Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Теоретическая информатика и компьютерные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
НА ТЕМУ:
Моделирование нестационарных линейных систем на основе метода Галёркина

Студент _____ (Группа)	_____ (Подпись, дата)	Хрипач Г. Е. (И.О. Фамилия)
Руководитель ВКР	_____ (Подпись, дата)	Домрачева А. Б. (И.О. Фамилия)
Консультант	_____ (Подпись, дата)	_____ (И.О. Фамилия)
Консультант	_____ (Подпись, дата)	_____ (И.О. Фамилия)
Нормоконтролер	_____ (Подпись, дата)	Алексеев С. М. (И.О. Фамилия)

2022 г.

АННОТАЦИЯ

В данной дипломной работе рассматривается использование метода Галёркина для исследования нестационарных линейных систем, заданных обыкновенными дифференциальными уравнениями с переменными коэффициентами.

Формулируются и описываются на языке `Python` алгоритмы решения краевых задач для дифференциальных уравнений первого и второго порядка методом Галёркина.

Описывается интерфейсное приложение для исследования сложных систем, заданных дифференциальным уравнением второго порядка с переменными коэффициентами, а также раскрываются некоторые детали его реализации.

Записка состоит из 55 страниц, содержит 4 разделов, 3 листингов, 1 таблиц и 16 рисунков.

Настоящая записка выполнена полностью в системе верски докуметов LaTeXс использованием болванки для выпускных квалификационных работ [1], удовлетворяющих требованиями нормоконтроля [2].

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Нестационарными линейными системами или *линейными системами с переменными параметрами* называют системы, которые описываются линейными дифференциальными уравнениями с переменными коэффициентами. Для их описания помимо дифференциальных уравнений могут быть использованы предаточные функции, переходные и весовые (импульсные переходные) функции, частотные функции и их характеристики.

В естественных науках уже давно выделилось весьма перспективное направление, связанное с применением вычислительных методов. Важность этого направления и его самостоятельность подтверждаются, в частности, появлением в научной литературе таких словосочетаний, как «вычислительная физика», «вычислительная гидродинамика» и т.п. Подобные названия давно уже никого не шокируют, более того, они характеризуют подход к исследованиям, существенно отличный от классического и тесно связанный с приближенным аналогом.

Целый ряд вычислительных методов, предназначенных для решения самых разнообразных задач математической физики и техники, базируется на идеях советских ученых И. Г. Бубнова и Б. Г. Галёркина. Несмотря на то что эти идеи были выдвинуты еще в «домашинную» эпоху, когда последним словом вычислительной техники был арифмометр, они оказались чрезвычайно плодотворными и после перехода к массовому использованию ЭВМ в научно-инженерной практике. Идеи Бубнова — Галёркина помогли развитию вариационных методов, методов взвешенных невязок, наконец, способствовали разработке метода конечных элементов и таких его многочисленных модификаций как, например, спектральный метод или метод граничных элементов.

Метод Галёркина является представителем класса методов взвешенных невязок. В этой дипломной работе рассматривается использование традиционного метода Галёркина для исследования нестационарных линейных систем. В часности, акцент будет сделан на формулировке общего алгоритма для таких систем, заданных обыкновенными дифференциальными уравнениями первого и второго порядка с переменными коэффициентами. Также подробно описывается функционал интерфейсного приложения, позволяющего легко задавать различные дифференциальные уравнения второго порядка, получать результаты (на основе

метода Галёркина) в виде графика и оценки погрешности, а также снабженного возможностью сохранять и загружать полученные результаты.

Структура записки имеет следующий вид. Вначале (раздел 1) сообщаются некоторые математические сведения, относящие к сути предметной области. Далее вводятся основные понятия теории автоматического управления, в частности стационарных и нестационарных линейных систем (раздел 2). Затем описывается общая идея метода Галёркина и выводятся алгоритмы для решения краевых задач для дифференциальных уравнений первого и второго порядка на основе данного метода. Наконец, раздел 4 представляет из себя инструкцию по установке и описание десктопного приложения, написанного на языке `Python`, реализующего графический интерфейс для исследования нестационарных линейных систем, а также раскрывает некоторые детали реализации алгоритмов метода Галёркина, связанные с работой с функциями на языке `Python`.

1 Математические сведения

1.1 Ортогональные системы в гильбертовом пространстве

За V будем обозначать векторное пространство над полем действительных чисел $\mathbb{R}$.

Определение 1. *Функция*

$$(\bullet,\bullet) : V \times V \to \mathbb{R}, \quad (v,u) \mapsto (v,u)$$

обладающая следующими свойствами

1. *Билинейность:* $(\alpha_1 v_1 + \beta_1 u_1, \alpha_2 v_2 + \beta_2 u_2) = \alpha_1\alpha_2(v_1,v_2) + \alpha_1\beta_2(v_1,u_2) + \beta_1\alpha_1(v_2,u_1) + \beta_1\beta_2(u_1,u_2) \ \ \forall \alpha_1,\alpha_2,\beta_1,\beta_2 \in \mathbb{R}, \forall v_1,v_2,u_1,u_2 \in V$
2. *Симметричность:* $(v,u) = (u,v) \ \ \forall v,u \in V$
3. *Положительность:* $(v,v) > 0 \ \ \forall v \in V, v \neq 0$,

называется ***евклидовым скалярным произведением***.

Определение 2. *Векторное пространство над полем действительных чисел, на котором задано евклидово скалярное произведение, называется* ***евклидовым***. *[3] [4]*

Определение 3. *Функция* $\| \bullet \| : V \to \mathbb{R}$, $v \mapsto \|v\|$, *на векторном пространстве* V *называется* ***нормой***, *если для всех* $\lambda \in \mathbb{R}$ *и* $v,u \in V$ *выполнены свойства*

1. *Положительность:* $\|v\| \geqslant 0$
2. *Невырожденность:* $\|v\| = 0 \iff v = 0$
3. *Однородность:* $\|\lambda \cdot v\| = |\lambda| \cdot \|v\|$
4. *Неравенство треугольника:* $\|v + u\| \leqslant \|v\| + \|u\|$.

Векторное пространство, на котором задана норма, называется **нормированным**.

В евклидовом пространстве действует **евклидова норма** $\|v\| = \sqrt{(v,v)}$. С помощью нормы можно измерять степень близости векторов и на этой основе дать определение сходимости. Последовательность (v_n) векторов нормированного

пространства сходится к вектору v, если числовая последовательность $(\|v_n - v\|)$ является бесконечно малой, т.е. сходится к нулю.

В конечномерном евклидовом пространстве верны многие теоремы математического анализа, касающиеся сходимости. В частности, верен **критерий Коши**: последовательность (v_n) сходится тогда и только тогда, когда она **фундаментальна**, т.е. $(\|v_n - v_m\|)$ стремится к нулю при $n,m \to +\infty$.

В бесконечномерных евклидовых пространствах критерий Коши может не иметь места. Если для нормированного пространства V выполняется критерий Коши, т.е. любая фундаментальная последовательность сходится, то такое нормированное пространство называют **полным** или **банаховым**. Если евклидово пространство, как нормированное с евклидовой нормой, является полным, его называют **гильбертовым**. Иногда есть дополнительное требование, что гильбертово пространство бесконечномерно.

Есть еще одна особенность бесконечномерных нормированных пространств. В конечномерном случае используют термин *подпространство* для подмножеств векторного пространства, замкнутых относительно линейных операций. Каждое подпространство конечномерного нормированного пространства является замкнутым множеством, т.е. предельный переход не выводит за пределы такого множества. В бесконечномерном случае множество, замкнутое относительно линейных операций, может не быть замкнутым множеством в смысле топологическом (т.е. относительно предельного перехода). В связи с этим в функциональном анализе используют два термина:

— множество, замкнутое относительно линейных операций (алгебраически замкнутое), называют **линейным многообразием**;

— множество, замкнутое и алгебраически, и топологически, называют **векторным подпространством**.

Пример 1. *Примером полного нормированного пространства является множество действительнозначных функций, непрерывных на отрезке $[a,b]$, с обычными операциями сложения и умножения на число и с нормой*

$$\|f\|_\infty = \sup_{x\in[a,b]} |f(x)|.$$

Его обозначают $C([a,b])$. Сходимость последовательности функций по указанной норме — это ***равномерная сходимость****.*

В качестве множества в этом пространстве, алгебраически замкнутого, можно указать совокупность всех многочленов. Это множество не является подпространством, поскольку, например, функция $f(x) = e^x$ есть сумма степенного ряда, сходящегося на любом отрезке равномерно. Более того, согласно известной теореме Вейерштрасса, любая функция, непрерывная на отрезке, является равномерным пределом некоторой последовательности многочленов. Это заключение означает, что множество всех многочленов ***всюду плотно*** *в $C([a,b])$, т.е. замыкание этого множества совпадает с $C([a,b])$.*

Итак, множество всех многочленов есть всюду плотное линейное многообразие в $C([a,b])$. Пример подпространства в $C([a,b])$ — множество всех непрерывных на $[a,b]$ функций, обращающихся в ноль в точке a (также в любой другой точке и на любом замкнутом подмножестве отрезка $[a,b]$).

Пример 2. *Канонический пример гильбертова пространства — множество l_2 числовых последовательностей $x = (x_n)$, удовлетворяющих условию*

$$\|x\|^2 = \sum_{n=1}^{\infty} x_n^2 < +\infty.$$

Скалярным произведением в l_2 является

$$(x,y) = \sum_{n=1}^{\infty} x_n y_n$$

(ряд справа для любых подпоследовательностей в l_2 сходится абсолютно).

В математической физике более существенную роль играет гильбертово пространство $L_2(\Omega)$ функций, определенных на измеримом пространстве Ω и суммируемых с квадратом:

$$\|f\|^2 = \int\limits_{\Omega} |f(x)|^2 d\mu$$

(здесь μ — конечная или счетно-конечная мера на измеримом пространстве Ω). На множествах в $\mathbb{R}$ в качестве μ чаще всего используют меру Лебега. Скалярное

произведение в $L_2(\Omega)$ определяется формулой

$$(f,g) = \int\limits_{\Omega} f(x)g(x)d\mu.$$

Особенностью этих пространств является то, что интеграл по мере не изменяется, если функцию изменить на множестве меры нуль. Чтобы избежать конфликта с третьей акисомой скалярного произведения, функции, различающиеся на множестве меры ноль, считают одинаковыми. Более строго говоря, элементами гильбертова пространства $L_2(\Omega)$ являются классы функций, построенные по отношению эквивалентности

$$f \sim g \Longleftrightarrow f(x) \overset{\text{п.в.}}{=} g(x)$$

(равны почти всюду, т.е. всюду, кроме некоторого множества меры ноль).

