Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра оптимального управления

Отчет по практикуму

Программа для решения краевых задач методом продолжения по параметру

Студент
Группы 313
Царьков Денис Владимирович

Преподаватели
Киселёв Юрий Николаевич
Аввакумов Сергей Николаевич
Дряженков Андрей Александрович

Москва, 2025

Содержание

1 О проекте

1.1 Цель проекта

Разработка программного обеспечения на Python для численного решения краевых задач систем обыкновенных дифференциальных уравнений методом параметрического продолжения. Программный комплекс включает:

- Графический интерфейс для задания уравнений и краевых условий
- Инструменты настройки параметров численного метода
- Визуализацию решений в графическом и табличном форматах
- Функционал вычисления интегралов от компонент решения

1.2 Технологический стек

Программа реализована с использованием следующих технологий:

- Язык программирования: Python 3
- Библиотеки:
  - `PyQt5` - графический интерфейс
  - `SciPy` - численные методы (`solve_ivp`, `root`)
  - `NumPy` - вычисления с массивами
  - `Matplotlib` - визуализация результатов

1.3 Архитектура программы

Программа имеет модульную структуру с многооконным интерфейсом. Основные компоненты:

- **Меню управления**:
  - Работа с файлами (сохранение/загрузка)
  - Доступ к библиотеке примеров
  - Справка и информация о программе
- **Блок ввода задачи**:
  - Поля для метаданных (название, комментарии)
  - Настраиваемые таблицы для системы ОДУ
  - Таблица краевых условий с автоматической адаптацией размерности
- **Параметры решения**:
  - Интервал интегрирования [a, b]
  - Контрольные точки (t*)

  - Точности вычислений
  - Параметры метода продолжения
  - Начальные приближения

- **Инструменты визуализации**:
  - Построение графиков решений
  - Табличное представление результатов
  - Вычисление интегралов
  - Настройка осей и экспорт данных

1.4 Функциональные возможности

Программа поддерживает:

- Решение нелинейных краевых задач для систем ОДУ
- Метод продолжения по параметру с адаптивным выбором шага
- Различные численные методы (Рунге-Кутта, BDF, Radau)
- Обработку сложных краевых условий (включая нелинейные)
- Интерактивное исследование решений

1.5 Области применения

Разработанный программный комплекс может быть использован для:

- Исследования динамических систем в механике и физике
- Анализа устойчивости решений
- Решения задач оптимального управления
- Образовательных целей при изучении численных методов

2 Метод продолжения по параметру

Для решения краевой задачи

$$\dot{x} = f(t, x), \quad t \in [a, b], \quad x \in \mathbb{R}^n$$
$$R(x(a), x(b)) = 0$$

где $f : [a, b] \times \mathbb{R}^n \to \mathbb{R}^n$ и $R : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ – гладкие векторные функции, используется метод продолжения по параметру. Основная идея метода заключается в сведении исходной краевой задачи к задаче Коши для специальной системы дифференциальных уравнений относительно параметра продолжения или к последовательности нелинейных алгебраических уравнений.

В данной реализации используется предиктор-корректор схема метода продолжения. Исходная краевая задача (**??**) сводится к решению нелинейного векторного уравнения относительно вектора начальных условий $p = x(a) \in \mathbb{R}^n$:

$$\Phi(p) \equiv R(x(a,p), x(b,p)) = 0$$

где $x(t,p)$ – решение задачи Коши $\dot{x} = f(t,x)$ с начальным условием $x(a) = p$.

Вводится гомотопия:

$$\Psi(p,\mu) = \Phi(p) - (1-\mu)\Phi(p_0) = 0$$

где $\mu \in [0,1]$ – параметр продолжения, а p_0 – начальное приближение для $x(a)$. При $\mu = 0$ уравнение $\Psi(p,0) = 0$ сводится к $\Phi(p) = \Phi(p_0)$, что удовлетворяется при $p = p_0$. При $\mu = 1$ уравнение $\Psi(p,1) = 0$ сводится к $\Phi(p) = 0$, что является исходной задачей.

