \text{если}\ f_w > f_{H+i},$$

где $f_w = \max \{ f_1, \ldots, f_H \}$, $i = 1,2$.

Процесс поиска повторяется на протяжении заданного числа итераций W. В качестве итогового решения выбираем наилучшее возможное решение во множестве.

Алгоритм дифференциальной эволюции

Алгоритм дифференциальной эволюции впервые был представлен в 1995 году [101; 139]. В алгоритме используется адаптивная мутация векторов параметров, которая зависит от распределения решения в пространстве поиска. При сильном распределении решений, алгоритм производит существенные вариации, а при малом распределении производятся лишь небольшие изменения векторов параметров во множестве возможных решений.

Рассмотрим алгоритм дифференциальной эволюции пошагово.

Генерируем множество Q возможных решений

$$\mathrm{Q} = \left(\mathbf{q}^1, \ldots, \mathbf{q}^H\right),$$

где $\mathbf{q}^j$ — вектор параметров, вычисленный по формуле

$$q_i^j = \xi\left(q_i^+ - q_i^-\right) + q_i^-, \quad i = \overline{1,p}, \;\; j = \overline{1,H}, \tag{4.3}$$

ξ — случайная равномерно распределенная величина в диапазоне $[0;1]$.

Определяем множество значений целевых функций для каждого возможного решения

$$\mathrm{F} = \left(f_1 = J\left(\mathbf{q}^1\right), \ldots, f_H = J\left(\mathbf{q}^H\right)\right).$$

Для выполнения эволюции множества возможных решений задаем параметр K, который определяет количество модификаций векторов во множестве. Для модификации с заданной величиной вероятности P_m отбираем случайно вектор $\mathbf{q}^j \in Q$, $1 \leqslant j \leqslant H$, выбираем случайно три различных вектора $\mathbf{q}^a$, $\mathbf{q}^b$, $\mathbf{q}^c$, $a \neq b \neq c$, отбираем случайно компоненту μ, $\mu \in \left\{\overline{1,p}\right\}$.

Вычисляем новое значение компоненты по формуле

$$\tilde{q}_\mu = \begin{cases} \xi_1\left(q_\mu^+ - q_\mu^-\right) + q_\mu^-, \text{ если } q_\mu^a + \xi_2\left(q_\mu^b - q_\mu^c\right) > q_\mu^+ \\ \xi_1\left(q_\mu^+ - q_\mu^-\right) + q_\mu^-, \text{ если } q_\mu^a + \xi_2\left(q_\mu^b - q_\mu^c\right) < q_\mu^- \\ q_\mu^a + \xi_2\left(q_\mu^b - q_\mu^c\right) \text{ — иначе} \end{cases},$$

где ξ_1 и ξ_2 — случайные равномерно распределенные величины в диапазоне $[0;1]$.

Формируем новый вектор $\tilde{\mathbf{q}} = \left[q_1^j \dots \tilde{q}_\mu \dots q_p^j\right]^T$ и вычисляем для него значение целевой функции $J(\tilde{\mathbf{q}})$. Если это значение меньше, чем значение целевой функции для первоначального отобранного вектора $\mathbf{q}^j$, $J(\tilde{\mathbf{q}}) < f_j = J\left(\mathbf{q}^j\right)$, то заменяем отобранный вектор новым вектором, $\mathbf{q}^j = \tilde{\mathbf{q}}$, $f_j = J(\tilde{\mathbf{q}})$.

Повторяем эволюции с модификациями K отобранных векторов заданное W количество раз. Решением считаем наилучший вектор в итоговом множестве возможных решений.

Метод роя частиц

Метод роя частиц на текущий момент является одним из наиболее популярных эволюционных алгоритмов. Данный метод является первым алгоритмом, основанным на имитации поведенческой модели самоорганизующихся живых систем — так называемом роевом интеллекте [99]. Канонический метод роя частиц был предложен в 1995 году [112]. Появление этого метода дало новый толчок к более интенсивному развитию эволюционных алгоритмов, а сам метод роя частиц получил широкое развитие в многочисленных научных работах, посвященных исследованию и модификации исходного алгоритма [34].

Модификация всего множества возможных решений производится с учетом некоторого числа лучших решений на текущей итерации, называемых информаторами и выбираемых из числа всех возможных решений или некоторого подмножества.

Рассмотрим метод роя частиц пошагово.

На начальном этапе задаем размер множества возможных решений H, размер подмножества информаторов N и число поисковых итераций W.

Генерируем начальное множество векторов возможных решений $\mathbf{q}^j$, $j = \overline{1,H}$, по формуле (4.3).

Задаем начальные векторы $\mathbf{v}^j$ направления изменения возможных решений

$$v_i^j = 0, \quad i = \overline{1,p}, \ j = \overline{1,H}.$$

Процесс поиска оптимального решения продолжаем до достижения максимального числа итераций W. На каждой итерации производим вычисление

вектора $\mathbf{q}^b$, доставляющего лучшее значение функционала

$$J\left(\mathbf{q}^b\right) = \min_j \left\{ J\left(\mathbf{q}^j\right) : j = \overline{1,H} \right\},$$

и вычисление опорного вектора $\mathbf{q}^{j_r}$, являющегося лучшим среди N случайно отобранных векторов $\mathbf{q}^{j_1}, \ldots, \mathbf{q}^{j_N}$

$$J\left(\mathbf{q}^{j_r}\right) = \min_k \left\{ J\left(\mathbf{q}^{j_k}\right) : k = \overline{1,N} \right\},$$

где $j_1, \ldots, j_N$ — случайные целые числа в диапазоне $[1; H]$.

Для каждого возможного решения $\mathbf{q}^j$ вычисляем новое значение вектора $\mathbf{v}^j$ с учётом найденных лучшего $\mathbf{q}^b$ и опорного $\mathbf{q}^{j_r}$ возможных решений

$$v_i^j \leftarrow \alpha v_i^j + \xi_\beta \left(q_i^b - q_i^j\right) + \xi_\gamma \left(q_i^{j_r} - q_i^j\right), \quad i = \overline{1,p},\ j = \overline{1,H},$$

где $\xi_\beta \in [0; \beta]$ и $\xi_\gamma \in [0; \gamma]$ — случайные величины, α, β, γ — заданные свободные параметры алгоритма, значение которых подбирают в зависимости от задачи. Далее вычисляем новое возможное решение $\tilde{\mathbf{q}}^j$ по формуле

$$\tilde{q}_i^j = \begin{cases} q_i^-, \text{ если } q_i^j + \delta v_i^j < q_i^- \\ q_i^+, \text{ если } q_i^j + \delta v_i^j > q_i^+ \\ q_i^j + \delta v_i^j \text{ — иначе} \end{cases}, \quad i = \overline{1,p},\ j = \overline{1,H},$$

где δ — заданный свободный параметр, $\delta \approx 1$.

Если $J\left(\mathbf{q}^j\right) > J\left(\tilde{\mathbf{q}}^j\right)$, то заменяем решение $\mathbf{q}^j$ во множестве возможных решений новым возможным решением $\tilde{\mathbf{q}}^j$, $\mathbf{q}^j \leftarrow \tilde{\mathbf{q}}^j$.

Завершаем вычисления при выполнении W итераций. В качестве решения задачи выбираем вектор $\mathbf{q}$ из итогового множества возможных решений, доставляющий наилучшее значение целевой функции.

Пчелиный алгоритм

Пчелиный алгоритм был предложен в 2005 году [19; 142]. Как следует из названия метода, на создание алгоритма авторов вдохновило поведение медоносных пчёл в дикой природе.

В пчелином алгоритме первоначально выявляется определенное число подобластей пространства поиска, в которых значение целевой функции меньше. Данные подобласти исследуются более интенсивно, а их радиус уменьшается с каждой итерацией.

Рассмотрим пчелиный алгоритм пошагово.

Задаем параметры алгоритма — размер множества возможных решений H, максимальное число итераций W, количество N элитных возможных решений, $N \approx 0{,}2H$, и количество L перспективных возможных решений, $L \approx 0{,}5H$, число E новых векторов в области элитного возможного решения, $E \approx H$, и число S новых векторов в области перспективного возможного решения, $S \approx 0{,}5H$.

Задаем начальные значения векторов радиусов областей поиска $\mathbf{r}^e$ и $\mathbf{r}^s$, $r_i^e > 0$, $r_i^s > r_i^e$, $i = \overline{1,p}$.

Генерируем множество векторов параметров $\mathbf{q}^j$, $j = \overline{1,H}$, по формуле (4.3).

На каждой итерации для всех возможных решений вычисляем значения функционала $J\left(\mathbf{q}^j\right)$, $j = \overline{1,H}$, и упорядочиваем возможные решения во множестве по возрастанию значения целевой функции

$$J\left(\mathbf{q}^1\right) \leqslant J\left(\mathbf{q}^2\right) \leqslant \ldots \leqslant J\left(\mathbf{q}^H\right). \tag{4.4}$$

Отбираем первые N элитных векторов $\mathbf{q}^j$, $j = \overline{1,N}$. Для каждого элитного вектора $\mathbf{q}^j$ строим набор из E векторов $\mathbf{g}^e$

$$g_i^e = \begin{cases} q_i^-, \text{ если } q_i^j + \xi_i^e < q_i^- \\ q_i^+, \text{ если } q_i^j + \xi_i^e > q_i^+ \\ q_i^j + \xi_i^e \text{ — иначе} \end{cases}, \quad i = \overline{1,p},\ e = \overline{1,E},$$

где $\xi_i^e \in [-r_i^e; r_i^e]$ — величина, случайно распределенная на интервале от $-r_i^e$ до r_i^e, r^e — заданный вектор радиусов области поиска вокруг элитного вектора; E — число новых векторов $\mathbf{g}^e$ в области элитного вектора $\mathbf{q}^j$.

Если вектор $\mathbf{g}^e$ доставляет значение функционала лучше, чем возможное решение $\mathbf{q}^j$, то производим замену

$$\text{— если } J\left(\mathbf{g}^e\right) < J\left(\mathbf{q}^j\right), \text{ то } \mathbf{q}^j \leftarrow \mathbf{g}^e,\ J\left(\mathbf{q}^j\right) \leftarrow J\left(\mathbf{g}^e\right),\ e = \overline{1,E}.$$

Производим модификацию перспективных возможных решений. Для каждого $\mathbf{q}^j$, $j = \overline{N+1,\, L}$, строим набор из S векторов $\mathbf{g}^s$ по формуле

$$g_i^s = \begin{cases} q_i^-, \text{ если } q_i^j + \xi_i^s < q_i^- \\ q_i^+, \text{ если } q_i^j + \xi_i^s > q_i^+ \\ q_i^j + \xi_i^s \text{ — иначе} \end{cases}, \quad s = \overline{1,S},$$

где $\xi_i^s \in [-r_i^s; r_i^s]$ — величина, случайно распределенная на интервале от $-r_i^s$ до r_i^s, r^s — заданный вектор радиусов области поиска вокруг перспективного возможного решения.

Если вектор $\mathbf{g}^s$ доставляет значение функционала лучше, чем возможное решение $\mathbf{q}^j$, то производим замену

$$\text{— если } J(\mathbf{g}^s) < J\left(\mathbf{q}^j\right), \text{ то } \mathbf{q}^j \leftarrow \mathbf{g}^s,\ J\left(\mathbf{q}^j\right) \leftarrow J(\mathbf{g}^s),\ s = \overline{1,S}.$$

Для остальных возможных решений $\mathbf{q}^j$, $j = \overline{L+1,\, H}$, во множестве производим генерацию новых случайных значений по формуле (4.3).

Уменьшаем величины радиусов области поиска вокруг элитных и перспективных возможных решений

$$r_i^e \leftarrow \alpha_e r_i^e, \quad r_i^s \leftarrow \alpha_s r_i^s, \quad i = \overline{1,p},$$

где α_e, α_s — заданные коэффициенты уменьшения области поиска, $\alpha_e \approx 0{,}95$, $\alpha_s \approx 0{,}95$.

Процесс поиска и модификации возможных решений продолжаем до достижения максимального числа итераций W. В качестве решения задачи выбираем лучший вектор $\mathbf{q}$ из итогового множества возможных решений.

Алгоритм летучих мышей

Алгоритм летучих мышей был впервые представлен в 2010 году [147]. На идею создания алгоритма автора вдохновили летучие мыши, обладающие уникальными средствами эхолокации, которая используется для обеспечения полетов в темноте и обнаружения добычи.

Рассмотрим алгоритм летучих мышей пошагово.

Генерируем множество возможных решений $\mathbf{q}^j$, $j = \overline{1,H}$, по формуле (4.3).

Задаем начальные векторы $\mathbf{v}^j$ направления изменения возможных решений

$$v_i^j = 0, \quad i = \overline{1,p}, \ j = \overline{1,H}.$$

Задаем параметры алгоритма

$$a_j = 1, \ r_j = r_j^0, \ r_j^0 = \xi_j, \ j = \overline{1,H},$$

где ξ_j, $j = \overline{1,H}$, — случайные величины, равномерно распределенные на интервале $[0;1]$.

Задаем максимальное число итераций W.

На каждой итерации производим вычисление вектора $\mathbf{q}^b$, доставляющего наилучшее значение функционала

$$J\left(\mathbf{q}^b\right) = \min_k \left\{ J\left(\mathbf{q}^k\right) : k = \overline{1,H} \right\}.$$

Вычисляем значения параметра ω_j и вектора скоростей $\tilde{\mathbf{v}}^j$ для каждого вектора $\mathbf{q}^j$ во множестве возможных решений

$$\omega_j = \xi\left(\omega^+ - \omega^-\right) + \omega^-, \ \tilde{v}_i^j = v_i^j + \omega_j\left(q_i^j - q_i^b\right), \ i = \overline{1,p}, \ j = \overline{1,H},$$

где ξ — случайная равномерно распределенная величина на интервале $[0;1]$, ω^-, ω^+ — заданные неотрицательные минимальное и максимальное значения параметра ω_j, $\omega^- \approx 0$, $\omega^+ \approx 2$.

Далее для каждого возможного решения $\mathbf{q}^j$, $j = \overline{1,H}$, вычисляем новый вектор $\tilde{\mathbf{q}}^j$. Для этой цели генерируем случайную величину $\alpha \in [0;1]$.

Если $\alpha < r_j$, то

$$\tilde{q}_i^j = \begin{cases} q_i^-, \text{ если } q_i^j + \tilde{v}_i^j < q_i^- \\ q_i^+, \text{ если } q_i^j + \tilde{v}_i^j > q_i^+ \\ q_i^j + \tilde{v}_i^j \text{ — иначе} \end{cases}, \quad i = \overline{1,p}, \ j = \overline{1,H},$$

иначе

$$\tilde{q}_i^j = \begin{cases} q_i^-, \text{ если } q_i^b + \bar{a}\,(2\xi - 1) < q_i^- \\ q_i^+, \text{ если } q_i^b + \bar{a}\,(2\xi - 1) > q_i^+ \\ q_i^b + \bar{a}\,(2\xi - 1) \text{ — иначе} \end{cases}, \quad i = \overline{1,p}, \ j = \overline{1,H},$$

где $\bar{a} = \frac{1}{H}\sum\limits_{i=1}^{H} a_i$, ξ — случайная величина, равномерно распределенная на интервале $[0;1]$.

Если выполняется условие $J\left(\mathbf{q}^j\right) > J\left(\tilde{\mathbf{q}}^j\right)$, то производим замену

$$\mathbf{q}^j \leftarrow \tilde{\mathbf{q}}^j,\ \mathbf{v}^j \leftarrow \tilde{\mathbf{v}}^j,\ a_j \leftarrow \alpha a_j,\ r_j = r_j^0\left(1 - e^{-\gamma w}\right),\ j = \overline{1,H},$$

где w — номер текущей итерации, α, γ — заданные параметры, $\alpha \approx 0{,}9$, $\gamma \approx 0{,}9$.

Повторяем вычисления W раз. В качестве решения задачи выбираем лучший вектор $\mathbf{q}$ из итогового множества возможных решений.

Алгоритм серых волков

Алгоритм серых волков появился в 2014 году [126]. Суть алгоритма, по словам авторов, строится на имитации поведения стаи волков во время охоты на добычу. Алгоритм использует иерархию полученных возможных решений по значению функционала. Для модификации возможных решений алгоритм использует три наилучших возможных решения, найденных к текущему моменту.

Рассмотрим алгоритм серых волков пошагово.

Генерируем множество возможных решений $\mathbf{q}^j$, $j = \overline{1,H}$, по формуле (4.3).

Задаем максимальное число итераций W, устанавливаем текущее значение счетчика итераций $w = 0$.

Далее на каждой итерации производим вычисление трех лучших векторов $\mathbf{q}^\alpha$, $\mathbf{q}^\beta$, $\mathbf{q}^\delta$ таких, что

$$J\left(\mathbf{q}^\alpha\right) = \min_k \left\{J\left(\mathbf{q}^k\right) : k = \overline{1,H}\right\}, \tag{4.5}$$

$$J\left(\mathbf{q}^\beta\right) = \min_k \left\{J\left(\mathbf{q}^k\right) : k = \overline{1,H},\ k \neq \alpha\right\}, \tag{4.6}$$

$$J\left(\mathbf{q}^\delta\right) = \min_k \left\{J\left(\mathbf{q}^k\right) : k = \overline{1,H},\ k \neq \alpha,\ k \neq \beta\right\}. \tag{4.7}$$

Вычисляем параметр линеаризации

$$a = 2 - \frac{2w}{W}. \tag{4.8}$$

Для каждого вектора $\mathbf{q}^j$ вычисляем три дополнительных вектора $\boldsymbol{\alpha}^j$, $\boldsymbol{\beta}^j$, $\boldsymbol{\delta}^j$ по формулам

$$\alpha_i^j = q_i^\alpha - a\left(2\xi_1 - 1\right)\left|2\xi_2 q_i^\alpha - q_i^j\right|, \tag{4.9}$$

$$\beta_i^j = q_i^\beta - a\,(2\xi_3 - 1)\left|2\xi_4 q_i^\beta - q_i^j\right|, \tag{4.10}$$

$$\delta_i^j = q_i^\delta - a\,(2\xi_5 - 1)\left|2\xi_6 q_i^\delta - q_i^j\right|, \tag{4.11}$$

где $\xi_1, \ldots, \xi_6$ — случайно распределенные величины в диапазоне $[0; 1]$, $i = \overline{1,p}$.

На основе полученных векторов $\boldsymbol{\alpha}^j$, $\boldsymbol{\beta}^j$, $\boldsymbol{\delta}^j$ производим вычисление нового значения вектора $\mathbf{q}^j$

$$q_i^j = \begin{cases} q_i^-, \text{ если } \dfrac{1}{3}\left(\alpha_i^j + \beta_i^j + \delta_i^j\right) < q_i^- \\ q_i^+, \text{ если } \dfrac{1}{3}\left(\alpha_i^j + \beta_i^j + \delta_i^j\right) > q_i^+ \\ \dfrac{1}{3}\left(\alpha_i^j + \beta_i^j + \delta_i^j\right) \text{ — иначе} \end{cases}, \quad i = \overline{1,p}. \tag{4.12}$$

Процесс поиска продолжаем до достижения максимального числа итераций W. Решением задачи будет лучшее по значению функционала возможное решение $\mathbf{q}$ в итоговом множестве.

4.4 Сравнение эффективности эволюционных алгоритмов для решения задачи оптимального управления

В научной литературе можно найти большое число работ, посвященных сравнению и исследованию эффективности разнообразных эволюционных алгоритмов [33; 150; 151]. Классический подход к оценке эффективности поисковых алгоритмов основывается на тестовых функциях [108; 148]. Многие из существующих тестовых функций обладают свойствами большой размерности и мультимодальности функционала, но практический интерес в разрезе диссертационного исследования представляет оценка эффективности данных алгоритмов именно на прикладных задачах оптимального управления. В контексте применения метаэвристических эволюционных алгоритмов при решении задачи оптимального управления прямым подходом следует учитывать, что полученное решение будет приблизительно оптимальным. В отличии от тестовых задач, для которых известны оптимальные решения, в прикладной задаче оптимального управления информации об оптимальном решении у исследователя обычно не имеется. В этом случае сравнение всех результатов производится с лучшим решением из найденных сравниваемыми алгоритмами.

Оптимизируемый функционал в задаче оптимального управления, редуцированной к задаче нелинейного программирования, имеет высокую размерность и не обладает свойствами унимодальности и дифференцируемости на пространстве поиска. Несмотря на это, для решения таких задач обычно предлагались оптимизационные методы на основе градиентного спуска, как правило, второго порядка [27]. Отсюда возникает задача сравнения эволюционных алгоритмов с классическими градиентными методами безусловной оптимизации в контексте решения задачи оптимального управления.