Как и в конечномерных евклидовых пространствах, в бесконечномерных большую роль играют **ортогональные системы**, т.е. множества ненулевых элементов, в которых любые два ортогональны друг другу. Любая конечная подсистема ортогональной системы линейно независима (это ключевая теорема линейной алгебры). Поэтому в бесконечномерном евклидовом пространстве есть бесконечные ортогональные системы. Отметим, что такая система может быть даже несчетной. В связи с этим введем понятие **сепарабельного нормированного пространства** — такого нормированного пространства, в котором есть счетная всюду плотная последовательность. Рассмотренные выше векторные пространства $C([a,b])$, l_2, $L_2(\Omega)$,$\Omega \in \mathbb{R}^n$ являются сепарабельными. В сепарабельном евклидовом пространстве любая ортогональная система не более чем счетна. Мы будем рассматривать бесконечные, т.е. счетные ортогональные системы.

Рядом по ортогональной системе (e_n) называют ряд вида

$$\sum_{n=1}^{\infty} \alpha_n e_n, \quad \alpha_n \in \mathbb{R}.$$

Такой ряд называется сходящимся, если последовательность его частичных сумм $\sum_{n=1}^{N} \alpha_n e_n$ сходится. На ряды в евклидовом пространстве распространяются многие понятия теории числовых рядов.

Теорема 1. *Если ряд* $\sum_{n=1}^{N} \alpha_n e_n$ *в евклидовом пространстве* V *сходится к некоторому элементу* $v \in V$*, т.е.*

$$v = \sum_{n=1}^{N} \alpha_n e_n,$$

то

$$\alpha_n = \frac{(v,e_n)}{\|e_n\|^2}. \tag{1}$$

Доказательство. Обозначим через S_N частичную сумму ряда: $S_N = \sum_{n=1}^{N} \alpha_n e_n$. Тогда по условию $\|v - S_N\| \to 0$ при $N \to \infty$. Выберем произвольный номер k и зафиксируем. При $N > k$, учитывая представление частичной суммы, имеем

$$(v,e_k) = (v-S_n,e_k)+(S_N,e_k) = (v-S_N,e_k)+\sum_{n=1}^{N} \alpha_n (e_n,e_k) = (v-S_N,e_k)+\alpha_k \|e_k\|^2.$$

Отсюда

$$(v,e_k) - \alpha_k \|e_k\|^2 = (v - S_N,e_k).$$

Устремив N к ∞, с помощью неравенства Коши — Буняковского находим, что правая часть равенства стремится к нулю:

$$\|(v - S_N,e_k)\| \leqslant \|v - S_N\| \|e_k\|,$$

поскольку $\|v - S_N\| \to 0$ при $N \to \infty$. Но левая часть вообще не зависит от N. Значит, она равна нулю, т.е.

$$(v,e_k) = \alpha_k \|e_k\|^2.$$

Это эквивалентно равенству (1), которое выполняется для любого k, поскольку этот номер выбирался произвольно. □

Из доказанной теоремы вытекает, что если элемент v представим как сумма ряда по ортогональной системе, то коэффициенты α_n этого ряда вычисляются по формуле (1). Основной наш вопрос — представление элементов евклидова пространства (функций) в виде ряда по ортогональной системе. В этом контексте мы сразу можем считать, что элементы ряда получены для некоторого $v \in V$

по формуле (1) и остановиться на вопросе, сходится ли такой ряд и, если да, то сходится ли он к рассматриваемому элементу v.

Ряд

$$\sum_{n=1}^{\infty} \alpha_n e_n, \quad \alpha_n = \frac{(v,e_n)}{\|e_n\|^2}, \, n = 1,2,\ldots \tag{2}$$

называют **рядом Фурье** элемента x, коэффициенты α_n такого ряда — **коэффициентами Эйлера — Фурье**, а формулы, по которым вычисляются эти коэффициенты, **формулами Эйлера — Фурье**.

Теорема 2. *Для любого ряда Фурье (2) имеет место* ***неравенство Бесселя***

$$\sum_{n=1}^{\infty} |\alpha_n|^2 \|e_n\|^2 \leqslant \|v\|^2. \tag{3}$$

Доказательство. Пусть S_N — частичная сумма ряда Фурье (2). Очевидно неравенство $\|v - S_n\|^2 \geqslant 0$. Однако при этом

$$\begin{aligned} \|v - S_n\|^2 = (v - S_N, v - S_N) &= \|v\|^2 - 2(v,S_N) + (S_N,S_N) = \\ &= \|v\|^2 - 2\sum_{n=1}^{N} \alpha_n (v,e_n) + \sum_{n=1}^{N}\sum_{m=1}^{n} \alpha_n \alpha_m (e_n,e_m) = \\ &= \|v\|^2 - 2\sum_{n=1}^{N} \alpha_n^2 \|e_n\|^2 + \sum_{n=1}^{N} \alpha_n^2 \|e_n\|^2 = \|v\|^2 - \sum_{n=1}^{N} \alpha_n^2 \|e_n\|^2. \end{aligned}$$

Отсюда

$$\sum_{n=1}^{N} \alpha_n^2 \|e_n\|^2 \leqslant \|v\|^2.$$

Последнее неравенство означает, что частичные суммы знакоположительного ряда $\sum_{n=1}^{\infty} \alpha_n^2$ ограничены сверху величиной $\|v\|^2$. Значит, ряд сходится, а его сумма также не превышает $\|v\|^2$. □

Из доказательства неравенства Бесселя можно получить еще одно важное свойство. Согласно доказательству, для суммы $S_N = \sum_{n=1}^{N} \alpha_n e_n$ с произвольными коэффициентами α_n имеем

$$\|v - S_n\|^2 = \|v\|^2 - 2\sum_{n=1}^{N} \alpha_n (v,e_n) + \sum_{n=1}^{N} \alpha_n^2 \|e_n\|^2.$$

Правую часть равенства можно рассматривать как квадратичную функцию с переменными α_n. Нетрудно показать, что она имеет наименьшее значение при $\alpha_n = \frac{(v,e_n)}{\|e_n\|^2}$. Но величина $\|v - S_N\|$ показывает отклонение суммы S_N от элемента v. Таким образом, наименьшее отклонение частичных сумм ряда по ортогональной системе от элемента v будет в случае, когда этот ряд есть ряд Фурье элемента v. Это свойство известно как **минимальное свойство ряда Фурье**.

Возвращаемся к вопросу, когда ряд Фурье сходится к своему элементу.

Теорема 3. *Ряд Фурье (2) элемента v сходится к v тогда и только тогда, когда*

$$\sum_{n=1}^{\infty} |\alpha_n|^2 \|e_n\|^2 = \|v\|^2, \tag{4}$$

т.е. когда неравенство Бесселя превращается в равенство.

Доказательство. Согласно доказательству теоремы 2, имеем

$$\|v - S_N\|^2 = \|v\|^2 - \sum_{n=1}^{\infty} \alpha_n^2 \|e_n\|^2. \tag{5}$$

Сходимость ряда Фурье к v означает, что $\|v - S_N\| \to 0$ при $N \to \infty$. В этом случае в силу (5)

$$\sum_{n=1}^{N} \alpha_n^2 \|e_n\|^2 \to \|v\|^2 \quad \text{при } N \to \infty.$$

Наоборот, если имеет место (4), то $\|v - S_N\| \to 0$ при $N \to \infty$, т.е. ряд Фурье сходится к v. □

Равенство (4) известно как **равенство Парсеваля**.

Из неравенства Бесселя вытекает, что частичные суммы ряда Фурье всегда образуют фундаментальную последовательность. Действительно,

$$\|S_N - S_{N+p}\|^2 = \left| \sum_{n=N+1}^{N+p} \alpha_n e_n \right|^2 = \sum_{n=N+1}^{N+p} \alpha_n^2 \|e_n\|^2.$$

Отсюда вытекает, что фундаментальность последовательности (S_N) равносильна фундаментальности последовательности частиыных сумм ряда $\sum_{n=1}^{\infty} \alpha_n^2 \|e_n\|^2$, т.е. сходимости этого ряда, а он всегда сходится.

Из условия фундаментальности последовательности частичных сумм можно сделать заключение о сходимости ряда в случае, когда выполняется критерий Коши, т.е. когда евклидово пространство является гильбертовым.

Теорема 4. *В гильбертовом пространстве любой ряд Фурье сходится.*

Рассмотрим элемент $v \in V$ в гильбертовом пространстве V, и пусть v_0 — сумма его ряда Фурье. Положим $u = v - v_0$. Оказывается, что u ортогонален каждому элементу системы $\{e_n\}$. Действительно,

$$(u,e_k) = (v - v_0,e_k) = (v,e_k) - (x_0,e_k) = \alpha_k\|e_k\|^2 - \alpha_k\|e_k\|^2 = 0.$$

Здесь равенство $(v,e_k) = \alpha_k\|e_k\|^2$ верно по определению (коэффициенты ряда являются коэффициентами Эйлера — Фурье элемента v), а равенство $(v_0,e_k) = \alpha_k\|e_k\|^2$ вытекает из теоремы 1. Если $u \neq 0$, то этот элемент можно добавить к ортогональной системе, т.е. расширить ортогональную систему, а в результате в ряд Фурье будет добавлено новое слагаемое, связанное с u. Получается, что ряд Фурье не сходился к своему элементу потому, что в ортогональной системе были учтены не все составляющие.

Ортогональная система называется **полной**, если она не является частью какой-либо другой ортогональной системы, т.е. из условия $(u,e_n) = 0$, $n = 1,2,\ldots$, вытекает, что $u = 0$. Если в гильбертовом пространстве ортогональная система $\{e_n\}$ является полной, то любой ряд Фурье по этой системе сходится к своему элементу. Мы получили свойство-критерий того, что любой элемент представим рядом по ортогональной системе: во-первых, евклидово пространство должно быть полным (гильбертовым), а во-вторых, ортогональная система должна быть полной.

Условие полноты ортогональной системы на самом деле не является простым для проверки. Кроме того, требуется еще полнота самого евклидова пространства. Есть еще один критерий представимости любого элемента рядом по ортогональной системе.

Ортогональную систему $\{e_n\}$ в евклидовом пространстве V называют **замкнутой**, если множество конечных линейных комбинаций элементов ортогональной системы всюду плотно в V. Если система $\{e_n\}$ замкнута, то для любого элемента v и для любого $\varepsilon > 0$ можно выбрать такую линейную комбинацию

$S_N = \sum_{n=1}^{N} \alpha_n e_n$ с некоторыми коэффициентами α_n, что $\|v - S_N\| < \varepsilon$. Но в силу минимального свойства этому же неравенству удовлетворяет и частичная сумма ряда Фурье элемента v, причем не только сумма, соответствующая индексу N, но и все последующие. Это значит, что ряд Фурье сходится к элементу x. Наоборот, если каждый элемент есть сумма своего ряда Фурье, то система $\{e_n\}$ замкнута, поскольку всюду плотное множество образуют всевозможные частичные суммы рядов Фурье.