Метод продолжения строит последовательность решений $p(\mu_k)$ для дискретных значений μ_k, где $0 = \mu_0 < \mu_1 < \cdots < \mu_N = 1$. На каждом шаге от μ_k к μ_{k+1}:

1. **Предиктор:** Используется информация о решении при μ_k для получения начального приближения $p_{k+1}^{(0)}$ для решения при μ_{k+1}. В данной реализации используется касательный предиктор:

   $$p_{k+1}^{(0)} = p_k + \frac{dp}{d\mu}\Big|_{\mu_k} (\mu_{k+1} - \mu_k)$$

   где $\frac{dp}{d\mu} = -[\Phi'(p)]^{-1}\Phi(p_0)$. Для вычисления $\Phi'(p)$ решается расширенная система ОДУ для $x(t,p)$ и матрицы чувствительности $X(t,p) = \frac{\partial x(t,p)}{\partial p}$:

   $$\dot{x} = f(t,x), \quad x(a) = p$$
   $$\dot{X} = A(t,x)X, \quad X(a) = I$$

   где $A(t,x) = \frac{\partial f}{\partial x}(t,x)$. Тогда $\Phi'(p) = R'_x(x(a,p),x(b,p))X(a,p) + R'_y(x(a,p),x(b,p))X(b,p)$.

2. **Корректор:** Решается нелинейное уравнение $\Psi(p,\mu_{k+1}) = 0$ для нахождения скорректированного решения p_{k+1} при μ_{k+1}, начиная с предсказания $p_{k+1}^{(0)}$. Используется итерационный метод (например, метод Ньютона или его модификации), реализованный в `scipy.optimize.root`.

Процесс продолжается до достижения $\mu = 1$.

3 Примеры

В программе реализована библиотека примеров краевых задач, демонстрирующих возможности метода:

- **Пример 1: Краевая задача двух тел.** Задача гравитационного движения

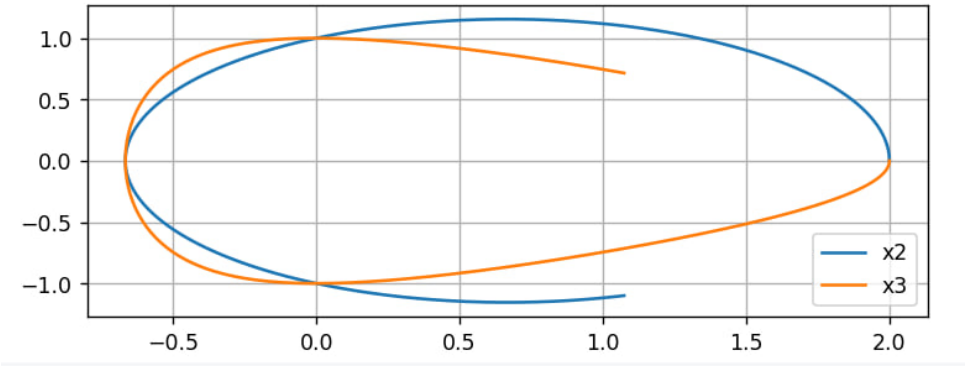

Рис. 1: Траектория гравитационного взаимодействия

двух тел, сводящаяся к системе из 4 ОДУ с краевыми условиями на координаты в начальный и конечный моменты времени. Параметры: $x_1(0) = 2$, $x_2(0) = 0$, $x_1(7) \approx 1.07$, $x_2(7) \approx -1.10$.

- **Пример 2: Предельные циклы в системе Эквейлера.** Задача поиска пе-

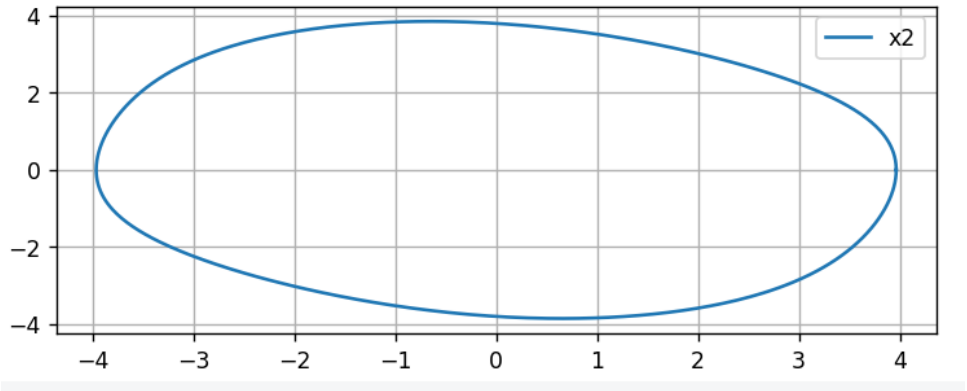

Рис. 2: Фазовый портрет предельного цикла

риодических решений для системы Эквейлера $\dot{x}_1 = x_3 x_2$, $\dot{x}_2 = x_3(-x_1 + \sin x_2)$. Формулируется как краевая задача на интервале $t \in [0, 1]$ с условиями периодичности.