Из анализа свойств эволюционных и градиентных алгоритмов следует, что первые будут более эффективны при решении задач высокой размерности, а также задач, в которых функционал не обладает свойствами унимодальности и дифференцируемости на пространстве поиска. Напротив, в задачах с гладкими и унимодальными оптимизируемыми функционалами более эффективными и быстродействующими окажутся классические градиентные методы безусловной оптимизации. На основе этого можно сделать предположение о преимуществе эволюционных алгоритмов над градиентными методами в области задач оптимального управления.

Исследования в области эффективности эволюционных алгоритмов для решения задачи оптимального управления проводились автором настоящей диссертации в составе научного коллектива во главе с профессором А.И. Дивеевым. В первых работах, появившихся в 2014 году, рассматривалась сама возможность применения эволюционных алгоритмов в задачах управления [156]. Значимые результаты, показавшие высокую эффективность и преимущество эволюционных алгоритмов, были получены к 2017 и опубликованы в [157; 167] и других работах автора. Более ранних работ в данной области в отечественной научной литературе найти не удалось. Известные работы в иностранных научных изданиях, освещающие результаты исследования эволюционных алгоритмов для решения задачи оптимального управления и их сравнения с градиентными методами, датируются началом нулевых годов нашего века [92; 121]. В этих работах утверждается преимущество использования именно эволюционных алгоритмов.

К 2019 году автором диссертации в составе научного коллектива была проведена серия масштабных сравнительных экспериментов, полностью подтвердивших приведённое выше предположение о превосходстве эволюционных алгоритмов над другими известными методами при решении задачи оптимального управления. Результаты данных исследований, показавших преимущество

эволюционных алгоритмов при сравнении с градиентными методами, опубликованы в работах [157; 160; 167; 169]. Также вполне ожидаемо подтвердилось превосходство эволюционных алгоритмов над методами случайного поиска, что показано в работах [157; 161; 165].

В наиболее значимой работе [160] рассматривалась прикладная задача оптимального управления гусеничным роботом в пространстве с двумя фазовыми ограничениями. В Таблице 2 приведены результаты сравнительного анализа градиентных и эволюционных алгоритмов, опубликованные в данной работе. Рассмотренные в сравнительном эксперименте алгоритмы и их обозначение в Таблице 2: **BA** — пчелиный алгоритм; **GW** — алгоритм серых волков; **PSO** — метод роя частиц; **GA** — генетический алгоритм; **MQ** — метод Марквардта; **AD** — метод Adam; **BIA** — метод летучих мышей; **DE** — алгоритм дифференциальной эволюции; **FGD** — метод наискорейшего градиентного спуска; **RS** — метод случайного поиска; **NR** — метод Ньютона-Рафсона.

Таблица 2 — Результаты сравнительного анализа эволюционных и градиентных алгоритмов на основе решения задачи об оптимальном быстродействии

Алгоритм	**Лучший результат**	**Среднее значение**	**СКО**
BA	2,4308	2,5284	0,0639
GW	2,4781	2,8418	0,0671
PSO	2,5100	2,7835	0,4287
GA	2,5116	2,9577	0,3543
MQ	2,5466	3,1735	0,3882
AD	2,5759	3,3047	0,5931
BIA	2,5421	3,3134	0,8846
DE	2,8599	3,6074	0,5899
FGD	2,7257	3,6222	1,0180
RS	4,2796	5,1516	0,6416
NR	3,1155	5,1373	1,6584

По условиям эксперимента задача оптимального быстродействия гусеничным роботом решалась прямым подходом путём редукции к задаче нелинейного программирования. Далее для каждого из перечисленных методов производилось 10 независимых испытаний по поиску решения задачи. Начальные данные

и настройки алгоритмов были подобраны таким образом, чтобы обеспечить одинаковые условия испытаний для каждого алгоритма. По результатам серии из 10 испытаний каждым алгоритмом в колонках Таблицы 2 приведены лучший результат из серии, среднее значение по 10 испытаниям и среднеквадратическое отклонение.

Как видно из Таблицы 2 лучшим по сумме мест по всем трём рассматриваемым критериям оказался пчелиный алгоритм. Следом с небольшим отрывом расположился алгоритм серых волков. Лучший из градиентных методов — метод Марквардта — занял 5-ую позицию в результатах сравнения. Для всех градиентных методов разница между средним значением и лучшим среди всех рассмотренных алгоритмов решением 2,4308, полученным пчелиным алгоритмом, составляет величины того же порядка. Значения среднеквадратического отклонения у градиентных методов варьируются в диапазоне от 12% до 31% от соответствующего среднего значения решения задачи.

Анализ итогов вычислительного эксперимента показал, что результаты решения прикладной задачи оптимального быстродействия, полученные рассмотренными градиентными алгоритмами, нельзя считать достоверными. Напротив, эволюционные алгоритмы в целом и пчелиный алгоритм и алгоритм серых волков в частности показывают высокую эффективность и достоверность результатов при решении задачи оптимального управления. Аналогичные выводы об эффективности эволюционных алгоритмов и их преимуществе над градиентными методами сделаны по итогам серии сравнительных экспериментов для других прикладных задач [157; 167; 169].

Таким образом, современные эволюционные алгоритмы можно рекомендовать для использования при решении задач оптимального управления. Для таких задач свойственна мультимодальность и недифференцируемость оптимизируемого функционала. Экспериментально показано, что эволюционные алгоритмы являются универсальными и гораздо менее подвержены застреванию в локальных оптимумах, чем известные градиентные методы. Качество решения различных прикладных задач эволюционными алгоритмами может зависеть как от выбора самого алгоритма, так и от значений его свободных параметров. При этом в подавляющем большинстве экспериментов пчелиный алгоритм и алгоритм серых волков показали лучшие результаты. С целью создания наиболее эффективного и универсального метода на базе данных алгоритмов можно воспользоваться методикой гибридизации.

4.5 Гибридизация алгоритмов

Как уже отмечалось ранее, главное отличие среди многообразия эволюционных алгоритмов состоит в разных схемах преобразования множества возможных решений на этапе эволюции. Несмотря на то, что эволюционные алгоритмы с успехом применяются во множестве прикладных оптимизационных задач, каждый отдельно взятый метод имеет свои достоинства и недостатки. Безусловно, есть более эффективные алгоритмы, использующие более выигрышные стратегии поиска оптимального решения, есть менее эффективные или специфичные методы. Но не существует одного универсального алгоритма, который бы одинаково эффективно решал весь спектр прикладных задач из разных областей.

В последнее время одним из основных способов повышения эффективности и области применения алгоритмов считается гибридизация [146]. В гибридных алгоритмах, объединяющих два или более алгоритма, слабость одного алгоритма компенсируется эффективностью другого и наоборот. Таким образом, гибридизация нескольких методов может породить новый более универсальный и эффективный метод для решения более широкого круга задач.

Возможность гибридных алгоритмов обойти слабые стороны своих исходных алгоритмов, но при этом не потерять свои отличительные сильные стороны, делает гибридизацию одним из основных направлений развития современных методов глобальной оптимизации, в результате которого ожидается появление новых высокоэффективных универсальных алгоритмов. На текущий момент создано и исследовано большое число гибридных алгоритмов и это число продолжает расти [84; 105; 138].

Известно несколько видов классификаций гибридизации алгоритмов глобальной оптимизации [134; 146]. Рассмотрим более детально классификацию Ванга (X. Wang) [146]. Согласно данной классификации, гибридные алгоритмы могут быть разделены на два вида по назначению проведенной гибридизации:

1. *Гибридизация с целью параметрической оптимизации.* В данном типе гибридизации второй алгоритм используется для более точной настройки параметров первого алгоритма.
2. *Гибридизация с целью улучшения поисковой стратегии.* В данном типе гибридизации поисковые стратегии нескольких алгоритмов могут

использоваться в определенном гибридизацией иерархическом порядке для создания более эффективной стратегии.

Гибридизация с целью улучшения поисковой стратегии в свою очередь может быть разделена на три типа:

1. *Гибридизация с препроцессингом и постпроцессингом.* Данный тип гибридизации является наиболее часто встречающимся. При таком типе гибридизации результаты преобразования множества возможных решений по итогам использования одного алгоритма (препроцессинг), могут быть улучшены с помощью использования второго алгоритма (постпроцессинг) (Рис. 4.1а).
2. *Гибридизация с кооператорством.* При таком типе гибридизации два или более алгоритма одновременно участвуют в процессе преобразования множества возможных решений, используя общую информацию о текущих решениях множества (Рис. 4.1б).
3. *Гибридизация вложением.* При таком типе гибридизации второй алгоритм целиком или частично вкладывается в первый алгоритм. Комбинирование разных поисковых стратегий позволяет улучшить сходимость гибридного метода (Рис. 4.1в).

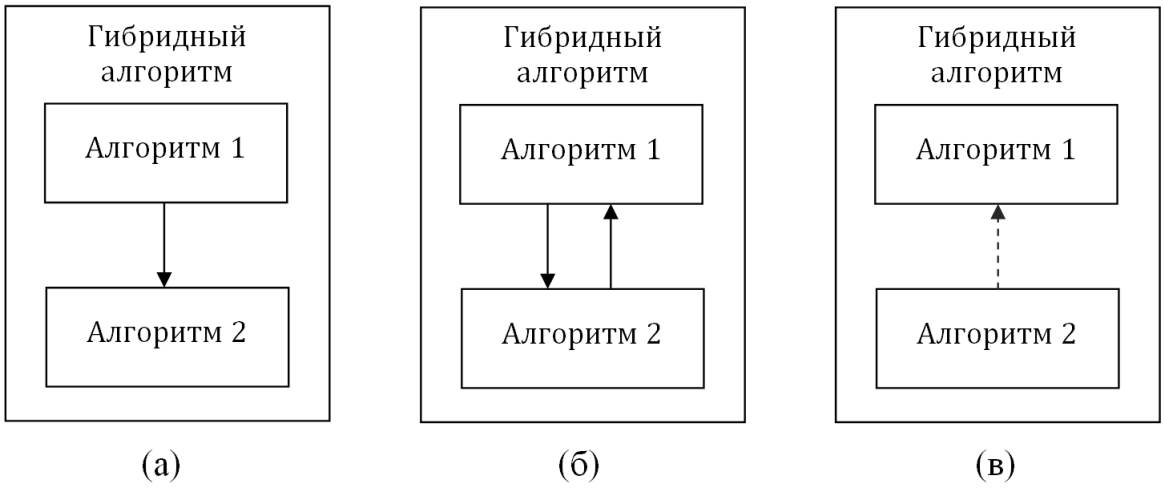


Рисунок 4.1 — Типы гибридизации с целью улучшения поисковой стратегии: (а) — гибридизация с препроцессингом и постпроцессингом; (б) — гибридизация с кооператорством; (в) — гибридизация вложением.

4.6 Гибридный алгоритм на основе алгоритма серых волков и пчелиного алгоритма

В данном разделе рассматривается новый гибридный алгоритм, разработанный на основе алгоритма серых волков и пчелиного алгоритма и впервые представленный автором диссертации в работе [177]. Методы, составившие основу оригинального гибридного алгоритма, хорошо зарекомендовали себя для решения разнообразных сложных мультимодальных оптимизационных задач, в том числе задач оптимального управления, что показано в работах автора [157—161; 165; 167; 169].

В соответствии с классификацией Ванга, рассматриваемый гибридный алгоритм по типу относится к гибридизации вложением, причём объединяемые методы имеют довольно высокую степень интегрированности, что позволяет говорить о получении нового метода. Процедура выбора элитных решений и поиска в заданном радиусе вокруг них и процедура обновления заданного числа худших решений новыми случайными возможными решениями, унаследованные от пчелиного алгоритма, составляют исследовательскую часть поиска. Локальную часть поиска составляет унаследованная от алгоритма серых волков процедура модификации возможных решений на основе информации о трех текущих лучших решениях, которая позволяет концентрировать решения в районе глобального лучшего решения. При этом предложенный гибридный метод имеет даже меньшее число настраиваемых параметров, чем исходные методы.

Рассмотрим основные этапы оригинального гибридного алгоритма.

Задаем параметры алгоритма — размер множества возможных решений H, число итераций W, количество B исследуемых возможных решений, $B = \lfloor 0{,}3H \rfloor$, число S худших решений, подлежащих замене.

Задаём текущее значение счетчика итераций $w = 0$, задаем начальные значения вектора радиусов поиска вокруг лучших решений $\mathbf{r} = [r_1 \dots r_p]^T$, $r_i > 0$, $i = \overline{1,p}$.

Генерируем множество векторов параметров $\mathbf{q}^j$, $j = \overline{1,H}$, по формуле (4.3).

На каждой итерации для всех возможных решений вычисляем значения функционала $J\left(\mathbf{q}^j\right)$, $j = \overline{1,H}$, и упорядочиваем возможные решения во множестве по возрастанию значения целевой функции (4.4).

Производим вычисление значения параметра линеаризации по формуле (4.8).

Производим исследующий поиск вокруг заданного числа B лучших возможных решений $\mathbf{q}^k$, $k = \overline{1,B}$. Для этого для каждого возможного решения $\mathbf{q}^k$ строим набор из E_k векторов $\mathbf{g}^{k_e} = \left[g_1^{k_e} \dots g_p^{k_e}\right]^T$

$$g_i^{k^e} = \begin{cases} q_i^-, \text{ если } q_i^k + \xi_i^{k_e} < q_i^- \\ q_i^+, \text{ если } q_i^k + \xi_i^{k_e} > q_i^+ \\ q_i^k + \xi_i^{k_e} \text{ — иначе} \end{cases}, \quad i = \overline{1,p},\ e = \overline{1,E_k},\ k = \overline{1,B},$$

где $\xi_i^{k_e}$ — величина, случайно распределенная на интервале $[-r_i; r_i]$; E_k — число новых векторов $\mathbf{g}^{k_e}$ в области исследуемого вектора $\mathbf{q}^k$. Значение числа новых векторов E_k может быть постоянным и выбираться под конкретную прикладную задачу (обычно $E_k = H$) или динамическим и уменьшаться с уменьшением привлекательности решения $\mathbf{q}^k$, тогда $E_k = \overline{\lfloor 1{,}5H \rfloor, \lfloor 0{,}5H \rfloor}$, $k = \overline{1,B}$.

Если какой-либо вектор $\mathbf{g}^{k_e}$ доставляет значение функционала меньше, чем вектор $\mathbf{q}^k$, то заменяем им вектор $\mathbf{q}^k$

— если $J\left(\mathbf{g}^{k_e}\right) < J\left(\mathbf{q}^k\right)$, то $\mathbf{q}^k \leftarrow \mathbf{g}^{k_e}$, $J\left(\mathbf{q}^k\right) \leftarrow J\left(\mathbf{g}^{k_e}\right)$, $e = \overline{1,E_k}$, $k = \overline{1,B}$.

Выбираем три наилучших решения $\mathbf{q}^\alpha$, $\mathbf{q}^\beta$, $\mathbf{q}^\delta$ из всего текущего множества возможных решений по формулам (4.5) — (4.7).

Для каждого возможного решения $\mathbf{q}^j$, $j = \overline{1,H}$ в текущем множестве производим вычисление трех вспомогательных векторов $\boldsymbol{\alpha}^j$, $\boldsymbol{\beta}^j$, $\boldsymbol{\delta}^j$ по формулам (4.9) — (4.11).

На основе полученных векторов $\boldsymbol{\alpha}^j$, $\boldsymbol{\beta}^j$, $\boldsymbol{\delta}^j$ производим модификацию возможного решения $\mathbf{q}^j$ по формуле (4.12).

Для модифицированного множества возможных решений производим вычисление значений функционала $J\left(\mathbf{q}^j\right)$, $j = \overline{1,H}$, и упорядочение возможных решений по возрастанию значения функционала. Далее для S худших решений производим генерацию новых случайных значений по формуле (4.3).

Производим уменьшение радиусов области поиска вокруг исследуемых векторов

$$r_i \leftarrow \boldsymbol{\gamma} r_i,\ i = \overline{1,p},$$

где $\boldsymbol{\gamma}$ — заданный параметр уменьшения области поиска, $\boldsymbol{\gamma} \approx 0{,}95$.

Увеличиваем значение счётчика итераций $w \leftarrow w + 1$ и повторяем описанные выше вычисления. Процесс поиска и модификации возможных решений продолжаем до достижения максимального числа итераций W. В качестве решения задачи выбираем лучший вектор $\mathbf{q}$ из итогового множества возможных решений.

Представленный в работе [177] сравнительный анализ эффективности данного гибридного алгоритма показал высокое качество гибридизации. В новом гибридном алгоритме удалось достичь взаимной компенсации слабых сторон пчелиного алгоритма и алгоритма серых волков. Совокупная эффективность гибридного метода по всем рассмотренным в [177] тестам оказалась выше чем у исходных алгоритмов по отдельности.

Выводы по Главе 4

В большинстве случаев при решении сложных прикладных задач оптимального управления, описываемых системами высокого порядка, применяют прямые методы. Данные методы используют идею редукции исходной задачи к задаче нелинейного программирования. Это позволяет перейти от задачи бесконечномерной оптимизации к задаче конечномерной оптимизации. Для её решения доступно большое число численных методов оптимизации, среди которых можно выделить классические методы нулевого порядка, методы случайного поиска, градиентные методы и эволюционные алгоритмы.

Переход к задаче конечномерной оптимизации достигается за счёт дискретизации функции управления, что сказывается на значительном увеличении размерности пространства поиска оптимального решения. Полученная в результате редукции целевая функция может быть многоэкстремальной. Определить её топологические свойства и гарантировать выпуклость даже на ограниченной области, как правило, не представляется возможным.

В данных обстоятельствах методы из класса эволюционных алгоритмов оказались намного эффективнее других методов. Каждый алгоритм данного класса предлагает определенную схему исследовательского и локального поиска. С одной стороны, это позволяет данным алгоритмам диверсифицировать поиск для уменьшения вероятности застревания алгоритма в точках локальных

экстремумов и седловых точках, а, с другой стороны, — интенсифицировать поиск в области лучших решений для обеспечения сходимости алгоритма.

Высокая эффективность эволюционных алгоритмов и их превосходство над методами случайного поиска и градиентными методами при решении задачи оптимального управления подтверждены рядом сравнительных экспериментов, опубликованных в работах автора [157—161; 165; 167; 169]. Наилучшие результаты среди всех сравниваемых алгоритмов показали пчелиный алгоритм и алгоритм серых волков.

Практический интерес представляет гибридизация алгоритмов. Создание гибридных алгоритмов в настоящее время является одним из основных способов создания новых универсальных поисковых алгоритмов. Преимущество гибридных алгоритмов достигается за счёт взаимной компенсации недостатков отдельно взятых исходных алгоритмов. Создание и использование гибридных эволюционных алгоритмов для решения задачи оптимального управления имеет хорошие перспективы.

В работе предлагается новый гибридный алгоритм на основе хорошо зарекомендовавших себя для решения разнообразных задач оптимального управления пчелиного алгоритма и алгоритма серых волков. Новый гибридный алгоритм имеет высокую степень интегрированности исходных алгоритмов, лежащих в его основе, и успешно наследует их сильные стороны. Сравнительные эксперименты показали его более высокую эффективность, чем у исходных алгоритмов. Новый гибридный алгоритм предлагается использовать для поиска численного решения прикладных задач оптимального управления.

Глава 5. Прикладная задача синтеза системы управления автомобилеподобным роботом

Главным показателем качества любого численного метода является его эффективность при решении прикладных задач. Предложенный в Главе 3 метод синтеза системы управления на основе аппроксимации множества оптимальных траекторий методом символьной регрессии был применен для поиска решения прикладной задачи синтеза системы управления автомобилеподобным роботом.

В соответствии с предложенным подходом для объекта управления на первом этапе необходимо численно решить задачу оптимального управления для разных начальных состояний. Далее на основе полученных оптимальных траекторий строится обучающая выборка и производится поиск структуры функции управления, наиболее точно аппроксимирующей обучающую выборку.

В рассматриваемой прикладной задаче искомое решение должно учитывать ограничения на положение автомобилеподобного робота в фазовом пространстве. Это существенно усложняет как поиск решения задачи оптимального управления, так и поиск структуры функции управления.

Решение задачи оптимального управления осуществлялось прямым методом на основе нового гибридного алгоритма, рассмотренного в Главе 4, а дальнейший поиск многомерной функции управления — с помощью метода сетевого оператора.