Теорема 5. *В гильбертовом пространстве ортогональная система замкнута тогда и только тогда, когда она полна.*

Доказательство. Условие полноты следует из условия замкнутости в любом евклидовом пространстве. Действительно, если $(u,e_n) = 0$, $n = 1,2\ldots$, то ряд Фурье элемента u имеет все коэффициенты, равные нулю. Такой ряд, естественно, сходится к нулю. Но из рассуждений выше вытекает, что для замкнутой системы любой элемент есть сумма своего ряда Фурье. Значит, $u = 0$. Тем самым полнота доказана. Если пространство гильбертово, а ортогональная система полна, то каждый элемент есть сумма своего ряда Фурье. В этом случае ортогональная система замкнута. □

1.2 Тригонометрические ряды

Пример ортогональных систем дают тригонометрические ряды.

Рассмотрим гильбертово пространство $L_2([-l,l])$ и в нем ортогональную систему $\{e_n\}_{n\in\mathbb{Z}_{\geqslant 0}}$ такую, что $e_0 = 1$, $e_{2k} = \sin\frac{k\pi x}{l}$,$k > 0$, $e_{2m+1} = \cos\frac{m\pi x}{l}$,$m \geqslant 0$. Эта система является ортогональной. Например,

$$\int_{-l}^{l} \cos\frac{m\pi x}{l}\sin\frac{k\pi x}{l} = \frac{1}{2}\int_{-l}^{l}\left(\sin\frac{(k+m)\pi x}{l} - \sin\frac{(k-m)\pi x}{l}\right)dx = 0.$$

Квадраты норм:

$$\int_{-l}^{l} 1\cdot dx = 2l, \quad \int_{-l}^{l}\cos^2\frac{n\pi x}{l} = \int_{-l}^{l}\sin^2\frac{n\pi x}{l} = l.$$

Ряд по этой ортогональной системе записывают в форме

$$\frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos \frac{n\pi x}{l} + b_n \sin \frac{n\pi x}{l} \right)$$

и называют **тригонометрическим рядом**.

Такой ряд можно составить как ряд Фурье функции $f(x)$, суммируемой на отрезке $[-l,l]$, для чего коэффициенты нужно вычислить по формулам Эйлера-Фурье:

$$a_n = \frac{1}{l} \int_{-l}^{l} f(x) \cos \frac{n\pi x}{l} dx,\ n \geqslant 0; \quad b_n = \frac{1}{l} \int_{-l}^{l} f(x) \sin \frac{n\pi x}{l} dx,\ n > 0. \quad (6)$$

Согласно варианту теоремы Вейерштрасса, в $L_2([-l,l])$ всюду плотны **тригонометрические многочлены** — частичные суммы тригонометрических рядов. Значит, тригонометрическая система замкнута в $L_2([-l,l])$, а любой ряд Фурье сходится к своей функции в $L_2([-l,l])$, т.е.

$$\int_{-l}^{l} \left| f(x) - \frac{a_0}{2} - \sum_{n=1}^{N} \left(a_n \cos \frac{n\pi x}{l} + b_n \sin \frac{n\pi x}{l} \right) \right|^2 dx \to 0 \quad \text{при } N \to \infty.$$

Условие сходимости на самом деле непростое и отнюдь не гарантирует сходимости ряда Фурье к своей основной функции в каждой точке. Можно лишь утверждать, что любая функция может быть восстановлена по своему ряду Фурье с точностью до значений на множестве меры ноль. Как это сделать, пока не ясно.

Можно сформулировать дополнительные условия, при которых ряд Фурье сходится к своей функции поточечно. Например, если ряд $\sum_{n=1}^{\infty}(|a_n| + |b_n|)$ сходится, то тригонометрический ряд сходится равномерно. Из равномерной сходимости вытекает сходимость его к той же функции в $L_2([-l,l])$. Значит, равномерно он сходится к своей функции, которую можно представить как непрерывную функцию. Сформулированное условие несложно проверить по известным коэффициентам. Например, так будет, если $a_n = O\left(\frac{1}{n^{1+\varepsilon}}\right)$, $b_n = O\left(\frac{1}{n^{1+\varepsilon}}\right)$,.

Другой вариант условий — на исходную функцию. Следующая теорема носит название **теоремы Дирихле**.

Теорема 6. *Пусть функция $f(x)$ на отрезке $[-l,l]$ удовлетворяет условиям:*

1. *$f(x)$ кусочно непрерывна, т.е. имеет на $[-l,l]$ конечное число точек разрыва, и все они первого рода;*
2. *$f(x)$ кусочно монотонна, т.е. отрезок $[-l,l]$ можно так разделить на конечное число подотрезков, что на каждом из этих подотрезков функция монотонна;*
3. *в каждой точке разрыва выполняется равенство $f(x) = \frac{f(x-0)+f(x+0)}{2}$, т.е. значение функции в точке разрыва равно полусумме односторонних пределов функции в этой точке, а в концах отрезка — равенство $f(-l) = f(l) = \frac{f(l-0)+f(-l+0)}{2}$.*

Тогда в каждой точке отрезка выполняется равенство

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos \frac{n\pi x}{l} + b_n \sin \frac{n\pi x}{l} \right),$$

где коэффициенты a_n, b_n вычислены по формулам Эйлера — Фурье (6).

Условия 1 и 2 в сформулированной теореме Дирихле называют **условиями Дирихле**. Эти условия взаимосвязаны. Так из кусочной монотонности вытекает отсутствие точек разрыва второго рода (правда, монотонная функция может иметь на отрезке счетное число точек разрыва). При наличии условия 1 условие 2 может быть сформулировано как существование конечного числа точек локального экстремума. Но в целом эти условия носят глобальный характер, поскольку характеризуют поведение функции на отрезке в целом. Существуют локальные условия.

Теорема 7. *Пусть функция $f(x)$ интегрируема (по Лебегу) на отрезке $[-l,l]$, а в каждой точке $x_0 \in [-l,l]$ выполняются условия:*

1. *существуют односторонние пределы $f(x_0 - 0)$ и $f(x_0 + 0)$;*
2. *существуют односторонние производные $f'(x_0 - 0) = \lim_{h \to -0} \frac{f(x_0+h)-f(x_0-0)}{h}$, $f'(x_0 + 0) = \lim_{h \to +0} \frac{f(x_0+h)-f(x_0+0)}{h}$.*

   Тогда

$$f(x_0) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos \frac{n\pi x_0}{l} + b_n \sin \frac{n\pi x_0}{l} \right),$$

   где коэффициенты a_n, b_n вычислены по формулам Эйлера — Фурье (6).

Данная теорема носит название **теоремы Дини**, а условия, накладываемые на функцию в ее формулировке — **условиями Дини**.

Теоремы Дирихле и Дини позволяют делать заключения о поточечной сходимости ряда Фурье по свойствам функции.

1.3 Самосопряженные операторы в гильбертовом пространстве

В конечномерном нормированном пространстве любой линейный оператор является непрерывным. В бесконечномерном случае это не так.

Определение 4. *Линейный оператор $F : V \to V$ на нормированном пространстве V называется **ограниченным**, если для некоторой константы $C > 0$ выполняется условие $\|F(v)\| \leqslant C\|v\|$, $v \in V$. Точная нижняя грань таких значений C называется **нормой ограниченного оператора**.*

Норму ограниченного оператора можно вычислить по формуле

$$\|F\| = \sup_{v \neq 0} \frac{\|F(v)\|}{\|v\|} = \sup_{\|v\|=1} \|F(v)\|.$$

Линейный оператор непрерывен, если для любого v_0 и любого $\varepsilon > 0$ существует такое $\delta > 0$, что $\|F(v) - F(v_0)\| < \varepsilon$ при $\|v - v_0\| < \delta$. Непрерывность линейного оператора в некоторой точке v_0 означает, что он непрерывен и в любой другой точке. Поэтому условие непрерывности можно формулировать лишь для точки $v_0 = 0$: линейный оператор F непрерывен, если для любого $\varepsilon > 0$ существует такое $\delta > 0$, что $\|F(v)\| < \varepsilon$ при $\|v\| < \delta$. Довольно очевидно, что ограниченный оператор является непрерывным: в определении непрерывности достаточно установить $\delta = \varepsilon/\|F\|$.

Лемма 1. *Непрерывный оператор в нормированном пространстве является ограниченным.*

Доказательство. Пусть линейный оператор F непрерывен. Выберем $\varepsilon = 1$ и по нему выберем соответствующее значение $\delta > 0$. Если $\|v\| = \delta/2$, то, согласно

условию непрерывности, имеем $\|F(v)\| < \varepsilon = 1$. Для произвольного $u \in V$

$$F(u) = \frac{2\|u\|}{\delta} F\left(\frac{\delta u}{2\|u\|}\right).$$

Полагая $x = \frac{\delta u}{2\|u\|}$, заключаем, что $\|v\| = \delta/2$. Поэтому

$$\|F(u)\| = \frac{2\|u\|}{\delta}\|F(x)\| \leqslant \frac{2}{\delta}\|u\| = C\|u|.$$

□

Таким образом, в нормированном пространстве понятия ограниченности и непрерывности линейного оператора эквивалентны.

В бесконечномерном векторном пространстве есть линейные операторы, не являющиеся непрерывными (ограниченными).

Пример 3. *Рассмотрим множество бесконечно дифференцируемых функций на отрезке* $[a,b]$. *Введем норму* $\|f\|_\infty = \sup_{x\in[a,b]} |f(x)|$. *Рассмотрим линейный оператор* $\frac{d}{dx}$, *который каждой функции ставит в соответствие ее производную:* $\frac{d}{dx}(f) = f'$. *Этот оператор не является ограниченным, поскольку функции* $f_n(x) = \sin nx$, $n \in \mathbb{N}$, *имеют по крайней мере начиная с некоторого номера, норму, равную* 1, *и в то же время функции* $\frac{d}{dx}(f_n)(x) = n\cos nx$ *имеют норму, равную* n. *Ясно, что неравенство* $\|\frac{d}{dx}(f_n)\| \leqslant C\|f_n\|$ *ни при каком* C *для всех* n *не выполняется.*

В банаховом пространстве типично, что линейный оператор, не являющийся непрерывным, определен не на всем векторном пространстве, а на не котором линейном многообразии, которое называют **областью определения линейного оператора**. Будем область определения линейного оператора F обозначать $\operatorname{dom} F$. Поскольку $\operatorname{dom} F$ — линейное многообразие, его замыкание — подпространство, которое, рассматриваемое как самостоятельное, будет банаховым пространством. Можно изначально ограничиваться этим подпространством, что позволяет считать, что $\operatorname{dom} F$ всюду плотно в рассматриваемом банаховом пространстве. В этом случае говорят, что **линейный оператор** F **плотно определен**.

Плотно определенный оператор в гильбертовом пространстве называется **самосопряженным**, если $(F(v),u) = (v,F(u))$ для любой пары элементов $v,u \in \operatorname{dom} F$.