- **Пример 3: Трехкратный интегратор.** Пример с малым параметром ν в

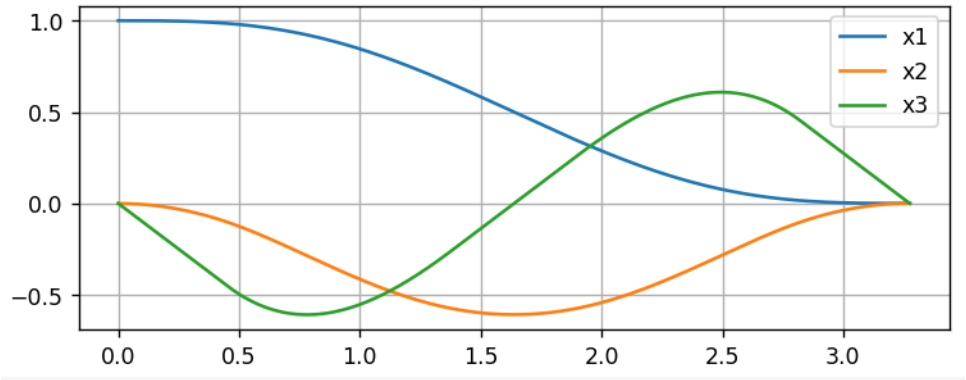

Рис. 3: Фазовые траектории интегратора

нелинейном управлении: $\dot{x}_3 = 0.5(\sqrt{\nu + (u+1)^2} - \sqrt{\nu + (u-1)^2})$. Краевые условия задают переход из начального состояния в нулевое за время $T = 3.275$.

- **Пример 4: Быстродействие с лункой.** Задача оптимального управления с

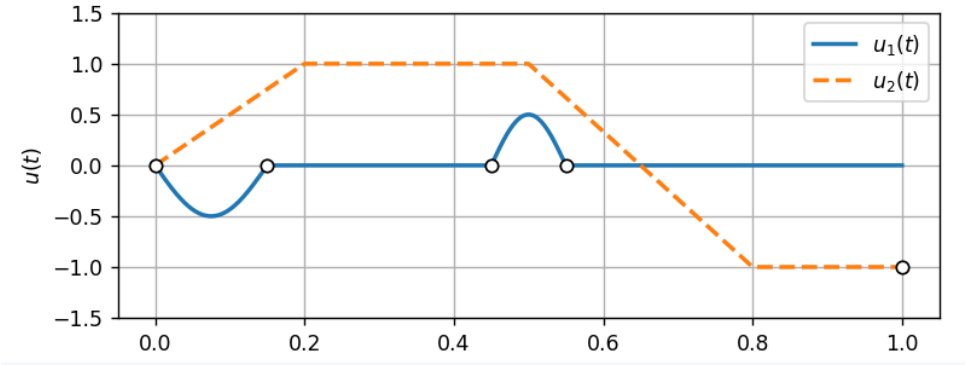

Рис. 4: Оптимальные управления $u_1(t)$ и $u_2(t)$

гладкими управлениями в форме "лунок". Управления имеют вид:

$$u_1(t) = \begin{cases} -0.5\sin(\pi t/0.15) & t \in (0, 0.15) \\ 0.5\sin(\pi(t-0.45)/0.1) & t \in (0.45, 0.55) \\ 0 & \text{иначе} \end{cases}$$

$$u_2(t) = \begin{cases} t/0.2 & t \in [0, 0.2) \\ 1 - 2(t-0.5)/0.3 & t \in [0.5, 0.8) \\ -1 & t \geq 0.8 \end{cases}$$

4 Описание интерфейса

Программа имеет интуитивно понятный графический интерфейс, реализованный с помощью библиотеки PyQt5. Основные элементы управления:

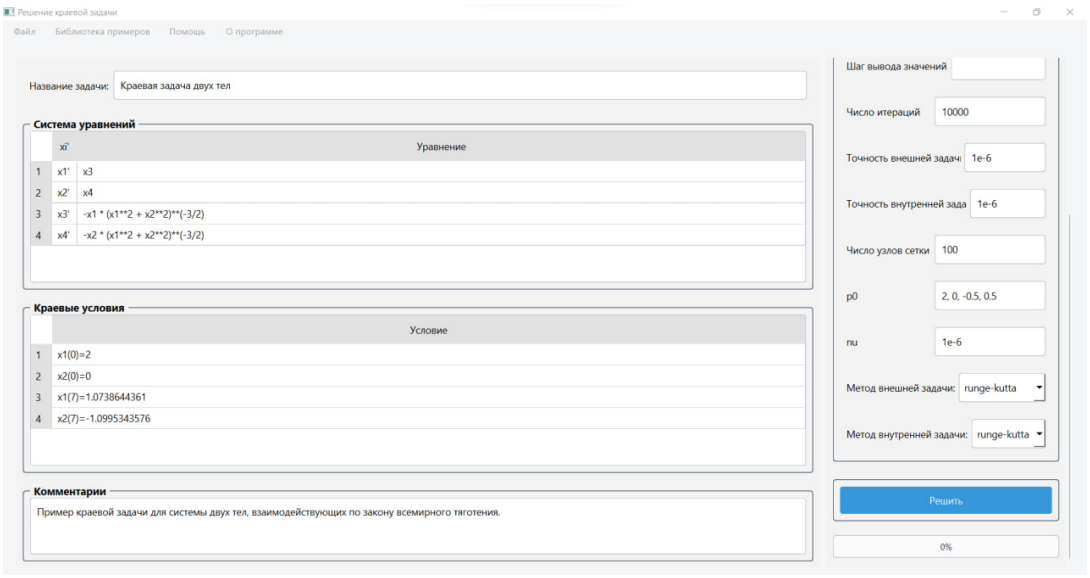

Рис. 5: Общий вид интерфейса программы

4.1 Основные компоненты интерфейса

- **Меню управления** (верхняя панель):
  - Файловые операции (создание/сохранение задач)
  - Доступ к библиотеке примеров
  - Справка и настройки

4.2 Окно результатов

После вычисления открывается отдельное окно с результатами:

Рис. 9: Окно визуализации результатов

Функционал окна результатов включает:

- Интерактивные графики решений
- Настройка отображаемых компонент
- Табличное представление данных
- Инструменты приближения/отдаления
- Вычисление интегралов от решений

5 Описание математических вычислений

Математические вычисления в программе основаны на алгоритме метода продолжения по параметру, описанном в учебном пособии [5].

Внутренняя задача: Решение систем обыкновенных дифференциальных уравнений (как исходной системы для получения траектории, так и расширенной системы для вычисления матрицы чувствительности) выполняется с использованием функции `scipy.integrate.solve_ivp`. Поддерживаются различные методы численного интегрирования ОДУ, доступные в SciPy (например, RK45, Radau, BDF).

Вычисление Якобианов: Якобиан правой части системы ОДУ $A(t, x) = \frac{\partial f}{\partial x}(t, x)$ в текущей реализации вычисляется с использованием конечных разностей. Якобианы краевых условий R'_x и R'_y вычисляются на основе парсинга строковых представлений краевых условий, предполагая их линейную структуру относительно $x(a)$ и $x(b)$.

Внешняя задача (Корректор): На каждом шаге продолжения решается нелинейное уравнение $\Psi(p, \mu_{k+1}) = 0$ относительно вектора p. Для этого используется функция `scipy.optimize.root`, которая предоставляет различные методы поиска корней нелинейных систем (например, 'hybr', 'krylov', 'broyden').

Численное интегрирование: Для вычисления определенного интеграла от функции, зависящей от t и $x(t)$, используется функция `scipy.integrate.quad`. Эта функция выполняет численное интегрирование с адаптивным алгоритмом. Подынтегральная функция динамически создается на основе введенной пользователем строки, используя функцию решения $x(t)$, полученную после успешного нахождения p.

6 Литература

Список литературы

[1] Мнацаканян Ш.А. Отчет по практикуму. Программа для решения краевых задач методом продолжения по параметру. Москва, 2017.

[2] Киселёв Ю.Н., Аввакумов С.Н., Орлов М.В. Оптимальное управление. Линейная теория и приложения. Издательский отдел факультета ВМК МГУ имени М.В. Ломоносова, 2007.

[3] Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., ... & van Mulbregt, P. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3), 261-272.

[4] Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., ... & Oliphant, T.E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357-362.

[5] Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90-95.

[6] Riverbank Computing Ltd. PyQt5. `https://www.riverbankcomputing.com/software/pyqt/`