5.1 Математическая модель автомобилеподобного робота

В качестве объекта управления был выбран четырехколесный мобильный робот [58], представленный на Рис. 5.1. На задней оси робота расположены ведущие колеса, а колеса передней оси отвечают за поворот робота. Положение в пространстве определяется фазовыми координатами x_c и y_c базовой точки робота, которая расположена на середине задней оси, а также углом θ между центральной осью мобильного робота и осью абсцисс неподвижной системы координат. Такой вид роботов принято называть автомобилеподобными [93].

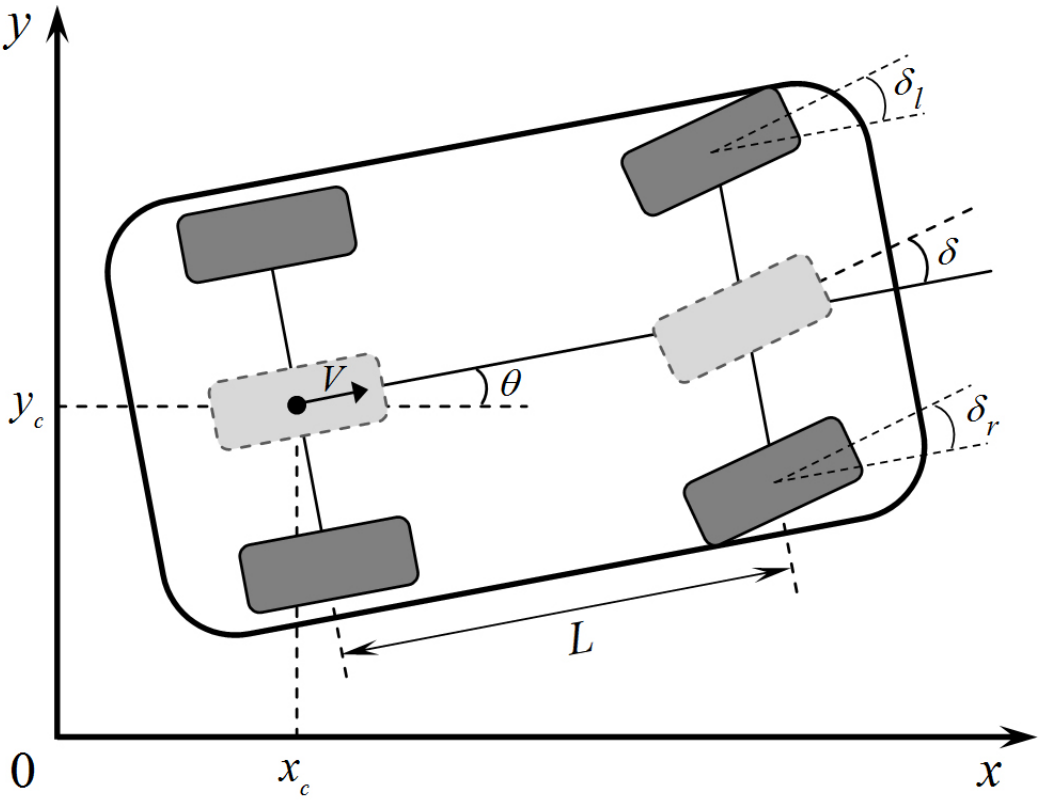


Рисунок 5.1 — Четырехколесный автомобилеподобный робот.

Математическая модель автомобилеподобного робота имеет вид

$$\begin{cases} \dfrac{dx}{dt} = V\cos(\theta) \\ \dfrac{dy}{dt} = V\sin(\theta) \\ \dfrac{d\theta}{dt} = \dfrac{V}{L}\operatorname{tg}(\delta) \\ \dfrac{d\delta}{dt} = \omega \end{cases}, \tag{5.1}$$

где V — вектор мгновенной скорости базовой точки робота, L — колесная база, δ — угол поворота передних колес, ω — угловая скорость поворота рулевого управления. В реальных условиях при совершении поворота, углы поворота двух передних колес различны, так как левое и правое передние колеса движутся по разным траекториям, т.е. $\delta_l \neq \delta_r$. Однако двухколесное рулевое управление можно интерпретировать как одноколесное, расположенное на центральной оси с соблюдением длины колесной базы L (Рис. 5.1). Тогда верно равенство

$$\delta = \frac{\delta_l + \delta_r}{2}.$$

Описанная математическая модель характерна для небольших мобильных роботов с шаговыми двигателями. В таких моделях проскальзыванием колес и инерционными составляющими движения можно пренебречь. Для упрощения математической модели (5.1) можно предположить, что угол поворота колес δ изменяется мгновенно. Тогда упрощенная математическая модель будет иметь вид [132]

$$\begin{cases} \dfrac{dx}{dt} = V\cos(\theta) \\ \dfrac{dy}{dt} = V\sin(\theta) \\ \dfrac{d\theta}{dt} = \dfrac{V}{L}\operatorname{tg}(\delta) \end{cases}. \tag{5.2}$$

При моделировании необходимо также учитывать ограничения на скорость движения мобильного робота V и на угол поворота его руля δ, которые могут варьироваться в зависимости от технических характеристик робота.

Введем вектор состояния объекта $\mathbf{x}$ и вектор управления $\mathbf{u}$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}, \tag{5.3}$$

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} V \\ \delta \end{bmatrix}. \tag{5.4}$$

Перезапишем модель (5.2) в новой форме с учетом (5.3) и (5.4)

$$\begin{cases} \dot{x}_1 = u_1\cos(x_3) \\ \dot{x}_2 = u_1\sin(x_3) \\ \dot{x}_3 = \dfrac{u_1}{L}\operatorname{tg}(u_2) \end{cases}. \tag{5.5}$$

Зададим ограничения на вектор управления $\mathbf{u}$

$$\mathbf{u}^- \leqslant \mathbf{u} \leqslant \mathbf{u}^+, \tag{5.6}$$

где $\mathbf{u}^- = \begin{bmatrix} u_1^- & u_2^- \end{bmatrix}^T = \begin{bmatrix} -10 & -1 \end{bmatrix}^T$, $\mathbf{u}^+ = \begin{bmatrix} u_1^+ & u_2^+ \end{bmatrix}^T = \begin{bmatrix} 10 & 1 \end{bmatrix}^T$.

5.2 Задача синтеза системы управления автомобилеподобным роботом в пространстве с фазовыми ограничениями

Решением задачи синтеза системы управления автомобилеподобным роботом является управление в форме многомерной функции от координат состояния робота, позволяющее оптимально по заданному критерию качества переместить робота из начального состояния в терминальное. При этом начальное состояние робота может быть любым или принадлежать какой-либо ограниченной области.

Для постановки задачи синтеза системы управления автомобилеподобным роботом необходимо задать начальное и терминальное состояние робота. Пусть начальное состояние робота определено на ограниченной области начальных условий

$$\mathrm{X}_0 = \{7 \leqslant x_{0,1} \leqslant 9,\ \ 9 \leqslant x_{0,2} \leqslant 11,\ \ x_{0,3} = \pi\}\,. \tag{5.7}$$

Зададим терминальное состояние

$$\mathbf{x}^f = \begin{bmatrix} x_1^f & x_2^f & x_3^f \end{bmatrix}^T = \begin{bmatrix} 0 & 0 & \pi \end{bmatrix}^T. \tag{5.8}$$

Пусть на плоскости движения робота имеются препятствия. Зададим их в виде фазовых ограничений

$$h_i\,(x) = r_i^* - \sqrt{\left(x_{i,1}^* - x_1\right)^2 + \left(x_{i,2}^* - x_2\right)^2} \leqslant 0,\ i = \overline{1,z}, \tag{5.9}$$

где r_i^*, $x_{i,1}^*$ и $x_{i,2}^*$ — заданные параметры фазовых ограничений, z — число фазовых ограничений.

Параметры фазовых ограничений имеют следующие значения: $z = 4$, $r_1^* = 3$, $x_{1,1}^* = 2$, $x_{1,2}^* = 8$, $r_2^* = 3$, $x_{2,1}^* = 8$, $x_{2,2}^* = 2$, $r_3^* = 1{,}5$, $x_{3,1}^* = 2{,}5$, $x_{3,2}^* = 2{,}5$, $r_4^* = 1{,}5$, $x_{4,1}^* = 7{,}5$, $x_{4,2}^* = 7{,}5$.

По условию задачи необходимо найти структуру многомерной функции управления $\mathbf{u}\,(\mathbf{x})$, обеспечивающую перемещение автомобилеподобного робота из любого начального положения $\mathbf{x}^0 \in \mathrm{X}_0$ в терминальное положение $\mathbf{x}^f$ за минимальное время. Рассматриваемая задача является задачей об оптимальном быстродействии. Функционал качества для данной задачи, помимо времени процесса управления, должен учитывать ошибку управления и все фазовые

ограничения

$$\forall \mathbf{x}^0 \in \mathrm{X}_0 : \; J = t_f(\mathbf{x}^0) + \left\| \mathbf{x}\left(t_f(\mathbf{x}^0)\right) - \mathbf{x}^f \right\| + \\ + \int\limits_{t_0}^{t_f(\mathbf{x}^0)} \left(\sum_{i=1}^{z} \alpha_i \vartheta\left(h_i\left(\mathbf{x}\right)\right) h_i\left(\mathbf{x}\right) \right) dt \to \min \quad , \tag{5.10}$$

где $t_f(\mathbf{x}^0)$ — время процесса управления для начального состояния $\mathbf{x}^0$,

$$t_f(\mathbf{x}^0) = \begin{cases} t(\mathbf{x}^0), \text{ если } \left\| \mathbf{x}\left(t_f(\mathbf{x}^0)\right) - \mathbf{x}^f \right\| \leqslant \varepsilon \\ t_{\max} \text{ — иначе} \end{cases} ,$$

ε — заданная малая положительная величина, t_0 и $t_{\max}$ — заданные значения начального времени и ограничения на время процесса управления соответственно, $t_{\max} > t_0$, α_i — штрафной коэффициент, $\vartheta\left(h_i\left(\mathbf{x}\right)\right)$ — функция Хэвисайда (1.22).

5.3 Численное решение задачи синтеза системы управления автомобилеподобным роботом на основе аппроксимации оптимальных траекторий

Поиск численного решения задачи синтеза системы управления автомобилеподобным роботом осуществлялся предложенным методом на основе аппроксимации оптимальных траекторий. На первом этапе требовалось определить конечное множество начальных условий и решить задачу оптимального управления для каждого из них. Результатом выполнения первого этапа является найденное множество оптимальных траекторий. На втором этапе множество оптимальных траекторий использовалось как обучающая выборка при поиске структуры математического выражения методом символьной регрессии. Решением поставленной задачи будет функция управления от координат состояния автомобилеподобного робота в форме математического выражения, максимально точно аппроксимирующего множество оптимальных траекторий.

5.3.1 Поиск множества оптимальных траекторий

Заменим ограниченную область начальных условий (5.7) на конечное множество $\tilde{\mathrm{X}}_0$ из $N = 9$ начальных состояний

$$\begin{aligned}\tilde{\mathrm{X}}_0 \ = \{ \ & \mathbf{x}^{0,1} = [7\ 9\ \pi]^T, \quad \mathbf{x}^{0,2} = [8\ 9\ \pi]^T, \quad \mathbf{x}^{0,3} = [9\ 9\ \pi]^T, \\ & \mathbf{x}^{0,4} = [7\ 10\ \pi]^T, \ \mathbf{x}^{0,5} = [8\ 10\ \pi]^T, \ \mathbf{x}^{0,6} = [9\ 10\ \pi]^T, \\ & \mathbf{x}^{0,7} = [7\ 11\ \pi]^T, \ \mathbf{x}^{0,8} = [8\ 11\ \pi]^T, \ \mathbf{x}^{0,9} = [9\ 11\ \pi]^T \ \}.\end{aligned} \tag{5.11}$$

Точки множества (5.11) выбраны с учетом равномерного покрытия области начальных условий (5.7) сеткой с шагом $\Delta = 1$. Все точки множества (5.11) кроме $\mathbf{x}^{0,5} = [8\ 10\ \pi]^T$ являются граничными точками множества (5.7).

Для каждого начального состояния из множества (5.11) необходимо найти решение задачи оптимального управления в форме $\mathbf{u}\left(\mathbf{x}^{0,j},t\right)$, обеспечивающее минимум функционалу

$$\begin{aligned}J_j^{ocp} = \ & t_f(\mathbf{x}^{0,j}) + \left\|\mathbf{x}\left(t_f(\mathbf{x}^{0,j})\right) - \mathbf{x}^f\right\| + \\ & + \int_{t_0}^{t_f(\mathbf{x}^{0,j})} \left(\sum_{i=1}^{z} \alpha_i \vartheta\left(h_i\left(\mathbf{x}\right)\right) h_i\left(\mathbf{x}\right)\right) dt \to \min \end{aligned}, \ j = \overline{1,N}. \tag{5.12}$$

Для поиска решения использовались прямые методы решения задачи оптимального управления. Для этого исходная задача оптимального управления должна быть редуцированна к задаче нелинейного программирования с помощью кусочно-линейной аппроксимации.

Зададим малый интервал $\Delta t > 0$ и определим количество интервалов M по формуле (4.2). Значение управления $\mathbf{u}\left(\mathbf{x}^{0,j},t\right) = \left[\begin{array}{cc} u_1\left(\mathbf{x}^{0,j},t\right) & u_2\left(\mathbf{x}^{0,j},t\right) \end{array}\right]^T$ в момент времени t определяется из соотношения

$$u_1\left(\mathbf{x}^{0,j},t\right) = \begin{cases} u_1^-, \text{ если } q_i + (q_{i+1} - q_i)\dfrac{(t-(i-1)\Delta t)}{\Delta t} < u_1^- \\ u_1^+, \text{ если } q_i + (q_{i+1} - q_i)\dfrac{(t-(i-1)\Delta t)}{\Delta t} > u_1^+ \\ q_i + (q_{i+1} - q_i)\dfrac{(t-(i-1)\Delta t)}{\Delta t} \text{ — иначе} \end{cases},$$

$$u_2\left(\mathbf{x}^{0,j},t\right) = \begin{cases} u_2^-, \text{ если } q_k(q_{k+1} - q_k)\dfrac{(t-(i-1)\Delta t)}{\Delta t} < u_2^- \\ u_2^+, \text{ если } q_j + (q_{j+1} - q_j)\dfrac{(t-(i-1)\Delta t)}{\Delta t} > u_2^+ \\ q_j + (q_{j+1} - q_j)\dfrac{(t-(i-1)\Delta t)}{\Delta t} \text{ — иначе} \end{cases}, \tag{5.13}$$

где $i\Delta t \leqslant t < (i+1)\Delta t$, $j = \overline{1,N}$, $i = \overline{1,M}$, $k = \overline{M+1,\,2M}$.

Таким образом, решением каждой отдельно взятой задачи оптимального управления будет вектор постоянных параметров

$$\mathbf{q}^j = \left[q_1^j \dots q_p^j\right]^T, \tag{5.14}$$

где $q_i^- \leqslant q_i^j \leqslant q_i^+$, q_i^-, q_i^+ — заданные значения ограничений на параметры, $\mathbf{q}^j \in \mathbb{R}^p$, $j = \overline{1,N}$, $i = \overline{1,p}$, $p = 2\,(M+1)$.

Увеличение размерности пространства поиска при переходе к задаче конечномерной оптимизации и наличие фазовых ограничений значительно усложняет поиск решения задачи оптимального управления. В Главе 4 показано, что в таких случаях наиболее эффективными оказываются эволюционные алгоритмы. В вычислительном эксперименте поиск решений задачи оптимального управления осуществлялся с помощью нового гибридного алгоритма на основе алгоритма серых волков и пчелиного алгоритма. Описание данного алгоритма приведено в разделе 4.6.

Во время поиска использовались следующие значения параметров: начальное время $t_0 = 0$; ограничение на время процесса управления $t_{max} = 2{,}5$; малый интервал времени $\Delta t = 0{,}25$; число интервалов $M = \left\lceil \frac{t_{max}}{\Delta t} \right\rceil = 10$; точность достижения терминального состояния при решении задачи оптимального управления $\varepsilon = 0{,}01$; число фазовых ограничений $z = 4$; штрафной коэффициент при нарушении фазовых ограничений $\alpha_i = 1$, $i = \overline{1,z}$; размер искомого вектора постоянных параметров $\mathbf{q}^j$, $j = \overline{1,N}$, $p = 2(M+1) = 22$; ограничения на значения компонент вектора постоянных параметров $\mathbf{q}^j$, $j = \overline{1,N}$, $q_i^- = -12$, $q_i^+ = 12$, $i = \overline{1,\,p/2}$, $q_i^- = -1$, $q_i^+ = 1$, $i = \overline{p/2+1,\,p}$; размер множества возможных решений $H = 30$; число поколений $W = 300$; число элитных решений $B = 10$; число возможных решений в области элитного решения $E^k = 30$, $k = \overline{1,B}$; радиус поиска вокруг элитного решения $r_i = 2{,}4$, $i = \overline{1,\,p/2}$, $r_i = 0{,}2$, $i = \overline{p/2+1,\,p}$; коэффициент редукции радиуса поиска $\gamma = 0{,}95$; число худших решений, подлежащих замене $S = 20$.

Значения функционала качества (5.12), полученные при решении задачи оптимального управления для различных начальных условий $\mathbf{x}^{0,j} \in \tilde{\mathrm{X}}_0$, $j = \overline{1,N}$, приведены в Таблице 3.

В результате решения редуцированной к задаче нелинейного программирования исходной задачи оптимального управления для каждого начального

Таблица 3 — Результаты решения задачи оптимального управления для различных начальных условий

j	$\mathbf{x}^{0,j}$	J_j^{ocp}
1	$\mathbf{x}^{0,1} = [7\ 9\ \pi]^T$	1,2883
2	$\mathbf{x}^{0,2} = [8\ 9\ \pi]^T$	1,4063
3	$\mathbf{x}^{0,3} = [9\ 9\ \pi]^T$	1,4712
4	$\mathbf{x}^{0,4} = [7\ 10\ \pi]^T$	1,3974
5	$\mathbf{x}^{0,5} = [8\ 10\ \pi]^T$	1,4182
6	$\mathbf{x}^{0,6} = [9\ 10\ \pi]^T$	1,5095
7	$\mathbf{x}^{0,7} = [7\ 11\ \pi]^T$	1,4855
8	$\mathbf{x}^{0,8} = [8\ 11\ \pi]^T$	1,5097
9	$\mathbf{x}^{0,9} = [9\ 11\ \pi]^T$	1,5935

состояния из множества (5.11) были получены значения векторов параметров (5.14), которые при подстановке в (5.13) дадут решение в форме множества функций управления от времени. Подставив найденные функции управления от времени в модель объекта (5.5), получим множество пар программных управлений и соответствующих их траекторий

$$\tilde{\mathrm{D}} = \left\{ \left(\tilde{\mathbf{x}}^1(\cdot), \tilde{\mathbf{u}}^1(\cdot) \right), \ldots, \left(\tilde{\mathbf{x}}^N(\cdot), \tilde{\mathbf{u}}^N(\cdot) \right) \right\}, \tag{5.15}$$

где $\tilde{\mathbf{x}}^j(\cdot)$ — частное решение системы (5.5) для начального условия $\mathbf{x}(0) = \mathbf{x}^{0,j}$ из множества (5.11), $\tilde{\mathbf{u}}^j(\cdot)$ — решение задачи оптимального управления для заданного начального условия, $j = \overline{1,N}$. Ввиду использования прямого подхода и применения эволюционных алгоритмов, найденные решения будут приблизительно оптимальными. Соответственно на втором этапе в аппроксимируемом множестве используются приблизительно оптимальные траектории.

На Рис. 5.2 представлены полученные при решении задачи оптимального управления траектории движения автомобилеподобного робота на плоскости.