Комплексное число $\lambda \in \mathbb{C}$ называется **регулярным** для линейного оператора $F : V \to V$, если линейный оператор $F - \lambda \operatorname{Id}$ биективен, причем обратный оператор является ограниченным. Множество всех регулярных значений называется **резольвентным множеством**. Дополнение в $\mathbb{C}$ к резольвентному множеству называется **спектром линейного оператора** и обозначается $\operatorname{Spec} F$. Таким образом, число λ принадлежит $\operatorname{Spec} F$, если оператор $F - \lambda \operatorname{Id}$ не имеет ограниченного обратного. Последнее может быть по нескольким причинам. Во-первых, линейный оператор $F - \lambda \operatorname{Id}$ может не быть инъективным, т.е. иметь ненулевое ядро. В этом случае существует ненулевой вектор v, для которого $(F - \lambda \operatorname{Id})(v) = 0$, или, что эквивалентно, $F(v) = \lambda v$. В таком случае вектор v называется **собственным вектором** линейного оператора F, а соответствующее число λ — **собственным числом** (или **собственным значением**) линейного оператора F. В этом случае говорят, что число $\lambda \in \mathbb{C}$ принадлежит **дискретному спектру**. Во-вторых, этот оператор может не быть сюръективным, или он биективен, но обратный оператор не является ограниченным. В таком случае говорят, что число λ принадлежит **непрерывному спектру**. Ядро оператора $F - \lambda \operatorname{Id}$ называется **собственным подпространством** оператора F, соответствующем собственному числу λ и обозначается $V_\lambda := \ker(F - \lambda \operatorname{Id})$.

В случае самосопряженных линейных операторов верна теорема, используемая в линейной алгебре.

Теорема 8. *Собственные векторы, отвечающие разным действительным собственным значениям, ортогональны.*

Доказательство. Пусть $F(v_1) = \lambda_1 v_1$, $F(v_2) = \lambda_2 v_2$, где $\lambda_1 \neq \lambda_2$, $v_1, v_2 \neq 0$. Тогда

$$\lambda_1(v_1, v_2) = (F(v_1), v_2) = (v_1, F(v_2)) = \lambda_2(v_1, v_2).$$

Поэтому $\lambda_1(v_1, v_2) - \lambda_2(v_1, v_2) = 0$, откуда $(\lambda_1 - \lambda_2)(v_1, v_2) = 0$. Так как $\lambda_1 \neq \lambda_2$, то $(v_1, v_2) = 0$. □

Именно самосопряженные операторы являются поставщиками ортогональных систем. Отметим, что если собственное число λ является кратным, т.е. размерность собственного подпространства V_λ больше единицы, то этому собственному числу λ соответствует несколько линейно независимых векторов, причем нет никаких оснований считать, что они ортогональны. Для обеспечения этого свойства

нужно, выбрав в собственном подпространстве систему независимых векторов, провести процесс ортогонализации.

2 Линейные системы

2.1 Многомерные стационарные линейные системы

Многомерными системами или **системами многсвязного управления** называют автоматические системы управления, в которых имеется несколько управляемых величин. Соответственно объекты, имеющие несколько управляемых величин, называют `многомерными объектами` или `объектами многосвязного управления`. [5]

Примерами многомерных объектов могут быть: самолет, у которого управляемыми величинами являются курс, углы тангажа и крена, высота, скорость; паровой котел, в котором регулируется температура, давление пара и другие величины.

Обычно управляемые величины называют выходами или выходными величинами. И поэтому многомерные системы еще определяют как автоматические системы с многомерным (векторным) выходом.

Многомерные системы и объекты называют **линейными** и **стационарными**, если они описываются системой линейных дифференциальных уравнений с постоянными коэффициентами.

Пусть $y_1, \ldots, y_k$ обозначают выходные величины, $u_1, \ldots, u_m$ — параметры управления или задающие воздействия и $f_1, \ldots, f_l$ — возмущающие воздействия. Уравнения многомерных стационарных линейных систем и объектов в общем случае можно записать в виде следующей системы:

$$\sum_{j=1}^{k} a_{ij}(p) y_j = \sum_{j=1}^{m} b_{ij}(p) u_j + \sum_{j=1}^{l} c_{ij}(p) f_j, \quad i = \overline{1,k}. \tag{7}$$

Здесь $a_{ij}(p)$, $b_{ij}(p)$, $c_{ij}(p)$ — стационарные линейные операторы, т.е. многочлены от оператора дифференцирования.

Переходя в обоих частях (7) к изображениям Лапласа, при нулевых начальных условиях получим систему алгбераических уравнений:

$$\sum_{j=1}^{k} a_{ij}(s) Y_j(s) = \sum_{j=1}^{m} b_{ij}(s) U_j(s) + \sum_{j=1}^{l} c_{ij}(s) F_j(s), \quad i = \overline{1,k}, \tag{8}$$

где

$$Y_j(s) = L[y_j(t)], \quad U_j(s) = L[u_j(t)], \quad F_j(s) = L[f_j(t)].$$

Для многомерных систем удобна матричная форма записи уравнений. Введем в рассмотрение матрицы

$$y = \begin{pmatrix} y_1 \\ \vdots \\ y_k \end{pmatrix}, \quad A(p) = \begin{pmatrix} a_{11}(p) & \cdots & a_{1k}(p) \\ \vdots & \ddots & \vdots \\ a_{k1}(p) & \cdots & a_{kk}(p) \end{pmatrix}, \quad u = \begin{pmatrix} u_1 \\ \vdots \\ u_m \end{pmatrix}$$

$$B(p) = \begin{pmatrix} b_{11}(p) & \cdots & b_{1m}(p) \\ \vdots & \ddots & \vdots \\ b_{k1}(p) & \cdots & b_{km}(p) \end{pmatrix}, \quad f = \begin{pmatrix} f_1 \\ \vdots \\ f_l \end{pmatrix}, \quad C(p) = \begin{pmatrix} c_{11}(p) & \cdots & c_{1l}(p) \\ \vdots & \ddots & \vdots \\ c_{k1}(p) & \cdots & b_{kl}(p) \end{pmatrix}.$$

С их помощью (7) в матричной форме будет

$$A(p)y = B(p)u + C(p)f. \tag{9}$$

Точно так же можно записать (9) в изображениях Лапласа в матричной форме:

$$A(s)Y(s) = B(s)U(s) + C(s)F(s). \tag{10}$$

Здесь

$$A(s) = \begin{pmatrix} a_{11}(s) & \cdots & a_{1k}(s) \\ \vdots & \ddots & \vdots \\ a_{k1}(s) & \cdots & a_{kk}(s) \end{pmatrix}, \quad Y(s) = \begin{pmatrix} Y_1(s) \\ \vdots \\ Y_k(s) \end{pmatrix},$$

$$B(s) = \begin{pmatrix} b_{11}(s) & \cdots & b_{1m}(s) \\ \vdots & \ddots & \vdots \\ b_{k1}(s) & \cdots & b_{km}(s) \end{pmatrix},$$

$$U(s) = \begin{pmatrix} U_1(s) \\ \vdots \\ U_m(s) \end{pmatrix}, \quad C(s) = \begin{pmatrix} c_{11}(s) & \cdots & c_{1l}(s) \\ \vdots & \ddots & \vdots \\ c_{k1}(s) & \cdots & b_{kl}(s) \end{pmatrix}, \quad F(s) = \begin{pmatrix} F_1(s) \\ \vdots \\ F_l(s) \end{pmatrix}.$$

В (9), после умножения и сложения матриц, в правой и левой частях получатся матрицы-столбцы. Приравняв их соответственные элементы, получим систему

уравнений (7). Аналогично, выполнив указанные операции над матрицами и приравняв соответствующие элементы матриц левой и правой частей матричного уравнения (10), получим систему (8).

Пусть управляющий параметр $u_j = \delta(t)$, а остальные управляющие параметры и возмущающие воздействия равны нулю. При этом решение (7) многомерной системы при нулевых начальных условиях обозначим $w^u_{1j}(t)$,$w^u_{2j}(t)$, ...,$w^u_{kj}(t)$. Эти функции называют весовыми или импульсными переходными функциями. Функция $w_{ij}(t)$ описывает реакцию системы на i-м выходе при действии в точке приложения j-го параметра управления единичного импульса и называется **импульсной переходной** или **весовой** функцией по j-му параметру управления и i-му выходу. Матрицу

$$w^u(t) = \begin{pmatrix} w^u_{11}(t) & \cdots & w^u_{1m}(t) \\ \vdots & \ddots & \vdots \\ w^u_{k1}(t) & \cdots & w^u_{km}(t) \end{pmatrix},$$

составленную из весовых функций по управлению, называют **импульсной переходной** или **весовой матрицей** по управлению. Аналогично определяют импульсную переходную или весовую матрицу по возмущению:

$$w^f(t) = \begin{pmatrix} w^f_{11}(t) & \cdots & w^f_{1m}(t) \\ \vdots & \ddots & \vdots \\ w^f_{k1}(t) & \cdots & w^f_{km}(t) \end{pmatrix}.$$

Здесь $w^f_{1j}(t), \ldots, w^f_{kj}(t)$ — решение (7) многомерной системы, когда $f_j = \delta(t)$, а все остальные возмущающие воздействия и параметры управления равны нулю.

Весовые матрицы дают полное описание многомерной системы (объекта).

2.2 Нестационарные линейные системы

Нестационарными линейными системами или **линейными системами с переменными параметрами** называют системы, которые описываются линейными дифференциальными уравнениями с переменными коэффициентами. Для их описания помимо дифференциальных уравнений могут быть использованы передаточные функции, переходные и весовые (импульсные переходные) функции,

частотные функции и их характеристики. Кроме того, для графического представления нестационарных систем могут быть использованы структурные схемы и графы. Однако методы, основанные на графических представлениях, не так эффективны, как в случае стационарных систем.

Рассмотрим некоторые способы описания одномерных нестационарных систем. Они могут быть обобщены на многомерные системы так, как это было сделано при описании стационарных линейных систем.

Так как для линейных систем (как стационарных, так и нестационарных) справедлив принцип суперпозиции, то для простоты можем ограничиться рассмотрением систем с одним входом.

Уравнение одномерной нестационарной системы (объекта) с одним входом в общем случае можно записать в виде

$$a_0(t)y^{(n)} + a_1 y^{(n-1)} + \cdots + a_n(t)y = b_0(t)u^{(m)} + b_1(t)u^{(m-1)} + \cdots + b_m(t)u \quad (11)$$

или, в операторной форме

$$Q(p,t)y = R(p,t)u, \quad (12)$$

где нестационарные линейные дифференциальные операторы

$$Q(p,t) = a_0(t)p^n + a_1(t)p^{n-1} + \cdots + a_n(t)$$
$$R(p,t) = b_0(t)p^m + b_1(t)p^{m-1} + \cdots + b_m(t)$$

Весовой функцией называют решение уравнения (11) при $u(t) = \delta(t-\tau)$ и нулевых «начальных» условиях, т.е. функцию, которая описывает реакцию на единичный импульс системы, находящейся в момент приложения импульса в исходном состоянии. Здесь τ обозначает момент приложения импульса и в определении под начальными условиями понимают значения выходной величины и ее производных в момент времени τ. При рассмотрении стационарных систем обычно в качестве начала отсчета времени принимают момент приложения входного сигнала, и поэтому в этих случаях полагают $\tau = 0$. В данном случае этого делать нельзя. Реакция нестационарной системы зависит не только от вермени $t - \tau$, отсчитываемого от момента приложения импульса, но и от самого значения τ. Поэтому весовая функция нестационарной системы — обозначим ее $w(t-\tau,\tau)$

— является функцией от двух переменных: от текущего времени t и момента τ приложения импульса.

Реакция — процесс на выходе системы — не может возникнуть до приложения входного сигнала: следствие не может предшествовать причине. Поэтому

$$w(t-\tau,\tau) \equiv 0 \quad \text{при} \quad t < \tau. \tag{13}$$

Это условие называют условием *физической осуществимости*. В случае стационарной системы условие физической осуществимости имеет вид $w(t) \equiv 0$ при $t < 0$.

Получим формулы, определяющие связь между выходной и входной величинами нестационарной линейной системы управления через ее весовую функцию. Так как, по определению, $w(t-\tau,\tau)$ есть решение уравнения (12) при $u(t) = \delta(t-\tau)$, то можем записать

$$Q(p,t)w(t-\tau,\tau) = R(p,t)\delta(t-\tau). \tag{14}$$

Умножим обе части на $u(\tau)d\tau$ и проинтегрируем по τ от $-\infty$ до ∞, а затем, вынеся коэффициенты уравнения за знак интеграла (это возможно, так как они не зависят от τ) и поменяв местами операции интегрирования и дифференцирования, получим

$$Q(p,t)\int\limits_{-\infty}^{\infty} w(t-\tau,\tau)u(\tau)d\tau = R(p,t)\int\limits_{-\infty}^{\infty} u(\tau)\delta(t-\tau)d\tau. \tag{15}$$

Из определения дельта-функции

$$\int\limits_{-\infty}^{\infty} u(\tau)\delta(t-\tau)d\tau = u(t),$$

поэтому (15) можно переписать в виде

$$Q(p,t)\int\limits_{-\infty}^{\infty} w(t-\tau,\tau)u(\tau)d\tau = R(p,t)u(t).$$

Из последнего равенства, которое выполняется тождественно, вытекает, что функция

$$y(t) = \int_{-\infty}^{\infty} w(t-\tau,\tau)u(\tau)d\tau \tag{16}$$

является решением уравнения (12) при произвольном заданном $u(t)$. Нижний предел интегрирования $\tau = -\infty$ в (16) совпадает с моментом подачи входного воздействия. Поэтому (16) является искомой формулой, определяющей связь между выходной и входной величинами нестационарной линейной системы в «установившемся» режиме. Учитывая условие физической осуществимости (13), формулу (16) можно записать также в виде

$$y(t) = \int_{-\infty}^{t} w(t-\tau,\tau)u(\tau)d\tau. \tag{17}$$

Аналогично, умножив обе части равенства (14) на $u(\tau)d\tau$ и проинтегрировав их от 0 до ∞, получим формулу

$$y(t) = \int_{0}^{\infty} w(t-\tau,\tau)u(\tau)d\tau,$$

определяющую выходную величину нестационарной линейной системы, когда на ее вход подается воздействие $u(t)$ в момент $t = 0$. С учетом физической осуществимости ее также можно записать в виде

$$y(t) = \int_{0}^{t} w(t-\tau,\tau)u(\tau)d\tau.$$

Если зафиксировать переменную τ, то весовая функция $w(t-\tau,\tau)$ будет функцией от одной переменной t, зависящей от параметра τ, и называться **нормальной весовой функцией**. Нормальная весовая функция определяет изменение выходной величины системы с течением времени при подаче на вход единичного импульса в заданный момент τ.

Если зафиксировать переменную t — рассматривать ее как параметр, — то весовая функция $w(t-\tau,\tau)$ будет функцией одной переменной τ и называться **со-**

пряженной весовой функцией. Сопряженная функция определяет зависимость реакции системы в фиксированный момент t от момента τ приложения единичного импульса.

2.3 Многомерные нестационарные линейные системы

Рассмотрим многомерную линейную систему, поведение которой описывается системой дифференциальных уравнений вида

$$\begin{cases} \dot{x_1} = a_{11}(t)x_1 + \cdots + a_{1n}(t)x_n + b_{11}(t)y_1 + \cdots + b_{1m}(t)y_m; \\ \dot{x_2} = a_{21}(t)x_1 + \cdots + a_{2n}(t)x_n + b_{21}(t)y_1 + \cdots + b_{2m}(t)y_m; \\ \cdots\cdots\cdots \\ \dot{x_n} = a_{n1}(t)x_1 + \cdots + a_{nn}(t)x_n + b_{n1}(t)y_1 + \cdots + b_{nm}(t)y_m; \end{cases} . \tag{18}$$

Введем обозначения:

$$A(t) = \begin{pmatrix} a_{11}(t) & \dots & a_{1n}(t) \\ \vdots & \ddots & \vdots \\ a_{n1}(t) & \dots & a_{nn}(t) \end{pmatrix} \quad B(t) = \begin{pmatrix} b_{11}(t) & \dots & b_{1m}(t) \\ \vdots & \ddots & \vdots \\ b_{n1}(t) & \dots & b_{nm}(t) \end{pmatrix}$$

$$X = (x_1, \dots, x_n)^t \quad Y = (y_1, \dots, y_m)^t.$$

Используя этио обозначения, система (18) может быть записана в виде одного векторно-матричного уравнения

$$\dot{X} = A(t)X + B(t)Y. \tag{19}$$

Алгоритмы анализа многомерных систем, описываемых уравнением (35) можно построить, используя частные свойства полиномов Чебышева 1-го рода. [6]

Сведем записанное уравнение (35) к эквивалентному интегральному уравнению

$$X(t) - X^0 = \int_0^t A(\tau)X(\tau)d\tau + \int_0^t B(\tau)Y(\tau)d\tau.$$

При этом воспользуемся спектральным представлением векторов $X(t)$, $Y(t)$ и матриц $A(t)$ и $B(t)$:

$$\widetilde{X}(t) = \sum_{i=0}^{m-1} C_i^X T_i^*(t), \quad \widetilde{Y}(t) = \sum_{i=0}^{m-1} C_i^Y T_i^*(t),$$
$$\widetilde{A}(t) = \sum_{i=0}^{m-1} C_i^A T_i^*(t), \quad \widetilde{B}(t) = \sum_{i=0}^{m-1} C_i^B T_i^*(t).$$

Для расчета C_i^Y, C_i^A, C_i^B можно применить алгоритм *быстрого преобразования Фурье*. Тогда справедлива зависимость

$$\widetilde{A}(t)\widetilde{X}(t) = \left(C_0^A, C_1^A, \ldots, C_{m-1}^A\right) \begin{pmatrix} C_0^X & C_1^X & \ldots & C_{m-1}^X \\ \frac{C_1^X}{2} & C_0 + \frac{C_2^X}{2} & \ldots & \frac{C_{m-2}^X}{2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{C_{m-1}^X}{2} & \frac{C_{m-2}^X}{2} & \ldots & C_0^X \end{pmatrix} \times$$
$$\times \begin{pmatrix} T_0^*(t) \\ T_1^*(t) \\ \vdots \\ T_{m-1}^*(t) \end{pmatrix} = \left(C_0^A, C_1^A, \ldots, C_{m-1}^A\right) \widetilde{C}^X T^*(t).$$

Аналогично,

$$\widetilde{B}(y)\widetilde{Y}(t) = \left(C_0^B, C_1^B, \ldots, C_{m-1}^B\right) \widetilde{C}^Y T^*(t).$$

Подставляя полученные выражения в интегральное уравнение, найдем

$$\left(C_0^X, C_1^X, \ldots, C_{m-1}^X\right) T^*(t) - \left(X^0, \underbrace{0, \ldots, 0}_{m-1}\right) T^*(t) =$$
$$= \left(C_0^A, C_1^A, \ldots, C_{m-1}^A\right) \widetilde{C}^X P T^*(t) + \left(C_0^B, C_1^B, \ldots, C_{m-1}^B\right) \widetilde{C}^Y P T^*(t).$$

Отсюда достаточно просто получить систему линейных алгебраических уравнений для определения C_i^X:

$$\left(C_0^X, C_1^X, \ldots, C_{m-1}^X\right) - \left(X^0, 0, \ldots, 0\right) =$$
$$= \left(C_0^A, C_1^A, \ldots, C_{m-1}^A\right) \widetilde{C}^X P + \left(C_0^B, C_1^B, \ldots, C_{m-1}^B\right) \widetilde{C}^Y P.$$

3 Метод Галёркина

Методы Галёркина к настоящему времени были применены при решении многочисленных задач механики конструкций, динамики сооружений, гидромеханики, теории гидродинамической устойчивости, магнитной гидродинамики, теории гидродинамической устойчивости, магнитной гидродинамики, теории тепло- и массообмена, акустики, теории распространения микроволн, теории переноса нейтронов и т. п. С помощью представлений Галёркина были проведены исследования обыкновенных дифференциальных уравнений, дифференциальных уравнений в частных производных и интегральных уравнений. Стационарные и нестационарные задачи, а также задачи на собственные значения оказались в равной степени поддающимися исследованию на основе подходов Галёркина. По существу, любая задача, для которой можно выписать определяющие уравнения, может быть решена с помощью одной из разновидностей метода Галёркина.

Происхождение метода обычно связывается со статьей, опубликованной Галёркиным [7], и посвященной упругому равновесию стержней и тонких пластин. Борис Григорьевич Галёркин, русский инженер и специалист по прикладной механике, родился в 1871 г., в 1899 г. окончил Технологический институт в Санкт-Петербурге и вслед за этим приобрел инженерный опыт, работая на Харьковском паровозостроительном заводе. Известно, что Галёркин начал свою очень плодотворную исследовательскую деятельность в период пребывания к тюрьме в 1906-1907 гг., куда он и попал за свои антимонархические взгляды. Первое назначение на преподавательскую должность Галёркин получил в Санкт-Петербургском политехническом институте в 1909 г. В 1920 г. Галёркин стал деканом факультета прикладной механики в том же учебном заведении, а в последующие годы своей замечательной карьеры занимал различные академические посты преимущественно в Ленинграде. [8]

Важнейшие особенности метода Галёркина можно сформулировать в следующей очень компактной форме. Согласно предположению, некая двумерная задача описывается линейным дифференциальным уравнением

$$L(z) = 0 \tag{20}$$

в области $D(x,y)$ при граничных условиях

$$S(z) = 0 \tag{21}$$

на линии ∂D, являющейся границей области D. В методе Галёркина предполагается, что неизвестная z может быть достаточно точно представлена приближенным решением

$$z_a(x,y) = \varphi_0(x,y) + \sum_{k=1}^{n} a_k \varphi_k(x,y), \tag{22}$$

где φ_k — это известные аналитические функции, часто называемые *пробными*; функция φ_0 введена, чтобы удовлетворять граничным условиям, тогда как a_k — это коэффициенты, подлежащие определению. Подстановка выражения (22) в уравнение (20) приводит к отличной от нуля *невязке* R, выражаемой в виде

$$R = L(z_a) = L(\varphi_0) + \sum_{k=1}^{n} a_k L(\varphi_k). \tag{23}$$

Удобно дать следующее определение скалярного произведения:

$$(f,g) = \iint\limits_D f(x,y) g(x,y) dx dy. \tag{24}$$

При обращению к методу Галёркина неизвестные коэффициенты a_k, входящие в выражение (22), должны определяться из решения следующей системы уравнений:

$$(R,\varphi_l) = 0, \quad l = \overline{1,n}. \tag{25}$$

Здесь R — невязка данного уравнения, а φ_l — те же самые аналитические функции, которые фигурируют в (22). Предложенный здесь пример связан с решением линейного дифференциального уравнения, а поэтому уравнения (25) могут быть записаны непосредственно в форме матричного уравнения относительно коэффициентов a_k, а именно

$$\sum_{k=1}^{n} a_k (L(\varphi_k),\varphi_l) = -(L(\varphi_0),\varphi_l). \tag{26}$$

Подстановка величин a_k, определяемых путем решения уравнения (30) в формулу (22) дает искомое приближенное решение z_a.