5.3.2 Синтез системы управления автомобилеподобным роботом

Для реализации второго этапа предлагаемого подхода — непосредственно синтеза системы управления автомобилеподобным роботом, необходимо под-

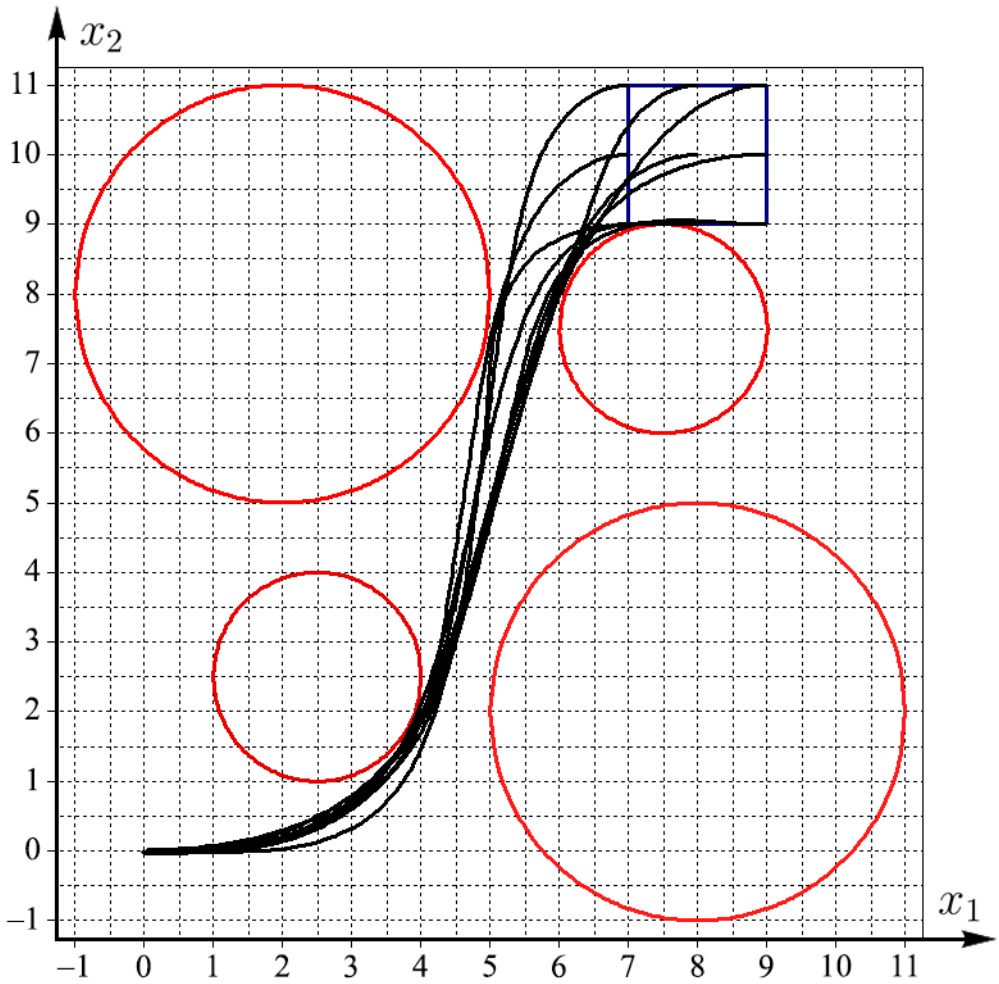


Рисунок 5.2 — Графики траекторий из различных начальных состояний.

готовить обучающую выборку на основе полученных на предыдущем шаге оптимальных траекторий. Синтез системы управления осуществляется с помощью одного из методов символьной регрессии путем численной аппроксимации значений обучающей выборки.

Для представления множества найденных траекторий (5.15) в форме обучающей выборки, пригодной для численной аппроксимации, произведем дискретизацию по времени.

Введем малое значение $\Delta_s t > 0$ и определим множество дискретных значений времени для всех найденных частных решений во множестве (5.15)

$$\mathrm{T}_j = (0,\, \Delta_s t,\, 2\Delta_s t,\, \ldots,\, M_j \Delta_s t)\,,$$

где $M_j = \left\lceil \dfrac{t_f(\mathbf{x}^{0,j})}{\Delta_s t} \right\rceil$, $t_f(\mathbf{x}^{0,j})$ — время процесса управления для найденного решения задачи оптимального управления из начального состояния $\mathbf{x}^{0,j}$, $j = \overline{1,N}$.

Обучение в ходе поиска оптимального математического выражения функции управления автомобилеподобным роботом производилось на основе оп-

тимальных траекторий. Значения найденных программных управлений в обучении не использовались. Таким образом, для каждого начального состояния (5.11) было сформировано множество из значений вектора состояния в дискретный момент времени $t_{j,i} \in \mathrm{T}_j$, $i = \overline{1,M_j}$, $j = \overline{1,N}$

$$\mathrm{D}_j = \left\{ \left(\tilde{\mathbf{x}}^{j,1} \right), \ldots, \left(\tilde{\mathbf{x}}^{j,M_j} \right) \right\}, \tag{5.16}$$

где $\tilde{\mathbf{x}}^{j,i} = \tilde{\mathbf{x}}^j \left(t_{j,i} \right)$, $i = \overline{1,M_j}$ — значение вектора состояния автомобилеподобного робота в дискретный момент времени $t_{j,i} \in \mathrm{T}_j$ при движении из начального состояния $\mathbf{x}^{0,j}$, $j = \overline{1,N}$.

Итоговая обучающая выборка представляла собой набор множеств значений вектора состояния робота (5.16)

$$\mathrm{D} = \left\{ \mathrm{D}_1, \ldots, \mathrm{D}_N \right\}. \tag{5.17}$$

Численным решением задачи синтеза системы управления автомобилеподобным роботом будет многомерная функция управления от координат вектора состояния объекта и вектора ее параметров

$$\tilde{\mathbf{u}} = \mathbf{g}^* \left(\mathbf{x}, \mathbf{s}^* \right). \tag{5.18}$$

Многомерная функция управления (5.18) должна обеспечивать перемещение автомобилеподобного робота из любого начального положения $\mathbf{x}^{0,j} \in \tilde{\mathrm{X}}_0$, $j = \overline{1,N}$ (5.11) в терминальное положение (5.8) по траектории, наиболее близкой к оптимальной.

Поиск математического выражения многомерной функции управления (5.18) осуществлялся с помощью методов символьной регрессии путем численной аппроксимации точек оптимальных траекторий из обучающей выборки (5.17). При достаточном качестве аппроксимации, найденная многомерная функция управления (5.18) обеспечит перемещение объекта управления по близким к оптимальным траекториям для любых начальных состояний из заданной области начальных условий $\mathbf{x}^0 \in \mathrm{X}_0$ (5.7).

Из условия аппроксимации точек оптимальных траекторий следует, что искомая многомерная функция управления (5.18) при подстановке в математическую модель автомобилеподобного робота (5.5) должна давать частные решения, удовлетворяющие критерию

$$J = \sum_{j=1}^{N} \sum_{i=1}^{M_j} \left\| \tilde{\mathbf{x}}^{j,i} - \mathbf{x}^j \left(\mathbf{x}^{0,j}, t_{j,i} \right) \right\| \to \min, \tag{5.19}$$

где $\mathbf{x}^j\left(\mathbf{x}^{0,j},t_{j,i}\right)$ — частное решение системы (5.5) для начального условия $\mathbf{x}^{0,j}$ в дискретный момент времени $t_{j,i} \in \mathrm{T}_j$, $\tilde{\mathbf{x}}^{j,i}$ — эталонное решение из обучающей выборки (5.17) для соответствующего начального условия и момента времени, $i = \overline{1,M_j}$, $j = \overline{1,N}$.

В рассматриваемой в вычислительном эксперименте задаче также заданы фазовые ограничения на вектор состояния объекта управления (5.9). Безусловно, оптимальные траектории, используемые для поиска математического выражения функции управления, удовлетворяют условиям данных фазовых ограничений. Однако сами возможные решения, рассматриваемые в процессе поиска, могут их нарушать. Чтобы избежать данного случая, который также проиллюстрирован в примере в разделе 3.1, добавим в критерий (5.19) штраф за нарушение фазовых ограничений

$$J = \sum_{j=1}^{N}\sum_{i=1}^{M_j}\left\|\tilde{\mathbf{x}}^{j,i} - \mathbf{x}^j\left(\mathbf{x}^{0,j},t_{j,i}\right)\right\| + \sum_{k=1}^{z}\alpha_k\vartheta\left(h_k\left(\mathbf{x}\right)\right)h_k\left(\mathbf{x}\right) \to \min, \qquad (5.20)$$

где α_k — штрафной коэффициент, $h_k\left(\mathbf{x}\right)$ — фазовое ограничение в пространстве вектора состояния объекта, $k = \overline{1,z}$, z — число фазовых ограничений, $\vartheta\left(h_k\left(\mathbf{x}\right)\right)$ — функция Хэвисайда (1.22).

Также следует отметить, что, согласно критерию (5.19), для всех точек из обучающей выборки близость их аппроксимации имеет одинаковый приоритет. Равновесность всех точек обучающей выборки не является применимым подходом для рассматриваемой задачи. Действительно, близость найденного решения к последней точке каждого множества $\mathrm{D}_j \in \mathrm{D}$, $j = \overline{1,N}$ (5.17), является более приоритетной, чем близость ко всем остальным точкам, ввиду того, что последняя точка каждого множества D_j по сути является терминальным состоянием объекта управления, которое необходимо достичь

$$\forall \mathrm{D}_j \in \mathrm{D} : \tilde{\mathbf{x}}^{j,M_j} \approx \mathbf{x}^f = \begin{bmatrix}0 & 0 & \pi\end{bmatrix}^T,\ j = \overline{1,N}.$$

Дополним критерий (5.20) значением оценки достижения возможным решением терминального состояния (5.8). Тогда критерий оценки качества обучения примет следующий окончальный вид

$$\begin{aligned} J = & \sum_{j=1}^{N}\left(\lambda\left\|\mathbf{x}^f - \mathbf{x}^j\left(\mathbf{x}^{0,j},t_{j,M_j}\right)\right\| + \sum_{i=1}^{M_j-1}\left\|\tilde{\mathbf{x}}^{j,i} - \mathbf{x}^j\left(\mathbf{x}^{0,j},t_{j,i}\right)\right\|\right) + \\ & + \sum_{k=1}^{z}\alpha_k\vartheta\left(h_k\left(\mathbf{x}\right)\right)h_k\left(\mathbf{x}\right) \to \min, \end{aligned}$$

где λ — весовой коэффициент оценки достижения терминального состояния $\mathbf{x}^f$, α_k — штрафной коэффициент нарушения фазового ограничения, $k = \overline{1,z}$. Для рассматриваемой задачи значения коэффициентов λ и α_k были подобраны эмпирически так, чтобы исключить возможность нарушения фазовых ограничений и соблюсти точность достижения терминального состояния: $\lambda = 3$; $\alpha_k = 5$.

В вычислительном эксперименте синтез системы управления автомобилеподобным роботом осуществлялся с помощью метода сетевого оператора, относящегося к классу методов символьной регрессии. Описание данного метода приведено в разделе 2.2.5. Выбор метода сетевого оператора из ряда других методов класса обусловлен тем, что при его создании учитывалась специфика задач синтеза управления, а следовательно, он больше других подходит для использования в данном вычислительном эксперименте.

Во время поиска математической формы функции управления использовались следующие значения параметров метода сетевого оператора: число входных переменных — 3; число параметров — 12; число выходных значений — 2; множество переменных и параметров

$$\begin{aligned} \mathrm{F}_0 = (\quad & f_{0,1} = x_1,\, f_{0,2} = x_2,\, f_{0,3} = x_3,\, f_{0,4} = s_1,\, f_{0,5} = s_2, \\ & f_{0,6} = s_3,\, f_{0,7} = s_4,\, f_{0,8} = s_5,\, f_{0,9} = s_6,\, f_{0,10} = s_7, \\ & f_{0,11} = s_8,\, f_{0,12} = s_9,\, f_{0,13} = s_{10},\, f_{0,14} = s_{11},\, f_{0,15} = s_{12}\ ); \end{aligned} \tag{5.21}$$

множество функций с одним аргументом

$$\begin{aligned} \mathrm{F}_1 = (\quad & f_{1,1}(z) = z,\, f_{1,2}(z) = z^2,\, f_{1,3}(z) = -z, \\ & f_{1,4}(z) = \operatorname{sgn}(z)\sqrt{|z|},\, f_{1,5}(z) = \frac{1}{z},\, f_{1,6}(z) = \mathrm{e}^z, \\ & f_{1,7}(z) = \ln(|z|),\, f_{1,8}(z) = \frac{1-\mathrm{e}^{-z}}{1+\mathrm{e}^{-z}},\, f_{1,9}(z) = \arctan(z), \\ & f_{1,10}(z) = z^3,\, f_{1,11}(z) = \sqrt[3]{z},\, f_{1,12}(z) = z - z^3\ ); \end{aligned} \tag{5.22}$$

множество функций с двумя аргументами

$$\begin{aligned} \mathrm{F}_2 = (\quad & f_{2,1}(z_1,z_2) = z_1 + z_2,\, f_{2,2}(z_1,z_2) = z_1 z_2, \\ & f_{2,3}(z_1,z_2) = \max(z_1,z_2),\, f_{2,4}(z_1,z_2) = \min(z_1,z_2), \\ & f_{2,5}(z_1,z_2) = z_1 + z_2 - z_1 z_2, \\ & f_{2,6}(z_1,z_2) = \operatorname{sgn}(z_1+z_2)\sqrt{z_1^2+z_2^2},\ ); \end{aligned} \tag{5.23}$$

размерность матрицы сетевого оператора $L = 40$; начальное базисное решение

$$\begin{aligned} u_1(\mathbf{x}) = u_2(\mathbf{x}) &= s_7\left(s_1x_1 + s_4\right) + s_{10} + \\ &+ s_8\left(s_2x_2 + s_5\right) + s_{11} + \\ &+ s_9\left(s_3x_3 + s_6\right) + s_{12}; \end{aligned}$$

начальное значение вектора параметров $\mathbf{s}$

$$\mathbf{s} = (1,\, 1,\, 1,\, 1,\, 1,\, 1,\, 1,\, 1,\, 1,\, 1,\, 1,\, 1)\,.$$

С учетом номеров функций, переменных и параметров из множеств (5.21) — (5.23) матрица сетевого оператора для начального базисного решения будет иметь вид

$$\mathbf{\Psi} = \left(\begin{array}{cccccccccccccccccccccccccccccccccccccccc}
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1
\end{array}\right).$$

Поиск оптимального математического выражения, являющегося решением поставленной задачи синтеза системы управления автомобилеподобным роботом, в форме матрицы сетевого оператора осуществлялся с помощью генетического алгоритма. Параметры генетического алгоритма имели следующие

значения: размер начальной популяции $H = 1024$; число поколений поиска $W = 3200$; вероятность скрещивания $P_c = 0{,}4$; вероятность мутации $P_m = 0{,}7$.

Вычисления проводились на компьютере Intel® Core™ i7 2.8 GHz, 8 GB RAM с использованием авторского комплекса программ, написанного на языке Free Pascal в среде Lazarus. Время поиска решений в рассмотренных примерах составляло от 5 до 14 часов.

5.4 Результаты вычислительного эксперимента

Поиск математического выражения функции управления (5.18) производился в трех независимых экспериментах. Эксперименты отличались друг от друга числом оптимальных траекторий в обучающей выборке N и значением параметра дискретизации оптимальных траекторий по времени $\Delta_s t$.

- **Эксперимент 1:** $N = 9$, $\Delta_s t = 0{,}01$;
- **Эксперимент 2:** $N = 9$, $\Delta_s t = 0{,}05$;
- **Эксперимент 3:** $N = 5$, $\Delta_s t = 0{,}01$.

В **Эксперименте 3** обучающая выборка $\breve{\mathrm{D}}$ являлась подмножеством обучающей выборки D (5.17), используемой в **Экспериментах 1** и **2**,

$$\breve{\mathrm{D}} = \{\mathrm{D}_1, \mathrm{D}_3, \mathrm{D}_5, \mathrm{D}_7, \mathrm{D}_9\}\,,\ \breve{\mathrm{D}} \in \mathrm{D}.$$

Значение параметра дискретизации $\Delta_s t$ влияет на число опорных точек в обучающей выборке для каждой используемой оптимальной траектории. Таким образом, суммарное число точек в обучающей выборке S и время поиска решения T составило

- **Эксперимент 1:** $S = 1311$, $T = 14$ ч. 14 мин.;
- **Эксперимент 2:** $S = 266$, $T = 4$ ч. 53 мин.;
- **Эксперимент 3:** $S = 728$, $T = 5$ ч. 46 мин.

В результате в каждом эксперименте было получено решение поставленной задачи синтеза системы управления автомобилеподобным роботом в форме матрицы сетевого оператора и вектора параметров. Оценка каждого решения производилась по значению функционала качества (5.10) для разных начальных состояний из области X_0 (5.7).

Функционал качества (5.10) для выбранного начального состояния $\mathbf{x}^0 \in \mathrm{X}_0$ совпадает с функционалом качества решения задачи оптимального управления (5.12) для этого же начального состояния. Тогда в качестве эталонного решения в сравнении будет использоваться значение функционала качества, полученное при решении задачи оптимального управления для данного начального состояния $\mathbf{x}^0$.

В разделе 5.3.1 для целей составления обучающей выборки приведено решение задачи оптимального управления автомобилеподобным роботом для 9 начальных условий из множества $\tilde{\mathrm{X}}_0$ (5.11). Полученные при этом значения функционала качества (5.12) показаны в Таблице 3. Данные значения будут использованы при сравнении с решениями, полученными найденными функциями управления. Однако, для адекватной оценки найденных функций управления необходимо произвести сравнение также с теми оптимальными решениями, которых изначально не было в обучающей выборке. Другими словами, нужно также получить оценку найденных функций управления при перемещении объекта управления в терминальное состояние из множества начальных состояний $\hat{\mathrm{X}}_0$, обладающего следующим свойством

$$\hat{\mathrm{X}}_0 \subset \mathrm{X}_0, \quad \hat{\mathrm{X}}_0 \cup \tilde{\mathrm{X}}_0 = \varnothing. \tag{5.24}$$

Выберем элементы множества $\hat{\mathrm{X}}$ таким образом, чтобы область начальных условий (5.7) была покрыта равномерной сеткой с шагом $\Delta = 0{,}5$ и при этом удовлетворялось условие (5.24)

$$\begin{aligned}
\hat{\mathrm{X}}_0 \ = \{ \ & \mathbf{x}^{0,10} = [7{,}5\ 9\ \pi]^T, && \mathbf{x}^{0,11} = [8{,}5\ 9\ \pi]^T, \\
& \mathbf{x}^{0,12} = [7\ 9{,}5\ \pi]^T, && \mathbf{x}^{0,13} = [7{,}5\ 9{,}5\ \pi]^T, \\
& \mathbf{x}^{0,14} = [8\ 9{,}5\ \pi]^T, && \mathbf{x}^{0,15} = [8{,}5\ 9{,}5\ \pi]^T, \\
& \mathbf{x}^{0,16} = [9\ 9{,}5\ \pi]^T, && \mathbf{x}^{0,17} = [7{,}5\ 10\ \pi]^T, \\
& \mathbf{x}^{0,18} = [8{,}5\ 10\ \pi]^T, && \mathbf{x}^{0,19} = [7\ 10{,}5\ \pi]^T, \\
& \mathbf{x}^{0,20} = [7{,}5\ 10{,}5\ \pi]^T, && \mathbf{x}^{0,21} = [8\ 10{,}5\ \pi]^T, \\
& \mathbf{x}^{0,22} = [8{,}5\ 10{,}5\ \pi]^T, && \mathbf{x}^{0,23} = [9\ 10{,}5\ \pi]^T, \\
& \mathbf{x}^{0,24} = [7{,}5\ 11\ \pi]^T, && \mathbf{x}^{0,25} = [8{,}5\ 11\ \pi]^T \ \}.
\end{aligned} \tag{5.25}$$

Для каждого начального состояния из множества (5.25) найдем решение задачи оптимального управления, обеспечивающее минимум функционалу (5.12). Таким образом, эталонные значения функционала качества (5.12) будут известны для решений задачи оптимального управления из всех начальных состояний $\mathbf{x}^0 \in \hat{\mathrm{X}}_0 \cup \tilde{\mathrm{X}}_0$.

Теперь составим множество

$$\check{\mathrm{X}}_0 = \hat{\mathrm{X}}_0 \cup \tilde{\mathrm{X}}_0$$

и используем найденные функции управления для оценки каждого решения по значению функционала качества (5.10) для всех начальных состояний из данного множества.

Для полноты сравнительного анализа, кроме оценок отклонений от эталонных значений, рекомендуется использовать оценку значений среднего отклонения из всех полученных результатов и среднеквадратического отклонения [89].

Полученные с помощью каждого решения значения функционала качества (5.10), эталонные значения, полученные с помощью решения задачи оптимального управления, и значения среднего и среднеквадратического отклонений приведены в Таблице 4.