Вообще говоря, вычислить абсолютную погрешность полученного приближения мы не сможем, ведь в общем случае точное решение задачи нам неизвестно (и порой оно может иметь весьма устрашающий вид), однако в качестве оценки погрешносит мы можем использовать *норму невязки*. Скажем, пусть невязка $R_n(x,y)$ зависит от выбранного числа членов разложения n. Тогда вполне естественно ожидать, что если наше пробное решение $z_a(x,y)$ приближает искомое решение, то

$$R_n(x,y) \underset{n\to\infty}{\longrightarrow} 0, \quad \|R_n\| \underset{n\to\infty}{\longrightarrow} 0, \tag{27}$$

ведь если z_a в точности равно искомому решению, то $R(x,y) = 0$ по определению невязки. На самом деле норма невязки *сильно больше* реальной абсолютной погрешности. В работе Финлейсона [9] используется норма невязки для оценки абсолютной погрешности. Таким образом, норма невязки с лихвой оценивает абсолютную погрешность сверху, и этим можно пользоваться для исследования. В качестве нормы можно брать стандартную норму, вытекающую из введенного скалярного произведения: $\|R\| = \sqrt{(R,R)}$.

3.1 Линейное ДУ первого порядка

В этом разделе мы сформулируем общий алгоритм решения краевой задачи для дифференциального уравнения первого порядка с переменными коэффициентами:

$$\begin{cases} y' + \alpha(x)y + \beta(x) = 0 \\ y(0) = d \end{cases}. \tag{28}$$

Также договоримся, что приближенное решение будем искать в области $x \in [0,1]$.

Представим решение в следующем виде

$$y_a(x) = \varphi_0(x) + \sum_{k=1}^{n} a_k \varphi_k(x). \tag{29}$$

В методе Галёркина полагается, что функция φ_0 должна удовлетворять граничному условию, то есть (в нашем случае) $\varphi_0(0) = d$. Тем временем остальные

функции φ_k, $k = \overline{1,n}$ должны удовлетворять *однородным* граничным условиям, то есть $\varphi_k(0) = 0$. Чтобы удовлетворить этим требованием, выберем полную систему функций φ_k так:

$$\varphi_0 = d, \quad \varphi_k = x^k, k = \overline{1,n}.$$

Тем самым, пробное решение представится в следующем виде:

$$y_a(x) = d + \sum_{k=1}^{n} a_k x^k. \tag{30}$$

Видно, что пробное решение (30) удовлетворяет граничному условию (это необходимое требование в методе Галёркина). Следующим шагом нужно подставить пробное решение (30) в уравнение (31) для получения невязки R. Для этого сначала вычислим производную пробного решения:

$$y'_a = \sum_{k=1}^{n} k a_k x^{k-1}. \tag{31}$$

А теперь подставляем пробное решение в изначальное и получаем невязку:

$$R(x) = \sum_{k=1}^{n} k a_k x^{k-1} + \alpha(x)\left(d + \sum_{k=1}^{n} a_k x^k\right) + \beta(x) = \sum_{k=1}^{n} a_k \left(k x^{k-1} + \alpha(x) x^k\right) + \alpha(x) d + \beta(x). \tag{32}$$

Так как мы договорились искать решение на отрезке $[0,1]$, удобно будет выбрать следующее скалярное произведение:

$$(f,g) = \int_0^1 g(x) f(x) dx. \tag{33}$$

В уравнениях (25) для обеспечения условий ортогональности невязка скалярно умножалась на те же пробные функции φ_l. Вообще говоря, это не обязательное условие. Функции, на которые умножается невязка называются *весовыми*, и могут отличаться от пробных. Метод Галёркина является представителем класса мето-

дов, носящих название *методов взвешенных невязок* (очень характерное название, учитывая уравнение (25)), в каждом из которых предъявляются свои требования к выбору весовых функций. В книге Флетчера [8] проделано подробное сравнение различных методов этого класса. В методе Галёркина весовые функции выбираются из числа тех же функций, что и пробные функции φ_l, однако мы не обязаны точь-в-точь повторять последовательность этих функций. И в данном примере вместо использования функций x^l в качестве весовых удобнее взять x^{l-1}. Это связано с тем, что выбор в качестве весовых функций членов более высокого порядка приводит к тому, что решения получаются менее точными.

Итак, составляем условия ортогональности:

$$(R,x^{l-1}) = 0$$

$$\left(\sum_{k=1}^{n} a_k \left(kx^{k-1} + \alpha(x)x^k\right) + \alpha(x)d + \beta(x), x^{l-1}\right) = 0$$

$$\sum_{k=1}^{n} a_k \left(kx^{k-1} + \alpha(x)x^k,x^{l-1}\right) + \left(\alpha(x)d + \beta(x),x^{l-1}\right) = 0.$$

В результате получаем систему линейных алгебраических уравнений

$$Ma = b, \tag{34}$$

где $M = (m_{lk})$ — матрица размера $n \times n$, состоящая из элементов $m_{lk} = \left(kx^{k-1} + \alpha(x)x^k,x^{l-1}\right)$, $b = (b_l)$ — столбец высоты n, где $b_l = -\left(\alpha(x)d + \beta(x),x^{l-1}\right)$, а a — столбец неизвестных — тех самых искомых коэффициентов a_k. Иначе составляющие системы уравнений (34) можно записать так:

$$m_{lk} = \int_0^1 \left(kx^{k+l-2} + \alpha(x)x^{k+l-1}\right) dx = \frac{k}{k+l-1} + \int_0^1 \alpha(x)x^{k+l-1}dx$$

$$b_l = -\int_0^1 \left(\alpha(x)d + \beta(x)\right) x^{l-1}dx = -d\int_0^1 \alpha(x)x^{l-1}dx - \int_0^1 \beta(x)x^{l-1}dx.$$

Таким образом, дифференциальное уравнение свелось к решению системы линейных алгебраических уравнений размера $n \times n$. На практике достаточно выбирать n равным от трех до пяти для достижения хорошей точности, а решить систему линейных уравнений размера 5×5 совсем не ресурсозатратно для ЭВМ. Вычисление скалярных произведений (определенных интегралов) также не занимает большого количества времени.

Пример 4. *Рассмотрим простейший пример использования данного алгоритма и решим краевую задачу следующего вида:*

$$\begin{cases} y' - y = 0 \\ y(0) = 1. \end{cases} \tag{35}$$

Легко видеть, что функция $y = e^x$ является решением данной задачи. Теперь воспользуемся методом Галёркина.

Представляем пробное решение в виде

$$y_a = 1 + \sum_{k=1}^{n} a_k x^k, \quad y' = \sum_{k=1}^{n} k a_k x^{k-1} \tag{36}$$

(и заодно сразу вычислим его производную). Число членов разложения n пока выбирать не будем. Вычисляем невязку, подставляя пробное решение (36) в уравнение (35):

$$R(x) = \sum_{k=1}^{n} k a_k x^{k-1} - 1 - \sum_{k=1}^{n} a_k x^k = \sum_{k=1}^{n} a_k \left(k x^{k-1} - x^k \right) - 1. \tag{37}$$

И накладываем условия ортогональности:

$$\begin{aligned} \left(R,x^{l-1}\right) &= \left(\sum_{k=1}^{n} a_k \left(k x^{k-1} - x^k \right) - 1, x^{l-1} \right) = \\ &= \sum_{k=1}^{n} a_k \left(k x^{k-1} - x^k, x^{l-1} \right) - \left(1, x^{l-1}\right) = 0 \end{aligned} \tag{38}$$

Итого для системы линейных уравнений $Ma = b$ получаем

$$m_{lk} = \left(kx^{k-1} - x^k, x^{l-1}\right) = \int\limits_0^1 \left(kx^{k+l-2} - x^{k+l-1}\right) dx = \frac{k}{k+l-1} - \frac{1}{k+l}$$

$$b_l = \left(1, x^{l-1}\right) = \int\limits_0^1 x^{l-1} dx = \frac{1}{l}.$$

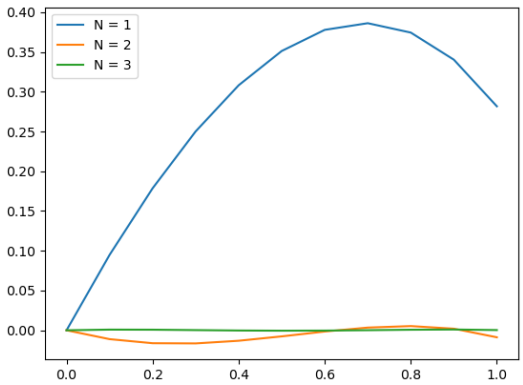
Остается решить полученную систему уравнений (вычислить коэффициенты a_k), подставить решение в разложение (36) и получить искомое приближение.

В таблице 1 можно увидеть, что точность решения быстро возрастает вместе с увеличением числа n.

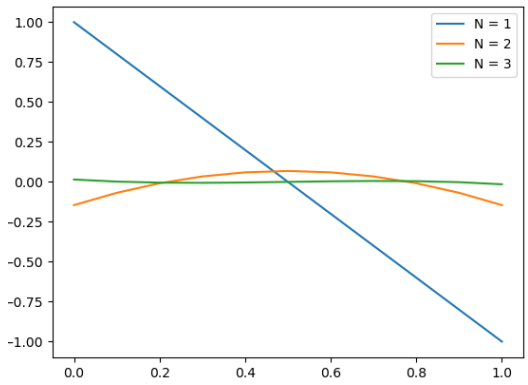
Таблица 1 — Решение уравнения $y' - y = 0$ методом Галёркина с n членами разложения.

x	$n = 1$	$n = 2$	$n = 3$	Точное значение
0.0	1.0	1.0	1.0	1.0
0.1	1.2	1.09402	1.105994	1.105171
0.2	1.4	1.205134	1.222121	1.221403
0.3	1.6	1.333343	1.350068	1.349859
0.4	1.8	1.478646	1.491526	1.491825
0.5	2.0	1.641044	1.648182	1.648721
0.6	2.2	1.820536	1.821725	1.822119
0.7	2.4	2.017123	2.013844	2.013753
0.8	2.6	2.230804	2.226228	2.225541
0.9	2.8	2.46158	2.460566	2.459603
1.0	3.0	2.70945	2.718545	2.718282
Погрешность $\|y - y_a\|$	0.300805	0.009756	0.000575	
Норма невязки $\|R\|$	0.560117	0.06057	0.004956	

Здесь выявляется очень важная особенность метода Галёркина, состоящая в том, что высокая степень точности может быть достигнута за счет весьма умеренных алгебраических усилий. Также по таблице 1 можно видеть, насколько быстро уменьшается погрешность при увеличения числа n, а вместе с ней и норма невязки. На рисунке 1 показаны абсолютная погрешность решения и функция невязки при различных числах n.



а) Распределение погрешности.



б) Распределение невязки.

Рисунок 1 — Распределение погрешности и невязки для решения уравнения $y' - y = 0$ с параметром n.

3.2 Линейное ДУ второго порядка

Сейчас мы сформулируем общий алгоритм решения краевой задачи для линейного дифференциального уравнения второго порядка с переменными коэффициентами на основе метода Галеркина. Задачу поставим в следующем, довольно общем виде:

$$\begin{cases} y'' + \alpha(x)y' + \beta(x)y = \gamma(x) \\ y(0) = y_0 \\ y(1) = y_1. \end{cases} \tag{39}$$

Опять же, приближенное решение будем искать в области $x \in [0,1]$. Здесь это уже более естественно, учитывая граничные условия.

Снова представим приближенное решение в виде (29), однако теперь для удовлетворения граничным условиям, выберем следующую полную систему функций:

$$\varphi_0(x) = (y_1 - y_0)x + y_0, \quad \varphi_k = x^k(1-x) = x^k - x^{k+1}, \, k = \overline{1,n}. \tag{40}$$

Таким образом, пробное решение представится в виде

$$y_a = (y_1 - y_0)x + y_0 + \sum_{k=1}^{n} a_k \left(x^k - x^{k+1}\right). \tag{41}$$

Вычислим первую и вторую производные:

$$y'_a = y_1 - y_0 + \sum_{k=1}^{n} a_k \left(kx^{k-1} - (k+1)x^k\right), \tag{42}$$

$$y''_a = \sum_{k=1}^{n} a_k \left(k(k-1)x^{k-2} - (k+1)kx^{k-1}\right). \tag{43}$$

Подставляем все в уравнение (39) и вычисляем невязку:

$$\begin{aligned} R(x) = \sum_{k=1}^{n} a_k \left(k(k-1)x^{k-2} - (k+1)kx^{k-1}\right) + \\ + \alpha(x)\left(y_1 - y_0 + \sum_{k=1}^{n} a_k \left(kx^{k-1} - (k+1)x^k\right)\right) + \\ + \beta(x)\left((y_1 - y_0)x + y_0 + \sum_{k=1}^{n} a_k \left(x^k - x^{k+1}\right)\right) - \gamma(x) = \\ = \sum_{k=1}^{n} a_k \Big(k(k-1)x^{k-2} + (\alpha(x) - k - 1)kx^{k-1} + (\beta(x) - \alpha(x)(k+1))x^k - \\ - \beta(x)x^{k+1}\Big) + \alpha(x)(y_1 - y_0) + \beta(x)(y_1 - y_0)x - \gamma(x). \end{aligned} \tag{44}$$

Условия ортогональности (в качестве весовых функций будем брать те же функции φ_l, что и в полной системе функций):

$$\begin{aligned} (R,x^l - x^{l-1}) = \sum_{k=1}^{n} a_k \Big(\big(k(k-1)x^{k-2} + (\alpha(x) - k - 1)kx^{k-1} + \\ + (\beta(x) - \alpha(x)(k+1))x^k - \beta(x)x^{k+1},x^l - x^{l-1}\big)\Big) - \\ - \big(\alpha(x)(y_1 - y_0) + \beta(x)(y_1 - y_0)x - \gamma(x),x^l - x^{l-1}\big) = 0. \end{aligned} \tag{45}$$

Итого получаем систему линейных алгебраических уравнений $Ma = b$, где

$$\begin{aligned} m_{lk} = \big(k(k-1)x^{k-2} + (\alpha(x) - k - 1)kx^{k-1} + \\ + (\beta(x) - \alpha(x)(k+1))x^k - \beta(x)x^{k+1},x^l - x^{l-1}\big), \end{aligned}$$

$$b_l = \big(\alpha(x)(y_1 - y_0) + \beta(x)(y_1 - y_0)x - \gamma(x),x^l - x^{l-1}\big).$$

Таким образом, мы снова получили систему линейных уравнений размера $n \times n$, которую несложно вычислить, учитывая тот факт, что для достижения достаточной точности чаще всего достаточно брать от трех до пяти членов разложения.

Именно этот алгоритм реализован в интерактивном приложении, разработанном в качестве поддержки исследования данной дипломной работы. Оно предоставляет удобный интерфейс для исследования различных краевых задач вида (39), а также допускает возможность сохранения и загрузки результатов. Полное описание приложения можно найти в разделе 4.

Пример 5. *Для примера решим следующую краевую задачу:*

$$\begin{cases} y'' - 2y' + xy = 2x \\ y(0) = 1 \\ y(1) = 2. \end{cases} \tag{46}$$

Пробное решение имеет вид:

$$y_a(x) = x + 1 + \sum_{k=1}^{n} a_k x^k (1 - x^k). \tag{47}$$

Его производные:

$$y_a'(x) = 1 + \sum_{k=1}^{n} a_k \left(k x^{k-1} - (k+1) x^k \right),$$
$$y_a''(x) = \sum_{k=1}^{n} a_k \left(k(k-1) x^{k-2} - (k+1) k x^{k-1} \right).$$

Невязка (сразу воспользуемся формулой (44)):

$$R(x) = \sum_{k=1}^{n} a_k \Big(k(k-1)x^{k-2} + (-k-3)kx^{k-1} + \\ + (x + 2(k+1))x^k - x^{k+2} \Big) + 2 + x^2 - 2x. \tag{48}$$

Остается наложить условия ортогональности:

$$\Big(R,x^l(1-x)\Big) = \Big(\sum_{k=1}^{n} a_k\big(k(k-1)x^{k-2} + (-k-3)kx^{k-1} + \\ + (x+2(k+1))x^k - x^{k+2}\big) + 2 + x^2 - 2x, x^l - x^{l+1}\Big) = \\ = \sum_{k=1}^{n} a_k\Big(k(k-1)x^{k-2} + (-k-3)kx^{k-1} + \\ + (x+2(k+1))x^k - x^{k+2}, x^l - x^{l+1}\Big) + \Big(2 + x^2 - 2x, x^l - x^{l+1}\Big) = 0. \quad (49)$$

и получить соответствующую систему уравнений $Ma = b$, где

$$m_{lk} = \Big(k(k-1)x^{k-2} + (-k-3)kx^{k-1} + \\ + (x+2(k+1))x^k - x^{k+2}, x^l - x^{l+1}\Big) \quad (50)$$

$$b_l = -\Big(2 + x^2 - 2x, x^l - x^{l+1}\Big). \quad (51)$$

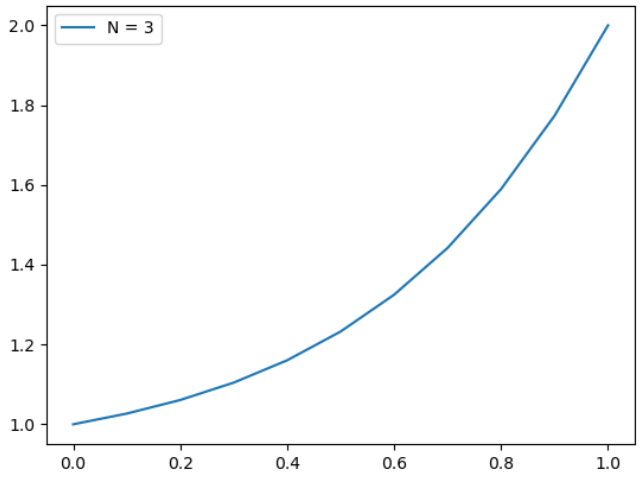


Рисунок 2 — График приближенного решения уравнения $y'' - 2y' + xy = 2x$ методом Галёркина.

График приближенного решения при $n = 3$ показан на рисунке 2. Норма невязки равна $\|R\| = 0.012149$.

4 Описание приложения

4.1 Требования для запуска

Приложение полностью написано на высокоуровневом языке Python 3.8 и тестировалось на персональном компьютере Lenovo ideapad 310-15isk с установленной операционной системой Windows 10. Последнюю версию исходного кода для приложения можно скачать по ссылке на Github [10].

Для запуска приложения на компьютере должен быть установлен интерпретатор языка Python версии не ниже 3.5. Например, последнюю версию можно скачать по ссылке [11].

Также должны быть установлены следующие библиотеки, используемые в приложении:

— math (набор математических функций и констант, встроен в интерпретатор по умолчанию)

— numpy (для общих математических и числовых операций, обычно встроен в интерпретатор по умолчанию)

— scipy (библиотека фундаментальных математических алгоритмов)

— matplotlib (для построения графиков)

— PyQt5 (реализует фреймворк для разработки кроссплатформенного программного обеспечения Qt)

Для установки недостающих библиотек нужно зайти в командной строке Windows (или Linux) в папку с установленным Python, зайти в директорию Scripts и выполнить (по очереди, но в любом порядке) команды из листинга 1. Если переменная pip3 добавлена в настройках среды в PATH, то эту команду

Листинг 1 — Установка библиотеки matplotlib

```
1 pip3 install numpy
2 pip3 install matplotlib
3 pip3 install scipy
4 pip3 install PyQt5
```

можно выполнить из любого места на компьютере.

4.2 Структура и запуск

Исходный код приложения состоит из следующих шести файлов и двух директорий:

— `app1.py` (основной файл запуска)
— `function.py` (вспомогательные финтифлюшки для работы с функциями)
— `SODE.py` (реализация метода Галёркина)
— `Main.py` (главное окно приложения)
— `SODEW.py` (основное рабочее окно приложения)
— `helper.py` (вспомогательные структуры данных)
— `miscellaneous` (директория для хранения вспомогательных файлов, например изображений)
— `saves` (директория для хранения сохранененных результатов)

Перед запуском приложения следует убедиться, что все эти восемь файлов присутствуют в директории приложения. Отсутствие какого-либо из них приведет к ошибкам при запуске приложения или работе с ним. Из директории `miscellaneous` также нельзя удалять никакие файлы, в то время как из директории `saves` — можно (однако саму эту папку удалять нельзя!). Добавление в директорию `saves` сторонних файлов извне приложения также может привести к ошибкам при работе с приложением.