В Таблице 4 в колонке J_i^1 приведено значение функционала качества (5.10), полученное с помощью найденного решения задачи синтеза в **Эксперименте 1**; в колонке J_i^2 — значение функционала качества, полученное с помощью найденного решения задачи синтеза в **Эксперименте 2**; в колонке J_i^3 — значение функционала качества, полученное с помощью найденного решения задачи синтеза в **Эксперименте 3**; в колонке J_i^{ocp} приведено эталонное значение функционала качества. Лучший среди трех экспериментов результат для каждого начального состояния выделен полужирным шрифтом. В последней строке приведены среднее и среднеквадратическое отклонения от эталонных значений по каждому решению.

В **Эксперименте 1** минимальное отклонение от эталонного значения составило 0,0181, максимальное отклонение — 0,1923, среднее значение отклонения — 0,0715, среднеквадратическое отклонение — 0,0388. В **Эксперименте 2** минимальное отклонение от эталонного значения составило 0,038, максимальное отклонение — 0,3551, среднее значение отклонения — 0,207, среднеквадратическое отклонение — 0,0936. В **Эксперименте 3** минимальное отклонение от эталонного значения составило 0,0073, максимальное отклонение — 0,2246, среднее значение отклонения — 0,0772, среднеквадратическое отклонение — 0,0481.

Из полученных результатов видно, что в проведенном сравнении наилучшие результаты показало решение, полученное в **Эксперименте 1**. Далее с незначительным отставанием следует решение, полученное в **Эксперименте 3**.

Таблица 4 — Сравнительные результаты для найденных решений задачи синтеза системы управления

i	$\mathbf{x}^{0,i}$	J_i^1	J_i^2	J_i^3	J_i^{ocp}
1	$\mathbf{x}^{0,1} = [7\ 9\ \pi]^T$	1,3363	1,3808	**1,3250**	1,2883
2	$\mathbf{x}^{0,2} = [8\ 9\ \pi]^T$	1,4455	**1,4443**	1,4549	1,4063
3	$\mathbf{x}^{0,3} = [9\ 9\ \pi]^T$	**1,5086**	1,5382	1,5446	1,4712
4	$\mathbf{x}^{0,4} = [7\ 10\ \pi]^T$	1,4155	1,6257	**1,4047**	1,3974
5	$\mathbf{x}^{0,5} = [8\ 10\ \pi]^T$	**1,4983**	1,7604	1,5026	1,4182
6	$\mathbf{x}^{0,6} = [9\ 10\ \pi]^T$	**1,6233**	1,8636	1,6642	1,5095
7	$\mathbf{x}^{0,7} = [7\ 11\ \pi]^T$	**1,5255**	1,5371	1,5258	1,4855
8	$\mathbf{x}^{0,8} = [8\ 11\ \pi]^T$	**1,5824**	1,6736	1,5888	1,5097
9	$\mathbf{x}^{0,9} = [9\ 11\ \pi]^T$	**1,7119**	1,7715	1,7230	1,5935
10	$\mathbf{x}^{0,10} = [7{,}5\ 9\ \pi]^T$	**1,3954**	1,5671	1,4008	1,3493
11	$\mathbf{x}^{0,11} = [8{,}5\ 9\ \pi]^T$	1,4995	**1,4914**	1,5107	1,4418
12	$\mathbf{x}^{0,12} = [7\ 9{,}5\ \pi]^T$	1,3804	1,5204	**1,3558**	1,3428
13	$\mathbf{x}^{0,13} = [7{,}5\ 9{,}5\ \pi]^T$	**1,4172**	1,6439	1,4279	1,3815
14	$\mathbf{x}^{0,14} = [8\ 9{,}5\ \pi]^T$	**1,4731**	1,6214	1,4861	1,4119
15	$\mathbf{x}^{0,15} = [8{,}5\ 9{,}5\ \pi]^T$	1,5507	1,6720	**1,5305**	1,4521
16	$\mathbf{x}^{0,16} = [9\ 9{,}5\ \pi]^T$	**1,6819**	1,7229	1,7142	1,4896
17	$\mathbf{x}^{0,17} = [7{,}5\ 10\ \pi]^T$	**1,4484**	1,6933	1,4499	1,4052
18	$\mathbf{x}^{0,18} = [8{,}5\ 10\ \pi]^T$	**1,5531**	1,8117	1,5623	1,4566
19	$\mathbf{x}^{0,19} = [7\ 10{,}5\ \pi]^T$	1,4806	1,6385	**1,4489**	1,4353
20	$\mathbf{x}^{0,20} = [7{,}5\ 10{,}5\ \pi]^T$	**1,5027**	1,7887	1,5669	1,4506
21	$\mathbf{x}^{0,21} = [8\ 10{,}5\ \pi]^T$	**1,5445**	1,7065	1,5453	1,4634
22	$\mathbf{x}^{0,22} = [8{,}5\ 10{,}5\ \pi]^T$	1,5853	1,7555	**1,5830**	1,5043
23	$\mathbf{x}^{0,23} = [9\ 10{,}5\ \pi]^T$	1,6567	1,8040	**1,6566**	1,5545
24	$\mathbf{x}^{0,24} = [7{,}5\ 11\ \pi]^T$	1,5581	1,6794	**1,5513**	1,4965
25	$\mathbf{x}^{0,25} = [8{,}5\ 11\ \pi]^T$	1,6728	1,7228	**1,6672**	1,5457
Среднее значение отклонения		0,0715	0,207	0,0772	–
Среднеквадратическое отклонение		0,0388	0,0936	0,0481	–

Решение, полученное в **Эксперименте 2**, показало наихудший результат, значительно уступив по значениям критерия качества.

Таким образом, экспериментально показано, что значение параметра дискретизации оптимальных траекторий $\Delta_s t$ имеет более существенное влияние на качество поиска решения задачи синтеза чем число оптимальных траекторий в обучающей выборке. Также анализ результатов вычислений в **Эксперименте 1** и **Эксперименте 3** свидетельствует, что дальнейшее увеличение числа оптимальных траекторий в обучающей выборке принесет незначительное улучшение результата при значительном росте времени расчета.

Далее рассмотрим более подробно лучшее решение — решение, полученное в **Эксперименте 1**. С помощью метода сетевого оператора было найдено решение в форме матрицы сетевого оператора $\mathbf{\Psi}$ и вектора оптимальных параметров $\mathbf{s}$. Оптимальная форма матрицы сетевого оператора имела следующий вид

$$\mathbf{\Psi} = \begin{pmatrix}
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&4&0&0&0&0&2&0&0&0&0&0&0&0&0&0&0&0&12&0&0&0&0&6&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&6&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&9&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&10&0&4&0&0&0&0&0&0&0&0&0&0&0&7&11&0&0&8&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&9&0&5&0&0&0&0&0&0&4&8&0&0&0&0&0&0&3&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&2&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&8&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&4&0&8&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&7&4&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&5&0&0&0&0&0&0&0&0&5&10&0&0&10&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&10&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&8&0&0&0&0&0&11&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&8&0&0&0&0&0&1&0&0&0&0&0&0&0&7&0&0&0&0&2&4&2&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2&5&10&0&7&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&7&0&0&0&0&0&0&0&0&0&2&0&0&0&0&0&0&7&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&10&1&0&3&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&5&7&0&1&0&0&2&0&0&0&0&0&0&0&5&0&12&11&0&7&10&3&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&6&0&0&1&0&1&0&0&0&0&5&0&0&0&0&5&0&0&0&0&0&0&0&8\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&5&7&0&1&0&0&0&8&9&0&0&0&0&0&0&0&0&0&6&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&6&0&0&11&8&0&0&0&0&0&0&0&0&0&3&0&11&0&0&0&8\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&4&1&0&1&0&0&0&0&0&8&0&0&0&0&3&0&0&0&0&8\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&9&0&0&8&0&0&0&11&0&5&0&0&11&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&4&0&0&1&1&0&0&0&0&0&0&10&0&0&0&0&6&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2&0&0&1&0&0&0&8&0&0&11&0&7&6&1&0&9&5\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2&9&0&1&4&6&0&0&0&0&6&4&1&0&0&8&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&5&1&5&0&0&0&0&11&0&0&0&12&0&5&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&8&0&5&0&12&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&0&0&0&0&6&0&9&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&3&1&0&0&0&0&5&8&4&1&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&5&1&9&8&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2&1&8&0&0&7&0&2&9&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&2&0&0&12&4&0&9&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&9&0&0&0&5&0&4&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&5&0&6&0&8\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&6&2&0&6&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&6&0&8&0&6&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&3&1&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&3&10&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&3&5&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&6&8\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&2
\end{pmatrix}. \tag{5.26}$$

Найденные оптимальные значения вектора параметров $\mathbf{s}$

$$\begin{aligned}\mathbf{s} = \; & (3{,}246338,\ 5{,}993164,\ 15{,}050049,\ 13{,}058105,\\ & 12{,}14209,\ 10{,}884277,\ 12{,}912842,\ 12{,}394775,\\ & 7{,}228271,\ 3{,}997803,\ 0{,}548096,\ 6{,}888184)\,.\end{aligned} \tag{5.27}$$

Используя выражения (2.39) — (2.41) для определения композиции функций в сетевом операторе (5.26) и с учётом найденных оптимальных значений вектора параметров (5.27) и имеющихся ограничений на управление (5.6), получим следующее математическое выражение для найденной функции управления автомобилеподобным роботом

$$u_1(\mathbf{x}) = \begin{cases} u_1^-, \text{ если } \tilde{u}_1 < u_1^- \\ u_1^+, \text{ если } \tilde{u}_1 > u_1^+ \\ \tilde{u}_1 \text{ — иначе} \end{cases}, \qquad u_2(\mathbf{x}) = \begin{cases} u_2^-, \text{ если } \tilde{u}_2 < u_2^- \\ u_2^+, \text{ если } \tilde{u}_2 > u_2^+ \\ \tilde{u}_2 \text{ — иначе} \end{cases}, \tag{5.28}$$

где

$$\begin{aligned}
\tilde{u}_1 &= f_{2,6}\left(Z_1^{-1},\, \mathrm{e}^{Z_4},\, \operatorname{sgn}(Z_7)\sqrt{|Z_7|},\, \arctan(Z_8),\, \frac{1-\mathrm{e}^{-Z_{15}}}{1+\mathrm{e}^{-Z_{15}}},\right.\\
&\qquad \left.\arctan(Z_{16}),\, s_9^2,\, s_6^3,\, \frac{1-\mathrm{e}^{-s_3}}{1+\mathrm{e}^{-s_3}},\, \mathrm{e}^{x_1}\right),\\
\tilde{u}_2 &= \frac{1-\mathrm{e}^{-\tilde{u}_1}}{1+\mathrm{e}^{-\tilde{u}_1}}\,\frac{1-\mathrm{e}^{-Z_6}}{1+\mathrm{e}^{-Z_6}}\,Z_7\,Z_{16}^{-1}\,\frac{1-\mathrm{e}^{-Z_{19}}}{1+\mathrm{e}^{-Z_{19}}}\,\frac{1-\mathrm{e}^{-Z_{20}}}{1+\mathrm{e}^{-Z_{20}}}\,\frac{1-\mathrm{e}^{-Z_{22}}}{1+\mathrm{e}^{-Z_{22}}},\\
Z_1 &= \min\left(Z_2^3,\, Z_3,\, \mathrm{e}^{Z_6},\, \arctan(Z_9),\, Z_{11},\, Z_{14}^{-1},\, \mathrm{e}^{Z_{17}},\, -Z_{23},\, \operatorname{sgn}(s_9)\sqrt{|s_9|}\right),\\
Z_2 &= \min\left(Z_3,\, \frac{1-\mathrm{e}^{-Z_4}}{1+\mathrm{e}^{-Z_4}},\, \mathrm{e}^{Z_5},\, Z_7^{-1},\, \operatorname{sgn}(Z_8)\sqrt{|Z_8|},\, Z_9^2,\, Z_{11},\, Z_{16},\right.\\
&\qquad \sqrt[3]{Z_{18}},\, Z_{23}^3,\, -s_{12},\, s_{11},\, \ln(|s_{10}|),\, s_9^2,\, \operatorname{sgn}(s_4)\sqrt{|s_4|},\\
&\qquad \left.\operatorname{sgn}(s_3)\sqrt{|s_3|},\, \frac{1-\mathrm{e}^{-x_3}}{1+\mathrm{e}^{-x_3}}\right),\\
Z_3 &= \min\left(Z_6^{-1},\, Z_8 - Z_8^3,\, \operatorname{sgn}(Z_{11})\sqrt{|Z_{11}|},\, \arctan(Z_{12}),\, Z_{14} - Z_{14}^3,\right.\\
&\qquad \left.Z_{15},\, \mathrm{e}^{Z_{16}},\, \sqrt[3]{Z_{20}},\, \mathrm{e}^{Z_{21}},\, \ln(|Z_{23}|),\, \sqrt[3]{s_8},\, s_6^3,\, \ln(|s_4|)\right),\\
Z_4 &= f_{2,6}\left(Z_5^2,\, \ln(|Z_9|),\, \frac{1-\mathrm{e}^{-Z_{10}}}{1+\mathrm{e}^{-Z_{10}}},\, \frac{1-\mathrm{e}^{-Z_{11}}}{1+\mathrm{e}^{-Z_{11}}},\, Z_{13} - Z_{13}^3,\, \operatorname{sgn}(Z_{15})\sqrt{|Z_{15}|},\right.\\
&\qquad \left.\ln(|Z_{16}|),\, -Z_{19},\, s_{12},\, \ln(|s_{11}|),\, s_{10}^3,\, s_7,\, s_6^{-1},\, \arctan(x_2)\right),\\
Z_5 &= f_{2,6}\left(\arctan(Z_{10}),\, Z_{11}^{-1},\, \mathrm{e}^{Z_{12}},\, \mathrm{e}^{Z_{15}},\, Z_{18}^{-1},\, -Z_{20},\right.\\
&\qquad \left.\sqrt[3]{Z_{23}},\, s_{12}^3,\, s_{10}^{-1},\, -s_1,\, \sqrt[3]{x_3},\, x_1 - x_1^3\right),
\end{aligned}$$

$$Z_6 = \arctan(Z_7) + Z_8^2 + Z_{10} + Z_{13}^{-1} + \sqrt[3]{Z_{16}} + Z_{17}^3 + Z_{23} - Z_{23}^3 + s_{10}^2 + \ln(|x_3|),$$

$$Z_7 = Z_8 + \frac{1-\mathrm{e}^{-Z_9}}{1+\mathrm{e}^{-Z_9}} + Z_{10}^{-1} + \sqrt[3]{Z_{14}} + \sqrt[3]{Z_{18}} + Z_{22}^{-1} + \ln(|s_9|) + x_2,$$

$$Z_8 = Z_9 + \frac{1-\mathrm{e}^{-Z_{13}}}{1+\mathrm{e}^{-Z_{13}}} + Z_{23}^{-1},$$

$$Z_9 = Z_{10}\,\frac{1-\mathrm{e}^{-Z_{16}}}{1+\mathrm{e}^{-Z_{16}}}\,\frac{1-\mathrm{e}^{-Z_{19}}}{1+\mathrm{e}^{-Z_{19}}}\,\frac{1-\mathrm{e}^{-s_8}}{1+\mathrm{e}^{-s_8}}, \qquad Z_{10} = Z_{11} + Z_{12} + \mathrm{e}^{Z_{15}},$$

$$Z_{11} = \min\left(Z_{12},\, Z_{13},\, \mathrm{sgn}(Z_{15})\sqrt{|Z_{15}|},\, \frac{1-\mathrm{e}^{-Z_{18}}}{1+\mathrm{e}^{-Z_{18}}},\, s_{11}^2,\, s_2^2\right),$$

$$Z_{12} = Z_{14}^{-1} + Z_{15} + Z_{22}^{-1} + s_{12} + \frac{1-\mathrm{e}^{-s_1}}{1+\mathrm{e}^{-s_1}},$$

$$Z_{13} = Z_{14} + Z_{16} + Z_{17} + \arctan(Z_{21}) + s_6^{-1} + \mathrm{sgn}(s_1)\sqrt{|s_1|},$$

$$Z_{14} = f_{2,5}\left(\arctan(Z_{15}),\, Z_{17},\, \arctan(Z_{18}),\, \frac{1-\mathrm{e}^{-Z_{21}}}{1+\mathrm{e}^{-Z_{21}}}\right),$$

$$Z_{15} = Z_{18}\,s_9, \qquad Z_{16} = Z_{19}\,\frac{1-\mathrm{e}^{-Z_{20}}}{1+\mathrm{e}^{-Z_{20}}}\,s_8,$$

$$Z_{17} = \max\left(\sqrt[3]{Z_{20}},\, Z_{22},\, Z_{23}^2,\, s_7^3,\, \mathrm{sgn}(x_3)\sqrt{|x_3|},\, x_1^2\right),$$

$$Z_{18} = Z_{19} + Z_{21} + s_6, \qquad Z_{19} = \max\left(Z_{22},\, s_5,\, x_3^3\right),$$

$$Z_{20} = f_{2,6}\left(\ln(|Z_{21}|),\, Z_{23},\, s_4,\, s_1^{-1}\right), \qquad Z_{21} = f_{2,5}(s_3,\, x_3),$$

$$Z_{22} = f_{2,6}\left(\ln(|Z_{23}|),\, \ln(|s_{11}|),\, \frac{1-\mathrm{e}^{-s_9}}{1+\mathrm{e}^{-s_9}},\, \frac{1-\mathrm{e}^{-s_3}}{1+\mathrm{e}^{-s_3}},\right.$$
$$\left. s_2,\, \arctan(s_1),\, \mathrm{e}^{x_2},\, \mathrm{sgn}(x_1)\sqrt{|x_1|}\right),$$

$$Z_{23} = f_{2,5}(s_1,\, x_1),$$

$f_{2,5}$ и $f_{2,6}$ — функции из множества (5.23), обладающие свойствами коммутативности и ассоциативности, s_i, $i = \overline{1,12}$ — элементы найденного вектора параметров (5.27).

Для любого начального состояния из ограниченной области (5.7) и в соответствии с текущим фазовым состоянием автомобилеподобного робота математическое выражение многомерной функции управления (5.28) позволяет получить управляющее воздействие, перемещающее робота в терминальное состояние (5.8) по траектории близкой к оптимальной.

Качество получаемых с помощью выражения (5.28) траекторий определяется их отклонением от эталонных значений. Среднеквадратическое отклонение

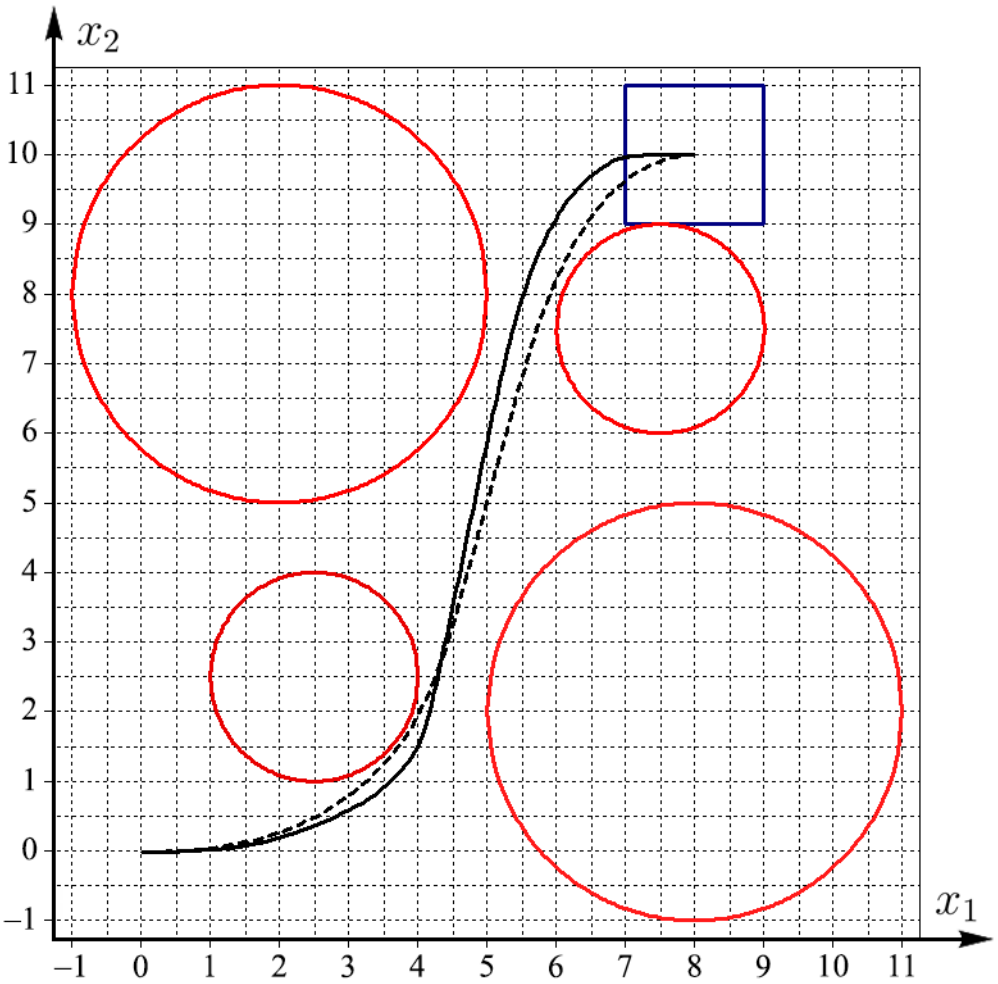


Рисунок 5.3 — Траектория движения автомобилеподобного робота на плоскости, полученная с помощью найденной функции управления (сплошная линия) и эталонная траектория (пунктирная линия).

для полученного решения составляет 0,0388, что свидетельствует о высокой степени близости получаемых с помощью данной функции управления траекторий к оптимальным. Таким образом, задача синтеза системы управления автомобилеподобным роботом считается решенной.

Проиллюстрируем полученные результаты графически. В качестве примера рассмотрим перемещение автомобилеподобного робота из начального состояния $\mathbf{x}^0 = [8\ 10\ \pi]^T$ в терминальное состояние (5.8).

На Рис. 5.3 сплошной линией представлена траектория движения робота на плоскости, полученная с помощью функции управления (5.28). Для сравнения пунктирной линией представлена оптимальная траектория, полученная путём решения задачи оптимального управления. Из Рис. 5.3 видно, что траектория движения робота не нарушает фазовых ограничений (области ограничений показаны красным) и близка к эталонной.

На Рис. 5.4 представлены графики изменения фазовых координат робота во времени, а на Рис. 5.5 — графики изменения управляющих воздействий u_1 и u_2 во времени.

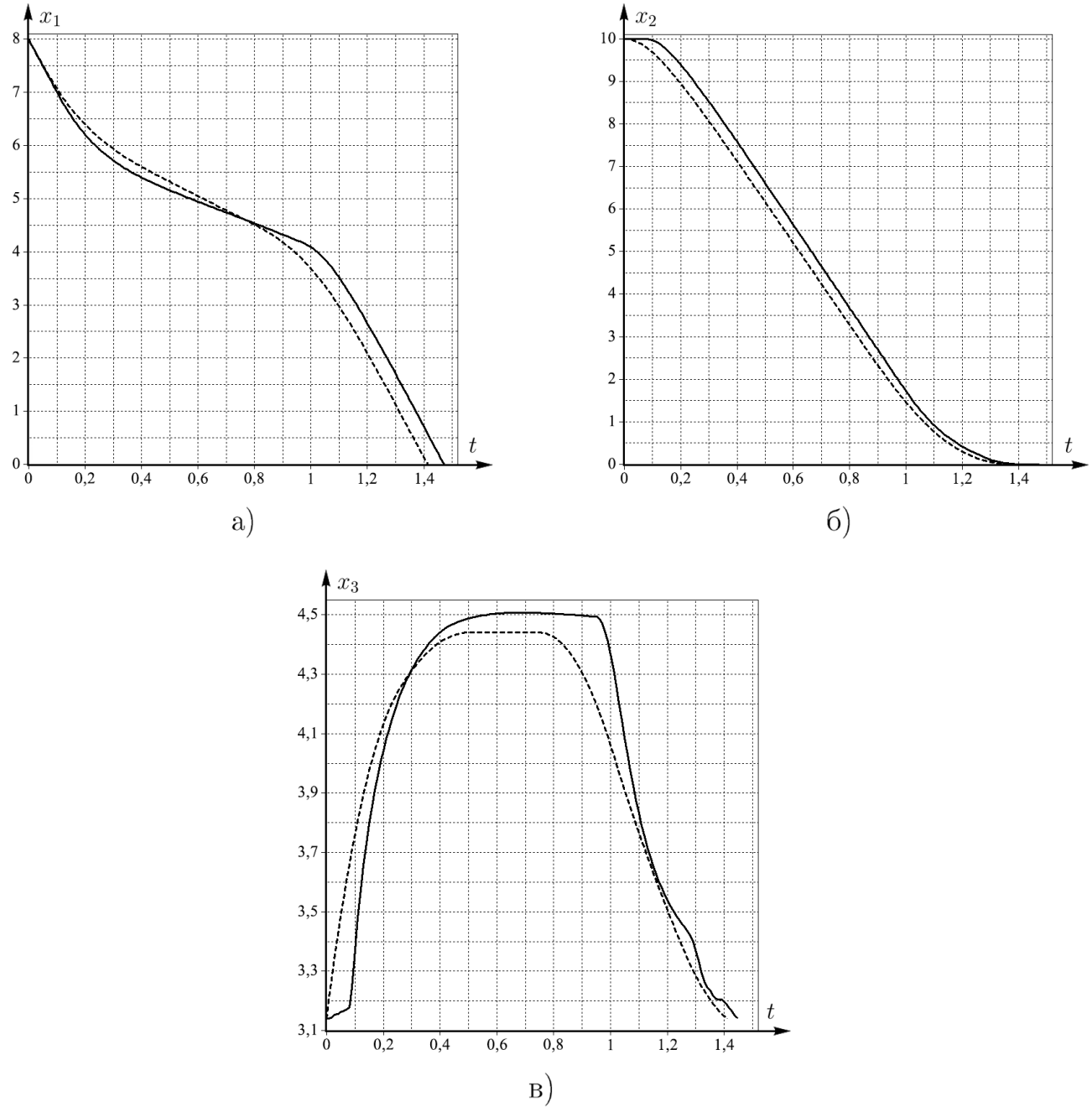
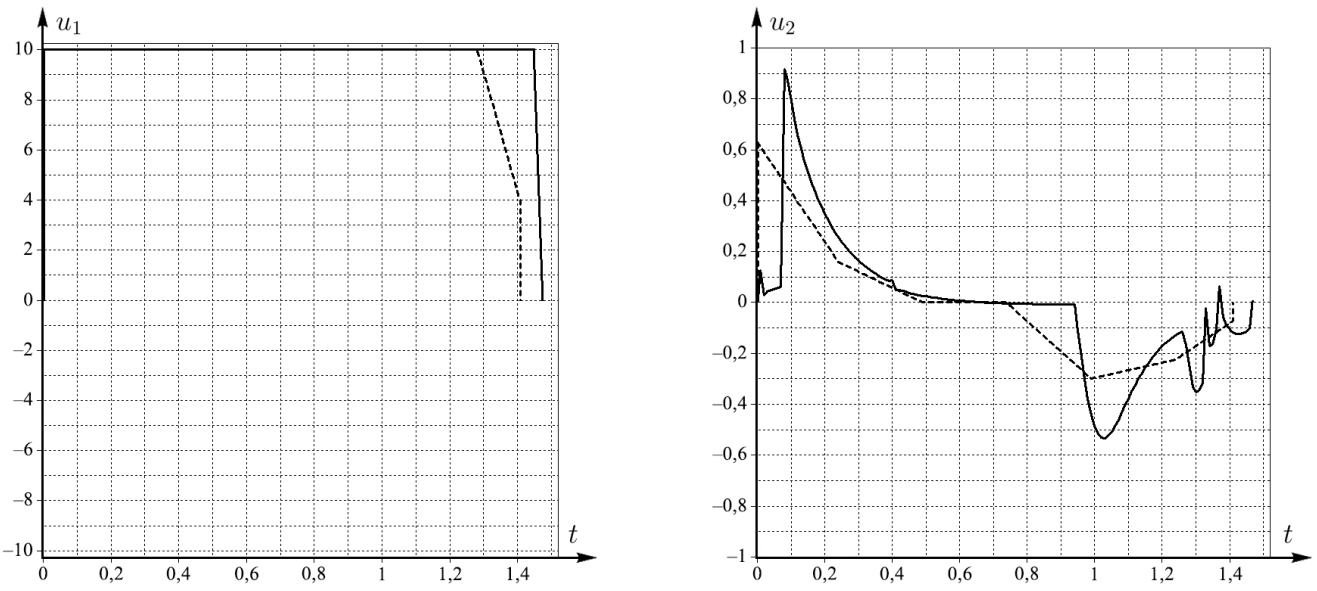


Рисунок 5.4 — Графики изменения фазовых координат автомобилеподобного робота во времени: а) — график $x_1(t)$; б) — график $x_2(t)$; в) — график $x_3(t)$; сплошная линия — полученное значение; пунктирная линия — эталонное значение.

Для рассмотренного примера графики на Рис. 5.3 — 5.5 подтвердили высокое качество найденной функции управления. Однако, информация об оптимальной траектории движения из начального состояния $\mathbf{x}^0 = [8\ \ 10\ \ \pi]^T$,

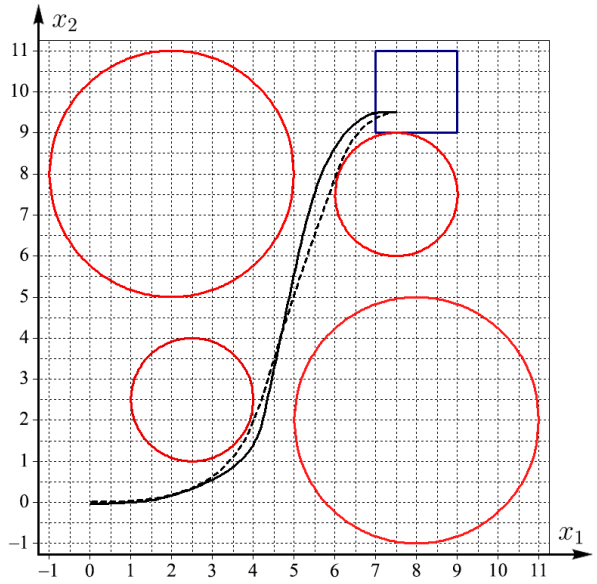


а) б)

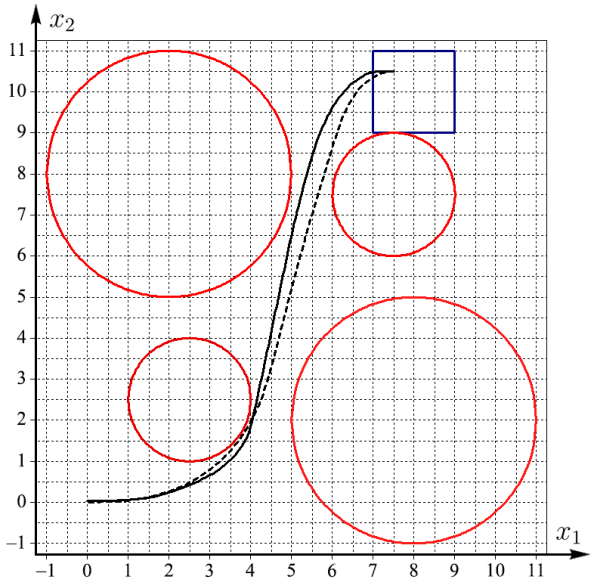
Рисунок 5.5 — Графики изменения управляющих воздействий во времени: а) — график $u_1(t)$; б) — график $u_2(t)$; сплошная линия — полученное значение; пунктирная линия — эталонное значение.

рассмотренного в примере, использовалась в обучающей выборке. Поэтому есть смысл дополнительно проиллюстрировать работу найденной функции управления для таких начальных состояний робота, по движению из которых в обучающей выборке данных не было. На Рис. 5.6 представлены 4 графика движения автомобилеподобного робота на плоскости из начальных состояний $\mathbf{x}^0 = [7{,}5\ 9{,}5\ \pi]^T$, $\mathbf{x}^0 = [7{,}5\ 10{,}5\ \pi]^T$, $\mathbf{x}^0 = [8{,}5\ 9{,}5\ \pi]^T$ и $\mathbf{x}^0 = [8{,}5\ 10{,}5\ \pi]^T$. Информации об оптимальной траектории движения из этих начальных состояний в обучающей выборке не было. При этом данные начальные состояния равноудалены от всех начальных состояний, для которых информация об оптимальной траектории имелась.

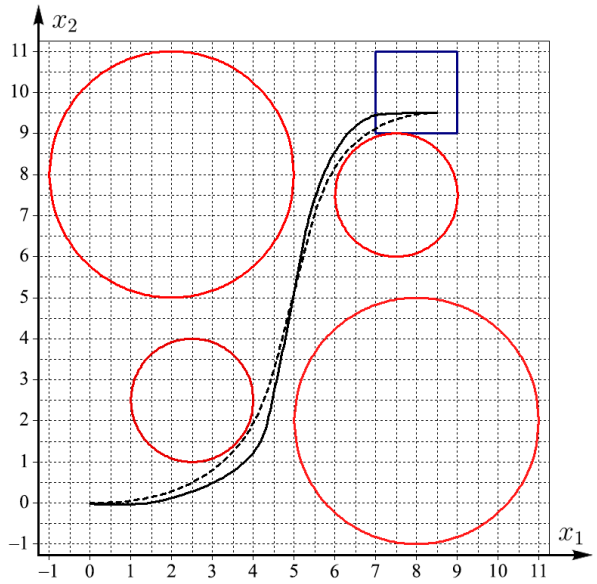
Полученные с помощью многомерной функции (5.28) траектории (сплошная линия) не нарушают фазовых ограничений и близки к оптимальным (пунктирная линия). Достижение терминального состояния производится с заданной точностью. Графики на Рис. 5.6 также подтвердили высокое качество найденной функции управления и предложенного метода решения задачи численного синтеза системы управления в целом.



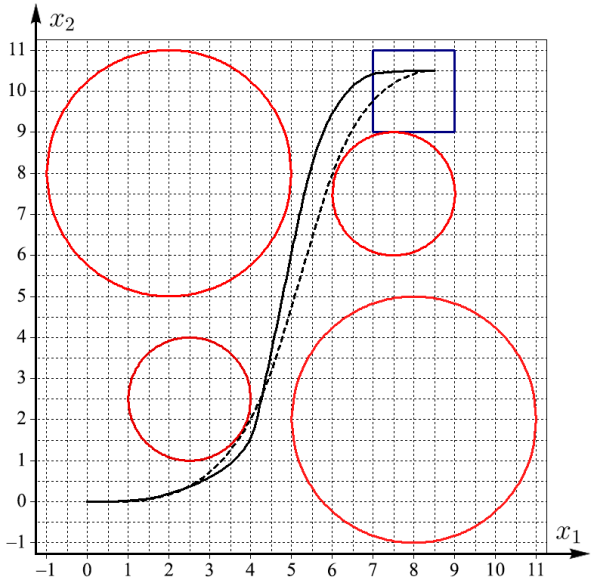
а)



б)



в)



г)

Рисунок 5.6 — Траектории движения автомобилеподобного робота на плоскости, полученные с помощью найденной функции управления (сплошная линия), и эталонные траектории (пунктирная линия): а) — начальное состояние $\mathbf{x}^0 = [7{,}5\ 9{,}5\ \pi]^T$; б) — начальное состояние $\mathbf{x}^0 = [7{,}5\ \ 10{,}5\ \ \pi]^T$; в) — начальное состояние $\mathbf{x}^0 = [8{,}5\ \ 9{,}5\ \ \pi]^T$; г) — начальное состояние $\mathbf{x}^0 = [8{,}5\ 10{,}5\ \pi]^T$.

Выводы по Главе 5

Современным прикладным задачам синтеза системы управления свойственна высокая сложность. Также следует учитывать возможные ограничения, накладываемые на фазовое состояние объекта управления. Поиск решения для таких задач требует высокой эффективности применяемых методов.

Предложенный в диссертации метод синтеза системы управления на основе аппроксимации множества оптимальных траекторий был протестирован на прикладной задаче синтеза системы управления автомобилеподобным роботом.

В соответствии с алгоритмом данного метода, на первом этапе была многократно решена задача оптимального управления для разных начальных условий и получены оптимальные траектории. Для решения задачи оптимального управления использовался прямой подход, заключающийся в редукции исходной задачи к задаче нелинейного программирования и её последующего решения новым гибридным алгоритмом. Результатом выполнения первого этапа стало получение обучающей выборки на основе множества найденных оптимальных траекторий.

На втором этапе производился непосредственно синтез системы управления. Для этого использовался метод сетевого оператора, относящийся к классу методов символьной регрессии. Поиск функции управления осуществлялся путем аппроксимации данных из обучающей выборки. Для оценки качества аппроксимации использовался функционал, учитывающий фазовое состояние робота, а также факт нарушения фазовых ограничений и степень достижения терминального состояния. Таким образом, при поиске использовались принципы обучения с подкреплением.

Результатом вычислительного эксперимента стало получение многомерной функции управления в форме математического выражения. Сравнительный анализ на основе 25 траекторий показал, что найденная функция позволяет для любого начального состояния из ограниченной области получать управления, перемещающие робота в терминальное состояние по близкой к оптимальной траектории. Среднеквадратическое отклонение от эталонных значений составило 0,0388. Данные результаты позволили утверждать об успешном решении прикладной задачи синтеза системы управления автомобилеподобным роботом предложенным методом.

Заключение

В диссертации предложен новый численный подход к решению задачи синтеза системы управления. Поиск структуры функции управления в новом подходе осуществляется с помощью методов символьной регрессии путем аппроксимации множества предварительно найденных оптимальных траекторий. При оценке качества аппроксимации учитывается совокупность факторов, в том числе текущее и предыдущее положение объекта в фазовом пространстве, степень достижения терминального состояния и, при наличии фазовых ограничений, контроль их нарушения. Эффективность предложенного в диссертации подхода показана на примере решения прикладной задачи синтеза системы управления автомобилеподобным роботом, а также результатами других исследований, опубликованных в рецензируемых научных изданиях.

Основные результаты диссертации заключаются в следующем.

1. Рассмотрена задача общего синтеза системы управления. Показано, что в настоящее время нет универсальных подходов к решению данной задачи, а существующие аналитические методы в основном применимы только для несложных объектов малой размерности. Известный численный метод синтеза системы управления на основе многокритериальной структурно-параметрической оптимизации методами символьной регрессии также содержит ряд недостатков, основным из которых является невозможность оценки близости найденного решения к оптимальному.
2. Разработан и предложен численный метод решения задачи синтеза системы управления на основе аппроксимации множества оптимальных траекторий. Поиск решения предложенным методом осуществляется в два этапа. На первом этапе многократно решается задача оптимального управления для разных начальных условий и формируется обучающая выборка на основе найденных оптимальных траекторий. На втором этапе осуществляется поиск математического выражения функции управления от координат состояния, аппроксимирующего данные из обучающей выборки.
3. Для реализации первого этапа предложено использовать прямой подход решения задачи оптимального управления, заключающийся в

редукции исходной задачи к задаче нелинейного программирования и её решения численным методом оптимизации. Экспериментально показано, что среди известных алгоритмов численной оптимизации, применительно к прикладным задачам оптимального управления наиболее эффективными оказались эволюционные алгоритмы.

4. Для решения задачи оптимального управления разработан и предложен гибридный алгоритм. В основу предложенного метода легли алгоритм серых волков и пчелиный алгоритм.
5. Для реализации второго этапа предложено использовать методы символьной регрессии. Для оценки качества аппроксимации использованы принципы обучения с подкреплением. Построен функционал качества, учитывающий информацию о фазовом состоянии объекта управления и факт нарушения фазовых ограничений.
6. Показано, что предложенный метод для найденного решения задачи синтеза системы управления позволяет оценить его близость к оптимальному решению.
7. Разработан комплекс программ, реализующих предложенный численный метод решения задачи синтеза системы управления на основе аппроксимации множества оптимальных траекторий. Произведен поиск решения прикладной задачи синтеза системы управления автомобилеподобным роботом в пространстве с фазовыми ограничениями. Полученные результаты подтвердили эффективность предложенного метода.

Автор диссертации выражает благодарность и большую признательность научному руководителю Асхату Ибрагимовичу Дивееву за научное руководство, поддержку, помощь и обсуждение результатов. Также автор благодарит соавторов научных публикаций Асхата Ибрагимовича Дивеева, Елену Анатольевну Софронову и Елизавету Юрьевну Шмалько за сотрудничество и обсуждение научных результатов.

Часть научных исследований, которые легли в основу настоящей диссертации, была выполнена при поддержке проекта 075-15-2020-799 «Методы построения и моделирования сложных систем на основе интеллектуальных и суперкомпьютерных технологий, направленные на преодоление больших вызовов» Минобрнауки России. Автор диссертации выражает благодарность научному коллективу данного проекта.

Список литературы

1. *Арутюнов А. В.*, *Магарил-Ильяев Г. Г.*, *Тихомиров В. М.* Принцип максимума Понтрягина. Доказательство и приложения. — М. : Факториал Пресс, 2006.
2. *Атанс М.*, *Фалб П. Л.* Оптимальное управление. — М. : Машиностроение, 1968. — 764 с.
3. *Афанасьев В. Н.* Оптимальные системы управления. Аналитическое конструирование. — М. : РУДН, 2007. — 259 с.
4. *Афанасьев В. Н.*, *Колмановский В. Б.*, *Носов В. Р.* Математическая теория конструирования систем управления. — М. : Высш. шк., 2003. — 614 с.
5. *Базара М.*, *Шетти К.* Нелинейное программирование. Теория и алгоритмы. — М. : Мир, 1982. — 584 с.
6. *Барбашин Е. А.* Введение в теорию устойчивости. — М. : Наука, 1967. — 224 с.
7. *Барбашин Е. А.* Функции Ляпунова. — М. : Наука, 1970. — 240 с.
8. *Бахарев А. Т.*, *Зуев А. К.*, *Камилов М. М.* Теория и применение случайного поиска. — Рига : Зинатне, 1969. — 309 с.
9. *Беллман Р.* Динамическое программирование. — М. : Издательство иностранной литературы, 1963.
10. *Беллман Р.*, *Гликсберг И.*, *Гросс О.* Некоторые вопросы математической теории процессов управления. — М. : Издательство иностранной литературы, 1962. — 336 с.
11. *Болтянский В. Г.* Достаточные условия оптимальности и обоснование метода динамического программирования // Изв. АН СССР. Сер. матем. — 1964. — Т. 28, № 3. — С. 481—514.
12. *Болтянский В. Г.* Математические методы оптимального управления. — М. : Наука, 1968.
13. *Болтянский В. Г.* Оптимальное управление дискретными системами. — М. : Наука, 1973.
14. *Болтянский В. Г.*, *Насритдинов Г.* Синтез оптимальных управлений в нелинейных колебательных системах второго порядка // Дифференциальные уравнения. — 1967. — Т. 3, № 3. — С. 380—394.

15. *Буков В. Н.* Адаптивные прогнозирующие системы управления полетом. — М. : Наука, 1987. — 232 с.
16. *Ванько В. И.*, *Ермошина О. В.*, *Кувыркин Г. Н.* Вариационное исчисление и оптимальное управление. — М. : Изд-во МГТУ им. Н.Э. Баумана, 2018. — 488 с.
17. *Васильев Ф. П.* Численные методы решения экстремальных задач. — М. : Наука, 1988. — 550 с.
18. *Грачев Н. И.*, *Евтушенко Ю. Г.* Библиотека программ для решения задач оптимального управления // Ж. вычисл. матем. и матем. физ. — 1979. — Т. 19, № 2. — С. 367—387.
19. *Гришин А. А.*, *Карпенко А. П.* Исследование эффективности метода пчелиного роя в задаче глобальной оптимизации [Электронный ресурс] // Наука и образование: научное издание МГТУ им. Н.Э. Баумана. — 2010. — № 8. — С. 1—28. — URL: http://engineering-science.ru/doc/154050.html (дата обр. 12.04.2022).
20. *Деменков Н. П.*, *Микрин Е. А.* Управление в технических системах. — М. : Издательство МГТУ им. Н.Э. Баумана, 2017. — 452 с.
21. *Дивеев А. И.* Метод сетевого опреатора. — М. : ВЦ РАН, 2010. — 178 с.
22. *Дивеев А. И.* Приближенные методы решения задачи синтеза оптимального управления. — М. : ВЦ РАН, 2015. — 184 с.
23. *Дивеев А. И.* Численные методы решения задачи синтеза управления. — М. : РУДН, 2019. — 192 с.
24. *Дивеев А. И.*, *Пупков К. А.*, *Софронова Е. А.* Синтез системы управления — задача тысячелетия // Вестник Российского университета дружбы народов. Серия: Инженерные исследования. — 2011. — № 2. — С. 113—125.
25. *Евтушенко Ю. Г.* Численные методы решения задач нелинейного программирования // Ж. вычисл. матем. и матем. физ. — 1976. — Т. 16, № 2. — С. 307—324.
26. *Евтушенко Ю. Г.* Методы решения экстремальных задач и их применение в системах оптимизации. — М. : Наука, 1982. — 432 с.
27. *Евтушенко Ю. Г.* Оптимизация и быстрое автоматическое дифференцирование. — М. : ВЦ РАН, 2013. — 144 с.
28. *Заболотнов Ю. М.*, *Лобанков А. А.* К задаче об оптимальной стабилизации углового движения малого космического аппарата при развёртывании орбитальной тросовой системы // Вестник Самарского университета.

Аэрокосмическая техника, технологии и машиностроение. — 2016. — Т. 15, № 1. — С. 46—54.

29. *Зубов В. И.* Устойчивость движения. — М. : Высшая школа, 1984. — 229 с.
30. *Калман Р.*, *Фалб П.*, *Арбиб М.* Очерки по математической теории систем. — М. : УРСС, 2010. — 400 с.
31. *Карманов В. Г.* Математическое программирование. — М. : ФИЗМАТЛИТ, 2008. — 264 с.
32. *Карпенко А. П.* Популяционные алгоритмы глобальной поисковой оптимизации. Обзор новых и малоизвестных алгоритмов // Информационные технологии, Приложение. — 2012. — № 7. — С. 1—32.
33. *Карпенко А. П.* Современные алгоритмы поисковой оптимизации. Алгоритмы, вдохновленные природой. — М. : Издательство МГТУ им. Н.Э. Баумана, 2014. — 448 с.
34. *Карпенко А. П.*, *Селиверстов Е. Ю.* Глобальная оптимизация методом роя частиц. Обзор // Информационные технологии. — 2010. — № 2. — С. 25—34.
35. *Ким Д. П.* Теория автоматического управления. Т.2. Многомерные, нелинейные, оптимальные и адаптивные системы. — М. : ФИЗМАТЛИТ, 2004. — 464 с.
36. *Колесников А. А.* Аналитический синтез нелинейных систем, оптимальных относительно линейных агрегированных переменных // Известия вузов. Электромеханика. — 1985. — № 11. — С. 9—18.
37. *Колесников А. А.* Аналитическое конструирование нелинейных агрегированных регуляторов по заданной совокупности инвариантных многообразий. I. Скалярное уравнение // Известия вузов. Электромеханика. — 1987. — № 3. — С. 100—108.
38. *Колесников А. А.* Аналитическое конструирование нелинейных агрегированных регуляторов по заданной совокупности инвариантных многообразий. II. Векторное уравнение // Известия вузов. Электромеханика. — 1987. — № 5. — С. 5—17.
39. *Колесников А. А.* Последовательная оптимизация нелинейных агрегированных систем. — М. : Энергоатомиздат, 1987. — 160 с.
40. *Колесников А. А.*, *Веселов Г. Е.*, *Кузьменко А. А.* Новые технологии проектирования современных систем управления процессами генерирования электроэнергии. — М. : МЭИ, 2016. — 280 с.

41. *Колесников А. А.*, *Колесников А. А.*, *Кузьменко А. А.* Методы АКАР и АКОР в задачах синтеза нелинейных систем управления // Мехатроника, автоматизация, управление. — 2016. — Т. 17, № 10. — С. 657—669.
42. *Красовский А. А.* Системы автоматического управления полетом и их аналитическое конструирование. — М. : Наука, 1973. — 560 с.
43. *Красовский А. А.*, *Поспелов Г. С.* Основы автоматики и технической кибернетики. — М.–Л. : Госэнергоиздат, 1962. — 600 с.
44. *Кунцевич В. М.*, *Лычак М. М.* Синтез систем автоматического управления с помощью функций Ляпунова. — М. : Наука, 1977. — 400 с.
45. *Лапшин В. П.*, *Туркин И. А.*, *Христофорова В. В.* Пример оценки близости управлений, синтезированных на основе принципа максимума и метода АКАР // Вестник Донского государственного технического университета. — 2018. — Т. 18, № 4. — С. 439—449.
46. *Летов А. М.* Математическая теория процессов управления. — М. : Наука, 1981. — 256 с.
47. *Ли Э. Б.*, *Маркус Л.* Основы теории оптимального управления. — М. : Наука, 1972. — 578 с.
48. *Ловчаков В. И.*, *Ловчаков Е. В.*, *Кретов Е. И.* Синтез быстродействующих систем управления с использованием теории аналитического конструирования оптимальных регуляторов // Мехатроника, автоматизация, управление. — 2016. — Т. 17, № 2. — С. 84—93.
49. *Ловчаков В. И.*, *Сухинин Б. В.*, *Сурков В. В.* Оптимальное управление электротехническими объектами. — Тула : ТулГУ, 2004. — 149 с.
50. *Лурье А. И.* Некоторые нелинейные задачи теории автоматического регулирования. — М.-Л. : ГИТТЛ, 1951. — 216 с.
51. Математическая теория оптимальных процессов. 4-е изд. / Л. С. Понтрягин [и др.]. — М. : Наука, 1983. — 392 с.
52. *Моисеев Н. Н.* Численные методы в теории оптимальных систем. — М. : Наука, 1971. — 424 с.
53. *Моисеев Н. Н.* Оптимизация и управление (эволюция идей и перспективы) // Техническая кибернетика. — 1974. — № 4. — С. 3—16.
54. *Моисеев Н. Н.* Методы динамического программирования в теории оптимальных управлений. I // Ж. вычисл. матем. и матем. физ. — 1964. — Т. 4, № 3. — С. 485—494.

55. *Моисеев Н. Н.* Асимптотические методы нелинейной механики. — М. : Наука, 1981. — 380 с.
56. *Пантелеев А. В.*, *Летова Т. А.* Методы оптимизации в примерах и задачах. — М. : Высшая школа, 2005. — 544 с.
57. *Пантелеев А. В.*, *Родионова Д. А.* Применение итерационного динамического программирования в задачах синтеза оптимального управления с полной обратной связью // Научный вестник Московского государственного технического университета гражданской авиации. — 2016. — Т. 226, № 2. — С. 5—13.
58. *Пестерев А. В.* Синтез линеаризующего управления в задаче стабилизации движения автомобилеподобного робота вдоль криволинейного пути // Известия РАН. Теория и системы управления. — 2013. — № 5. — С. 153—165.
59. *Поллак Э.* Численные методы оптимизации. Единый подход. — М. : Мир, 1974. — 374 с.
60. *Пшихопов В. Ч.*, *Медведев М. Ю.* Синтез систем управления подводными аппаратами с нелинейными характеристиками исполнительных органов // Известия ЮФУ. Технические науки. — 2011. — Т. 3, № 116. — С. 147—156.
61. *Рагимов А. Б.* Об одном подходе к решению задач оптимального управления на классах кусочно-постоянных, кусочно-линейных и кусочно-заданных функций // Вестн. Том. гос. ун-та. Управление, вычислительная техника и информатика. — 2012. — № 2. — С. 20—30.
62. *Растригин Л. А.* В мире случайных событий. — Рига : ИЭВТ, 1963. — 79 с.
63. *Растригин Л. А.* Статистические методы поиска. — М. : Наука, 1968. — 376 с.
64. *Растригин Л. А.* Теория и применение случайного поиска. — Рига : Зинатне, 1969. — 307 с.
65. *Решмин С. А.*, *Черноусько Ф. Л.* Оптимальный по быстродействию синтез управления нелинейным маятником // Известия РАН. Теория и системы управления. — 2007. — № 1. — С. 13—22.
66. *Ройтенберг Я. Н.* Автоматическое управление. — М. : Наука, 1971. — 396 с.

67. *Саттон Р. С.*, *Барто Э. Г.* Обучение с подкреплением. — М. : ДМК-Пресс, 2020. — 552 с.
68. Синергетика и проблемы теории управления. / под ред. А.А. Колесникова. — М. : Физматлит, 2004. — 504 с.
69. Синтез оптимального управления на основе объединенного принципа максимума / А. А. Костоглотов [и др.] // Известия высших учебных заведений. Северо-Кавказский регион. Технические науки. — 2010. — № 2. — С. 31—37.
70. Синтез системы управления беспилотного летательного аппарата по высоте методом бэкстеппинга / С. А. Ахрамович [и др.] // Вестник Самарского университета. Аэрокосмическая техника, технологии и машиностроение. — 2018. — Т. 17, № 2. — С. 7—22.
71. *Скобцов Ю. А.* От генетических алгоритмов к метаэвристикам // Информатика и кибернетика. — 2021. — Т. 23/24, № 1/2. — С. 101—107.
72. *Соболь Б. В.*, *Месхи Б. Ч.*, *Каныгин Г. И.* Методы оптимизации. Практикум. — Ростов н/Д : Феникс, 2009. — 380 с.
73. Современная прикладная теория управления. Ч. I: Оптимизационный подход в теории управления / под ред. А.А. Колесникова. — Таганрог : Изд-во ТРТУ, 2000. — 400 с.
74. Справочник по теории автоматического управления. / под ред. А.А. Красовского. — М. : Наука, 1987. — 712 с.
75. *Сю Д.*, *Мейер А.* Современная теория автоматического управления и ее применение. — М. : Машиностроение, 1972. — 544 с.
76. *Федоренко Р. П.* Приближенное решение задач оптимального управления. — М. : Наука, 1978. — 488 с.
77. *Фуртат И. Б.* Модифицированный алгоритм обратного обхода интегратора // Мехатроника, автоматизация, управление. — 2009. — № 10. — С. 2—7.
78. *Фуртат И. Б.*, *Нехороших А. Н.* Метод бэкстеппинга для структурно неопределенных объектов // Научно-технический вестник информационных технологий, механики и оптики. — 2016. — Т. 16, № 1. — С. 61—67.
79. *Фуртат И. Б.*, *Тупичин Е. А.* Упрощенный алгоритм бэкстеппинга для управления нелинейными системами // Известия высших учебных заведений. Приборостроение. — 2015. — Т. 58, № 3. — С. 173—178.

80. *Халил Х. К.* Нелинейные системы. — М.-Ижевск : Регулярная и хаотическая динамика, 2009. — 832 с.

81. *Чебыкин Д. В.* Backstepping – метод синтеза управления для нелинейных объектов // Труды международной конференции студентов, аспирантов и молодых ученых “Информационные технологии, телекоммуникации и системы управления”. — Екатеринбург : УрФУ, 2015. — С. 248—254.

82. *Чураков Е. П.* Оптимальные а адаптивные системы. — М. : Энергоатомиздат, 1987. — 256 с.

83. *Яковенко П. Г.* Методика последовательного многошагового синтеза оптимальных управлений // Известия Томского политехнического университета. Инжиниринг георесурсов. — 2003. — Т. 306, № 2. — С. 95—98.

84. A hybrid approach to modeling metabolic systems using genetic algorithm and simplex method / J. Yen [et al.] // Proceedings of the 11th IEEE Conference on Artificial Intelligence for Applications. — IEEE, 1995. — P. 277—283.

85. *Atkeson C. G.*, *Moore A. W.*, *Schaal S.* Locally Weighted Learning for Control // Artificial Intelligence Review. — 1997. — Vol. 11. — P. 75—113.

86. Automatic Synthesis of both the Topology and Parameters for a Robust Controller for a Non-Minimal Phase Plant and a Three-Lag Plant by Means of Genetic Programming / J. R. Koza [et al.] // Proceedings of the 38th IEEE Conference on Decision and Control. Vol. 5. — IEEE, 1999. — P. 5292—5300.

87. *Bard Y.* Comparison of gradient methods for the solution of nonlinear parameter estimation problems // SIAM Journal on Numerical Analysis. — 1970. — Vol. 7, no. 1. — P. 157—186.

88. *Bartz-Beielstein T.*, *Preuss M.* Experimental Analysis of Optimization Algorithms: Tuning and Beyond // Theory and Principled Methods for the Design of Metaheuristics / ed. by Y. Borenstein, A. Moraglio. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2014. — P. 205—245.

89. *Beiranvand V.*, *Hare W.*, *Lucet Y.* Best practices for comparing optimization algorithms // Optimization and Engineering. — 2017. — Vol. 18, no. 4. — P. 815—848.

90. Constructing parsimonious analytic models for dynamic systems via symbolic regression / E. Derner [et al.] // Applied Soft Computing. — 2020. — Vol. 94. — P. 106432.

91. Continuous control with deep reinforcement learning [Электронный ресурс] / T. P. Lillicrap [et al.] // 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings / ed. by Y. Bengio, Y. LeCun. — 2016. — URL: https://arxiv.org/abs/1509.02971 (visited on 04/12/2022).

92. *Crispin Y. J.* Evolutionary Computation for Discrete and Continuous Time Optimal Control Problems // Proceedings of the Second International Conference on Informatics in Control, Automation and Robotics. Vol. 4. — INSTICC. SciTePress, 2005. — P. 45—54.

93. *De Luca A.*, *Oriolo G.*, *Samson C.* Feedback control of a nonholonomic car-like robot // Robot Motion Planning and Control / ed. by J.-P. Laumond. — Berlin, Heidelberg : Springer, 1998. — P. 171—253.

94. *Diveev A. I.* Small Variations of Basic Solution Method for Non-numerical Optimization // IFAC-PapersOnLine – 16th IFAC Workshop on Control Applications of Optimization CAO'2015. — 2015. — Vol. 48, no. 25. — P. 28—33.

95. *Diveev A. I.*, *Kazaryan D. E.*, *Sofronova E. A.* Symbolic regression methods for control system synthesis // 22nd Mediterranean Conference on Control and Automation. — 2014. — P. 587—592.

96. *Diveev A. I.*, *Sofronova E. A.* Application of network operator method for synthesis of optimal structure and parameters of automatic control system // IFAC Proceedings Volumes – 17th IFAC World Congress. — 2008. — Vol. 41, no. 2. — P. 6106—6113.

97. *Dorigo M.*, *Gambardella L. M.* Ant colony system: A cooperative learning approach to the traveling salesman problem // IEEE Transactions on Evolutionary Computation. — 1997. — Vol. 1, no. 1. — P. 53—66.

98. *Duriez T.*, *Brunton S.*, *Noack B. R.* Machine Learning Control – Taming Nonlinear Dynamics and Turbulence. — Switzerland : Springer International Publishing, 2017. — 229 p.

99. *Eberhart R.*, *Shi Y.*, *Kennedy J.* Swarm Intelligence. — San Francisco : Morgan Kaufmann, 2001. — 512 p.

100. *Ernst D.*, *Geurts P.*, *Wehenkel L.* Tree-based Batch Mode Reinforcement Learning // Journal of Machine Learning Research. — 2005. — Vol. 6. — P. 503—556.

101. *Feoktistov V.* Differential Evolution. In Search of Solutions. — Boston, MA : Springer, 2006. — 208 p.

102. Genetic Programming IV. Routine Human-Competitive Machine Intelligence / J. R. Koza [et al.]. — Boston, MA : Springer, 2003. — 590 p.

103. *Goldberg D. E.* Genetic Algorithms in Search, Optimization, and Machine Learning. — Boston, MA : Addison-Wesley, 1989. — 432 p.

104. *Goodfellow I.*, *Bengio Y.*, *Courville A.* Deep Learning. — Cambringe, MA : The MIT Press, 2016. — 800 p.

105. *Grosan C.*, *Abraham A.*, *Nicoara M.* Search optimization using hybrid particle sub-swarms and evolutionary algorithms // International Journal of Simulation Systems, Science and Technology. — 2005. — Vol. 6, no. 10. — P. 60—79.

106. *Holland J. N.* Adaptation in Natural and Artificial Systems. — Cambridge, MA : A Bradford Book, 1992. — 232 p.

107. *Ibadulla S. I.*, *Shmalko E. Y.*, *Daurenbekov K. K.* The Comparison of Genetic Programming and Variational Genetic Programming for a Control Synthesis Problem on the Model "Predator-victim" // Procedia Computer Science. — 2017. — Vol. 103. — P. 155—161.

108. *Jamil M.*, *Yang X.-S.* A literature survey of benchmark functions for global optimization problems // International Journal of Mathematical Modelling and Numerical Optimisation. — 2013. — Vol. 4, no. 2. — P. 150—194.

109. *Kaelbling L. P.*, *Littman M. L.*, *Moore A. W.* Reinforcement Learning: A Survey // Journal of Artificial Intelligence Research. — 1996. — Vol. 4. — P. 237—285.

110. *Kalman R. E.* Contributions to the theory of optimal control // Boletin de la Sociedad Matematica Mexicana. — 1960. — Vol. 5. — P. 102—119.

111. *Kanellakopoulos I.*, *Kokotović P. V.*, *Morse A. S.* Systematic design of adaptive controllers for feedback linearizable systems // IEEE Transactions on Automatic Control. — 1991. — Vol. 36, no. 11. — P. 1241—1253.

112. *Kennedy J.*, *Eberhart R.* Particle swarm optimization // Proceedings of ICNN'95 – International Conference on Neural Networks. Vol. 4. — IEEE, 1995. — P. 1942—1948.

113. *Kingma D. P.*, *Ba J.* Adam: A Method for Stochastic Optimization [Электронный ресурс] // 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings / ed. by Y. Bengio, Y. LeCun. — 2015. — URL: http://arxiv.org/abs/1412.6980 (visited on 04/12/2022).

114. *Kokotović P. V.* The joy of feedback: nonlinear and adaptive // IEEE Control Systems Magazine. — 1992. — Vol. 12, no. 3. — P. 7—17.

115. *Koza J.*, *Keane M. A.*, *Rice J. P.* Performance improvement of machine learning via automatic discovery of facilitating functions as applied to a problem of symbolic system identification // IEEE International Conference on Neural Networks. Vol. 1. — IEEE, 1993. — P. 191—198.

116. *Koza J. R.* Hierarchical genetic algorithms operating on populations of computer programs // Proc. 11th Int. Joint Conf. on Artificial Intelligence, Vol. 1. — San Mateo, CA : Morgan Kaufmann, 1989. — P. 768—774.

117. *Koza J. R.* Genetic Programming: On the Programming of Computers by Means of Natural Selection. — Cambridge, MA : The MIT Press, 1992. — 840 p.

118. *Koza J. R.*, *Keane M. A.*, *Streeter M. J.* Evolving inventions // Scientific American. — 2003. — Vol. 288, no. 2. — P. 52—59.

119. *Krstić M.*, *Kanellakopoulos M.*, *Kokotović P. V.* Adaptive nonlinear control without overparametrization // Systems & Control Letters. — 1992. — Vol. 19, no. 3. — P. 177—185.

120. *Langdon W. B.*, *Poli R.* Foundations of Genetic Programming. — Berlin : Springer, 2002. — 260 p.

121. *Lopez Cruz I. L.*, *Van Willigenburg L. G.*, *Van Straten G.* Efficient Differential Evolution algorithms for multimodal optimal control problems // Applied Soft Computing. — 2003. — Vol. 3, no. 2. — P. 97—122.

122. *Luo C.*, *Zhang S.-L.* Parse-matrix evolution for symbolic regression // Engineering Applications of Artificial Intelligence. — 2012. — Vol. 25, no. 6. — P. 1182—1193.

123. *Miller J. F.* Cartesian Genetic Programming. — Berlin, Heidelberg : Springer, 2011. — 346 p.

124. *Miller J. F.* An Empirical Study of the Efficiency of Learning Boolean Functions Using a Cartesian Genetic Programming Approach // GECCO'99: Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation. Vol. 2. — San Francisco, CA : Morgan Kaufmann, 1999. — P. 1135—1142.

125. *Miller J. F.*, *Thomson P.* Cartesian Genetic Programming // Genetic Programming – EuroGP 2000, LNCS. Vol. 1802 / ed. by R. Poli [et al.]. — Berlin, Heidelberg : Springer, 2000. — P. 121—132.

126. *Mirjalili S.*, *Mirjalili S. M.*, *Lewis A.* Grey Wolf Optimizer // Advances in Engineering Software. — 2014. — Vol. 69. — P. 46—61.

127. *More J. J.*, *Wild S.* Benchmarking derivative-free optimization algorithms // SIAM Journal on Optimization. — 2009. — Vol. 20, no. 1. — P. 172—191.

128. *Munos R.*, *Moore A.* Variable Resolution Discretization in Optimal Control // Machine Learning. — 2002. — Vol. 49, no. 2. — P. 291—323.

129. *Nikolaev N. I.*, *Iba H.* Inductive genetic programming of polynomial learning networks // Proceedings of the First IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks. — 2000. — P. 158—167.

130. *O'Neill M.*, *Ryan C.* Grammatical evolution // IEEE Transactions on Evolutionary Computation. — 2001. — Vol. 5, no. 4. — P. 349—358.

131. *O'Neill M.*, *Ryan C.* Grammatical Evolution. Evolutionary Automatic Programming in an Arbitrary Language. — Springer, 2002. — 160 p.

132. *Oyama K.*, *Nonaka K.* Model predictive parking control for nonholonomic vehicles using time-state control form // 2013 European Control Conference (ECC). — 2013. — P. 458—465.

133. *Quinlan J. R.* Induction of decision trees // Machine Learning. — 1986. — Vol. 1. — P. 81—106.

134. *Raidl G. R.* A Unified View on Hybrid Metaheuristics // Hybrid Metaheuristics / ed. by F. Almeida [et al.]. — Berlin, Heidelberg : Springer, 2006. — P. 1—12.

135. *Rardin R. L.*, *Uzsoy R.* Experimental evaluation of heuristic optimization algorithms: A tutorial // Journal of Heuristics. — 2001. — Vol. 7, no. 3. — P. 261—304.

136. *Ryan C.*, *Collins J.*, *O'Neill M.* Grammatical evolution: Evolving programs for an arbitrary language // Genetic Programming – EuroGP 1998. Lecture Notes in Computer Science. Vol. 1391 / ed. by W. Banzhaf [et al.]. — Berlin, Heidelberg : Springer, 1998. — P. 83—96.

137. *Schoen F.* A wide class of test functions for global optimization // Journal of Global Optimization. — 1993. — Vol. 3, no. 2. — P. 133—137.

138. *Sinha A.*, *Chen Y.-P.*, *Goldberg D. E.* Designing Efficient Genetic and Evolutionary Algorithm Hybrids // Recent Advances in Memetic Algorithms / ed. by W. E. Hart, J. E. Smith, N. Krasnogor. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2005. — P. 259—288.

139. *Storn R.*, *Price K.* Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces // Journal of Global Optimization. — 1997. — No. 11. — P. 341—359.

140. Symbolic regression methods for reinforcement learning [Электронный ресурс] / J. Kubalík [et al.] // arXiv:1903.09688 [cs.LG]. — 2019. — P. 1—12. — URL: https : / / arxiv . org / abs / 1903 . 09688 (visited on 04/12/2022).

141. *Tabak D.* Comparative study of various minimization techniques used in mathematical programming // IEEE Transactions on Automatic Control. — 1969. — Vol. 14, no. 5. — P. 572—572.

142. The Bees Algorithm — A Novel Tool for Complex Optimisation Problems / D. T. Pham [et al.] // Intelligent Production Machines and Systems - 2nd I*PROMS Virtual International Conference 3-14 July 2006 / ed. by D. Pham, E. Eldukhri, A. Soroka. — Oxford : Elsevier Science Ltd, 2006. — P. 454—459.

143. Variational Analytic Programming for Synthesis of Optimal Control for Flying Robot / A. I. Diveev [et al.] // IFAC-PapersOnLine – 11th IFAC Symposium on Robot Control SYROCO 2015. — 2015. — Vol. 48, no. 19. — P. 75—80.

144. Variational Genetic Programming for Optimal Control System Synthesis of Mobile Robots / A. I. Diveev [et al.] // IFAC-PapersOnLine – 11th IFAC Symposium on Robot Control SYROCO 2015. — 2015. — Vol. 48, no. 19. — P. 106—111.

145. *Walker J. A.*, *Miller J. F.* Evolution and Acquisition of Modules in Cartesian Genetic Programming // Genetic Programming – Proceedings of 7th European Conference, EuroGP 2004, Coimbra, Portugal, April 5-7, 2004. Vol. 3003 / ed. by M. Keijzer [et al.]. — Berlin, Heidelberg : Springer-Verlag, 2004. — P. 187—197.

146. *Wang X.* Hybrid nature-inspired computation methods for optimization [Электронный ресурс] // TKK dissertations, 161. — 2009. — URL: http://lib.tkk.fi/Diss/2009/isbn9789512298594/isbn9789512298594.pdf (visited on 04/12/2022).

147. *Yang X.-S.* A New Metaheuristic Bat-Inspired Algorithm // Studies in Computational Intelligence – Proceedings of Nature Inspired Cooperative Strategies for Optimization (NISCO 2010). Vol. 284 / ed. by C. Cruz [et al.]. — Berlin, Heidelberg : Springer-Verlag, 2010. — P. 65—74.

148. *Yang X.-S.* Engineering Optimization: An Introduction with Metaheuristic Applications. — John Wiley & Sons, 2010. — 376 p.

149. *Yang X.-S.* Firefly Algorithm. Stochastic Test Functions and Design optimization // International Journal of Bio-Inspired Computation. — 2010. — Vol. 2, no. 2. — P. 78—84.

150. *Yang X.-S.* Swarm intelligence based algorithms: a critical analysis // Evolutionary Intelligence. — 2014. — Vol. 7, no. 1. — P. 17—28.

151. *Yang X.-S.* Nature-Inspired Optimization Algorithms, 2nd Edition. — Academic Press, 2020. — 310 p.

152. *Yang X.-S.*, *Deb S.* Cuckoo Search via Lévy flights // 2009 World Congress on Nature Biologically Inspired Computing (NaBIC). — IEEE, 2009. — P. 210—214.

153. *Yegorov I.*, *Bratus A. S.*, *Todorov Y.* Synthesis of optimal control in a mathematical model of economic growth under R&D investments // Applied Mathematical Sciences. — 2015. — Vol. 9, no. 91. — P. 4523—4564.

154. *Zelinka I.*, *Oplatkova Z.*, *Nolle L.* Analytic Programming – Symbolic Regression by Means of Arbitrary Evolutionary Algorithms // Special Issue on Inteligent Systems of International Journal of Simulation, Systems, Science and Technology. — 2005. — Vol. 6, no. 9. — P. 44—55.

155. *Zhang S.*, *Qian W.* Dynamic backstepping control for pure-feedback nonlinear systems [Электронный ресурс] // arXiv:1706.08641 [cs.SY]. — 2017. — P. 1—20. — URL: https://arxiv.org/abs/1706.08641 (visited on 04/12/2022).

Публикации автора по теме диссертации

156. *Дивеев А. И.*, *Константинов С. В.* Сравнительный экспериментальный анализ эволюционных алгоритмов оптимизации // Труды 11-го международного симпозиума «Интеллектуальные системы» (INTELS'2014, Москва, 30 июня – 4 июля 2014 г.) — М. : РУДН, 2014. — С. 139—144.

157. *Дивеев А. И.*, *Константинов С. В.* Исследование эволюционных алгоритмов для решения задачи оптимального управления // Труды МФТИ. — 2017. — Т. 9, № 3. — С. 76—85.

158. *Дивеев А. И.*, *Константинов С. В.* Эволюционные алгоритмы для решения задачи оптимального управления // Вестник РУДН. Серия: Инженерные исследования. — 2017. — Т. 18, № 2. — С. 254—265.

159. *Дивеев А. И.*, *Константинов С. В.* Задача оптимального управления и ее решение эволюционным алгоритмом «серого волка» // Вестник РУДН. Серия: Инженерные исследования. — 2018. — Т. 19, № 1. — С. 67—79.

160. *Дивеев А. И.*, *Константинов С. В.* Исследование практической сходимости эволюционных алгоритмов оптимального программного управления колесным роботом // Известия РАН. Теория и системы управления. — 2018. — № 4. — С. 75—98.

161. *Дивеев А. И.*, *Константинов С. В.* Экспериментальное сравнение алгоритмов случайного поиска и эволюционных вычислений в задаче оптимального управления группой роботов // Фундаментально-прикладные проблемы безопасности, живучести, надёжности, устойчивости и эффективности систем. Материалы III международной научно-практической конференции, посвященной 110-летию со дня рождения академика Н.А. Пилюгина, Елец. 3-5 июня 2019 г. — Елец, 2019. — С. 249—253.

162. *Константинов С. В.*, *Мишинева М. А.* Обзор современных популяционных методов глобальной оптимизации // Труды VIII международной научно-практической конференции "Инженерные системы — 2015" (Москва, 20-22 апреля 2015г.) — М. : РУДН, 2015. — С. 197—203.

163. Поиск структуры и параметров закона взаимодействия веществ в химической реакции методом сетевого оператора / И. М. Губайдуллин [и др.] // Наукоемкие технологии. — М., 2016. — Т. 17, № 6. — С. 76—82.

164. Разработка кинетических моделей сложных химических реакций методом сетевого оператора [Электронный ресурс] / И. М. Губайдуллин [и др.] // Современные проблемы науки и образования. — 2014. — № 6. — С. 1—11. — URL: https://science-education.ru/ru/article/view?id=16113 (дата обр. 12.04.2022).

165. Comparative Research of Random Search Algorithms and Evolutionary Algorithms for the Optimal Control Problem of the Mobile Robot / S. V. Konstantinov [et al.] // Procedia Computer Science. — 2019. — Vol. 150. — P. 462—470.

166. *Diveev A. I.*, *Balandina G. I.*, *Konstantinov S. V.* Binary variational genetic programming for the problem of synthesis of control system // 2017 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD). — 2017. — P. 186—191.

167. *Diveev A. I.*, *Konstantinov S. V.* Study of the Practical Convergence of Evolutionary Algorithms for the Optimal Program Control of a Wheeled Robot // Journal of Computer and Systems Sciences International. — 2018. — Vol. 57, no. 4. — P. 561—580.

168. *Diveev A. I.*, *Konstantinov S. V.*, *Danilova A. M.* Solution of the optimal control problem by symbolic regression method // Procedia Computer Science. — 2021. — Vol. 186. — P. 646—653.

169. *Diveev A. I.*, *Konstantinov S. V.*, *Sofronova E. A.* A comparison of evolutionary algorithms and gradient-based methods for the optimal control problem // 2018 5th International Conference on Control, Decision and Information Technologies (CoDIT). — 2018. — P. 259—264.

170. *Diveev A.*, *Konstantinov S.* Applying Neural Networks for the Identification of Control Object Mathematical Models for the Control Problems // 2022 8th International Conference on Control, Decision and Information Technologies (CoDIT). — IEEE, 2022. — P. 1059—1063.

171. *Diveev A.*, *Sofronova E.*, *Konstantinov S.* Approaches to Numerical Solution of Optimal Control Problem Using Evolutionary Computations // Applied Sciences. — 2021. — Vol. 11, no. 15. — P. 7096.

172. *Diveev A. I.*, *Konstantinov S. V.* Solution of the problem of the control system general synthesis by approximation of a set of extremals // Advances in Optimization and Applications. OPTIMA 2020. Communications in Computer and Information Science. Vol. 1340 / ed. by N. Olenev [et al.]. — Springer, 2021. — P. 113—128.

173. *Konstantinov S. V.*, *Baryshnikov A. A.* Comparative analysis of evolutionary algorithms for the problem of parametric optimization of PID controllers // Procedia Computer Science. — 2017. — Vol. 103. — P. 100—107.

174. *Konstantinov S. V.*, *Diveev A. I.* Control system synthesis based on optimal trajectories approximation by symbolic regression for group of robots // 2020 7th International Conference on Control, Decision and Information Technologies (CoDIT). — IEEE, 2020. — P. 19—24.

175. *Konstantinov S. V.*, *Diveev A. I.* Solving the Problem of the Optimal Control System General Synthesis Based on Approximation of a Set of Extremals using the Symbol Regression Method // Herald of the Bauman Moscow State Technical University. Series Instrument Engineering. — 2020. — Vol. 2, no. 131. — P. 59—74.

176. *Konstantinov S. V.*, *Diveev A. I.* A new two-step approach for solving a control system synthesis problem by symbolic regression methods // Procedia Computer Science. — 2021. — Vol. 186. — P. 636—645.

177. *Konstantinov S. V.*, *Khamidova U. K.*, *Sofronova E. A.* A Novel Hybrid Method of Global Optimization Based on the Grey Wolf Optimizer and the Bees Algorithm // Procedia Computer Science. — 2019. — Vol. 150. — P. 471—477.

178. *Konstantinov S. V.*, *Diveev A. I.* Evolutionary Algorithms for Optimal Control Problem of Mobile Robots Group Interaction // Advances in Optimization and Applications. OPTIMA 2021. Communications in Computer and Information Science. Vol. 1514 / ed. by N. N. Olenev [et al.]. — Springer, 2021. — P. 123—136.

179. Machine Learning Control Based on Approximation of Optimal Trajectories / A. Diveev [et al.] // Mathematics. — 2021. — Vol. 9, no. 3. — P. 265.

180. Optimal control system synthesis based on the approximation of extremals by symbolic regression / S. V. Konstantinov [et al.] // 2020 European Control Conference (ECC). — IEEE, 2020. — P. 2021—2026.

Приложение А. Акты о внедрении

Акт об использовании результатов диссертационного исследования в научно-исследовательской и опытно-конструкторской работе инжинирингового центра «Интеллектуальные роботизированные системы и технологии» Белгородского государственного технологического университета им. В.Г. Шухова (Белгород, Россия)

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г. ШУХОВА»
(БГТУ им. В.Г. Шухова)

Костюкова ул., д.46, Белгород, 308012, тел.(4722)54-20-87, факс (4722)55-71-39.
E-mail: rector@intbel.ru, http://www.bstu.ru

«26» 07 2022 № 1503
На № _____ от _____

АКТ
о применении результатов диссертационного исследования Константинова Сергея Валерьевича в научно-исследовательской работе Инжинирингового центра «Интеллектуальные роботизированные системы и технологии» УНИР БГТУ им. В.Г. Шухова

Настоящим актом подтверждается, что результаты диссертационной работы Константинова Сергея Валерьевича «Решение задачи синтеза системы управления на основе аппроксимации множества оптимальных траекторий методом сетевого оператора», представленной на соискание ученой степени кандидата технических наук, применяются в научно-исследовательской и опытно-конструкторской работе Инжинирингового центра «Интеллектуальные роботизированные системы и технологии» управления научно-исследовательских работ Федерального государственного бюджетного образовательного учреждения высшего образования «Белгородский государственный технологический университет им. В.Г. Шухова» при проектировании и создании робототехнических и управляющих систем.

Разработанные автором в диссертационном исследовании алгоритмы синтеза оптимальных систем управления, в том числе на основе эволюционных методов, а также программное обеспечение для их реализации использованы в научно-исследовательской и опытно-конструкторской работе Центра, что позволило повысить эффективность применяемых подходов при проектировании робототехнических и управляющих систем.

Руководитель ИЦ «Интеллектуальные роботизированные системы и технологии», д.т.н. проф. **Л.А. Рыбак**

Первый проректор, д.т.н., профессор **Е.И. Евтушенко**

Акт о внедрении результатов диссертационного исследования в деятельность ООО «Экспериментальная мастерская НаукаСофт» (Москва, Россия)

Утверждаю
Генеральный директор
ООО «Экспериментальная мастерская
НаукаСофт»,

С. П. Халютин
«01» _____2022 г.

Акт
о реализации результатов диссертации
Константинова Сергея Валерьевича
на тему «Решение задачи синтеза системы управления на основе аппроксимации множества оптимальных траекторий методом сетевого оператора»
в ООО «Экспериментальная мастерская НаукаСофт»

Комиссия в составе: председателя – главного конструктора – заместителя генерального директора, кандидата технических наук, доцента Жмурова Бориса Владимировича, членов комиссии – начальника научно-исследовательского отдела, доктора технических наук, доцента Давидова Альберта Оганезовича и начальника лаборатории технической документации, кандидата технических наук Морозова Михаила Игоревича, составила настоящий акт о том, что технические решения представленные в диссертации Константинова С.В., а именно:

1. Численный метод синтеза системы управления на основе использования данных об оптимальных траекториях;
2. Алгоритм поиска оптимального математического выражения, аппроксимирующего данные методами символьной регрессии, и представлена его программная реализация;
3. Экспериментальные данные об эффективности применения эволюционных алгоритмов в задачах оптимального управления;
4. Гибридный метод оптимизации для задач оптимального управления и представлена его программная реализация.

использованы в деятельности ООО «Экспериментальная мастерская НаукаСофт» при проектировании и разработке программно-аппаратных комплексов в рамках выполнения СЧ НИР «Волна-НС», что позволило расширить сферу применения проектируемых автоматизированных систем, а также повысить их эффективность.

Председатель комиссии: Б.В. Жмуров

Члены комиссии: А.О. Давидов

М.И. Морозов