Есть два варианта запуска приложения. Первый заключается в том, чтобы зайти в интегрированную среду разработки и обучения на языке `Python` — `IDLE`, открыть в ней файл `app1.py` и запустить его. Должно открыться главное окно приложения (рис. 3). Второй способ — из командной строки `Windows` (или `Linux`). Для этого нужно в терминале зайти в файл с приложением и прописать в командной строке одну из двух строчек из листинга 2. Какую именно строчку прописывать, зависит от того, какая из двух переменных (`python` или `python3`) добавлена в `PATH`, если не сработает одна из них, нужно попробовать другую. После этого также должно открыться главное окно приложения (рис. 3).

Листинг 2: Запуск приложения из командной строки

```
1 python app1.py
2 python3 app1.py
```

Рисунок 3 — Главное окно.

4.3 Инструкция по использованию

После того как мы открыли главное окно приложения (рис. 3), следует нажать на кнопку `Start` для запуска основного рабочего окна (рис. 4). Кнопка `Exit` может быть нажата в любой момент работы с приложением для полного закрытия всех окон и завершения работы приложения.

Опишу теперь по очереди компоненты рабочего окна приложения. На рис. 5 можно видеть компоненту, отображающую текущий вид краевой задачи для линейного дифференциального уравнения второго порядка. По умолчанию она имеет простенький вид:

$$\begin{cases} y'' = 0 \\ y(0) = 0 \\ y(1) = 1 \end{cases} .$$

Кнопка `SOLVE` решает данную краевую задачу и отображает результаты в правой части рабочего окна приложения (подробнее ниже).

Следующая компонента (рис. 13) позволяет задавать треуемые коэффициенты $a(x)$, $b(x)$, $c(x)$ дифференциального уравнения и соответственно значения на границе y_0, y_1. В выпадающем списке (рис. 7 а)) можно выбрать одну из восьми

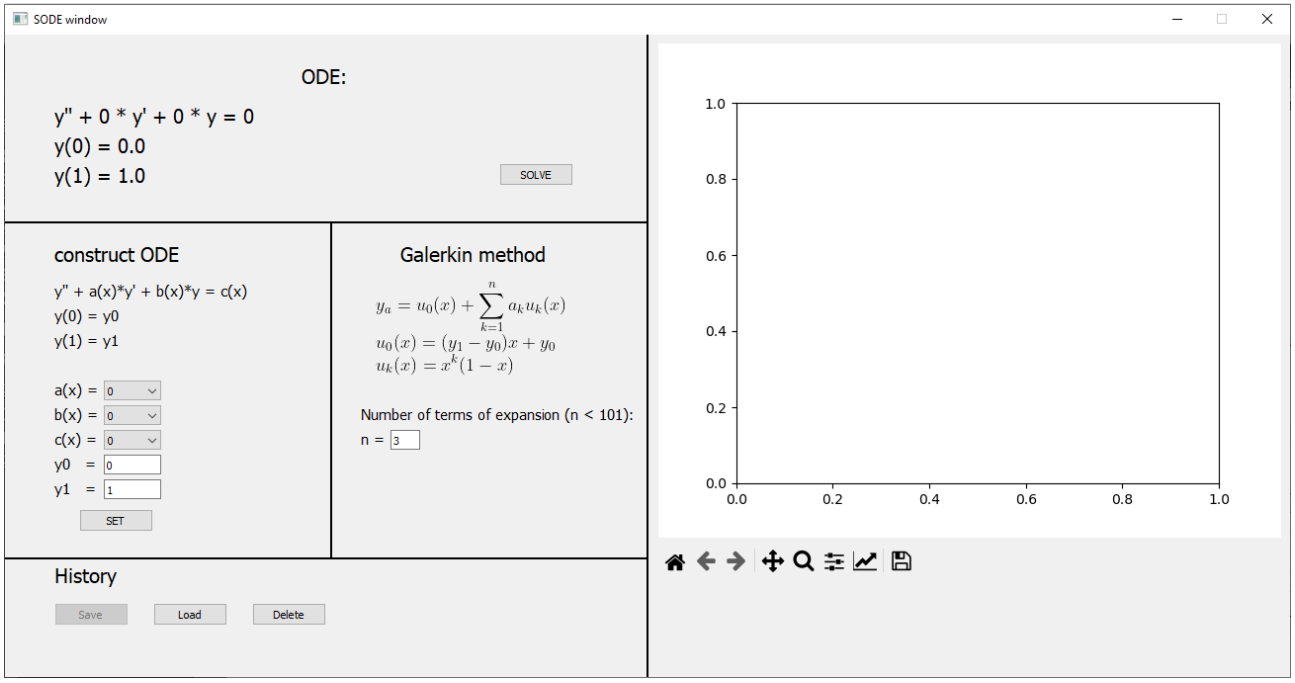


Рисунок 4 — Рабочее окно.

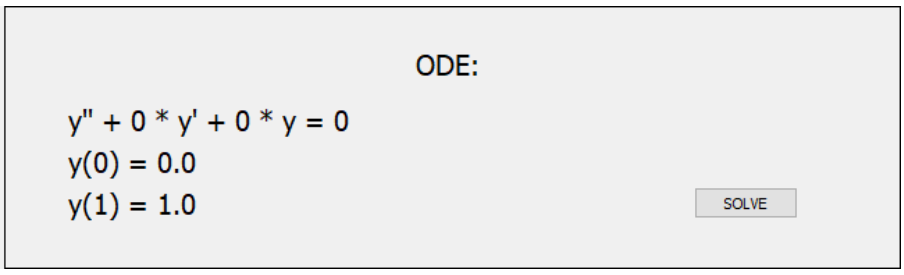


Рисунок 5 — Компонента ODE.

допустимых функций, пять из которых имею параметр a:

— `0` (тождественно равная нулю функция)

— `1` (тождественно равная единице функция)

— `x` (тождественная функция $f(x) = x$)

— `const` (константная функция, принимающая всюду значение, равное a)

— `pow` (степенная функция, параметром обозначается степень $f(x) = x^a$)

— `sin(x)` ($f(x) = \sin(ax)$)

— `cos(x)` ($f(x) = \cos(ax)$)

— `exp(x)` ($f(x) = \exp(ax) = e^{ax}$)

Для задания параметра a нужно ввести его в появляющейся строке справа от названия функции (рис. 7 б)). Также можно ввести желаемые значения функции на границе отрезка $[0,1]$. После введения коэффициентов и граничных условий

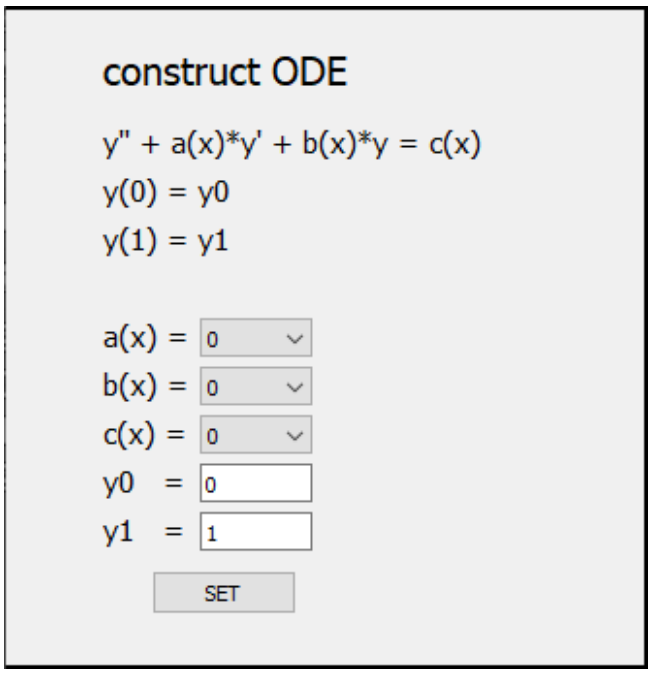
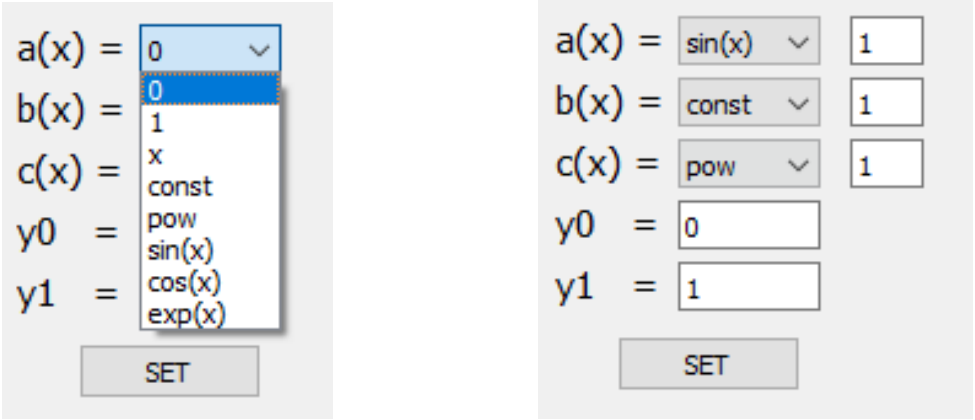


Рисунок 6 — Компонента construct ODE.



а) Список допустимых функций. б) Параметры функций.

Рисунок 7 — Задание коэффициентов.

нужно нажать на кнопку SET, после чего краевая задача из компоненты ODE (рис. 5) примет требуемый вид. При введении параметров функции и значений на границах следует убедиться в том, что введены именно числовые значения, а также что введенные значения не слишком длинны. При несоблюдении этих условий после нажатия на кнопку SET будут выведены соответствующие сообщения об ошибках (рис. 8).

В следующей компоненте (рис. 9) описывается, какая система функций выбрана для приближения решения поставленной задачи методом Галёркина. Также

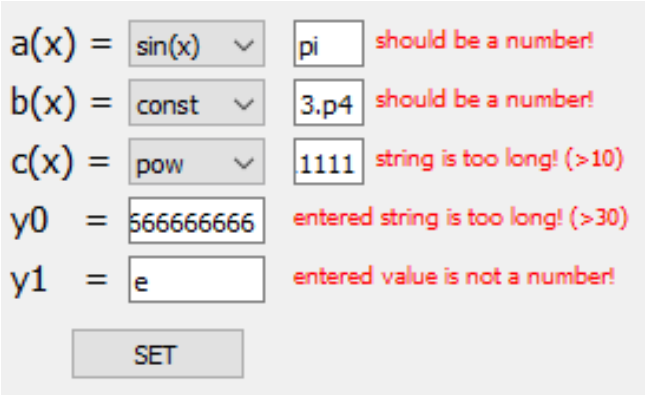


Рисунок 8 — Примеры ошибок введенных значений.

предлагается выбрать число членов разложения n. Чем больше членов разложения

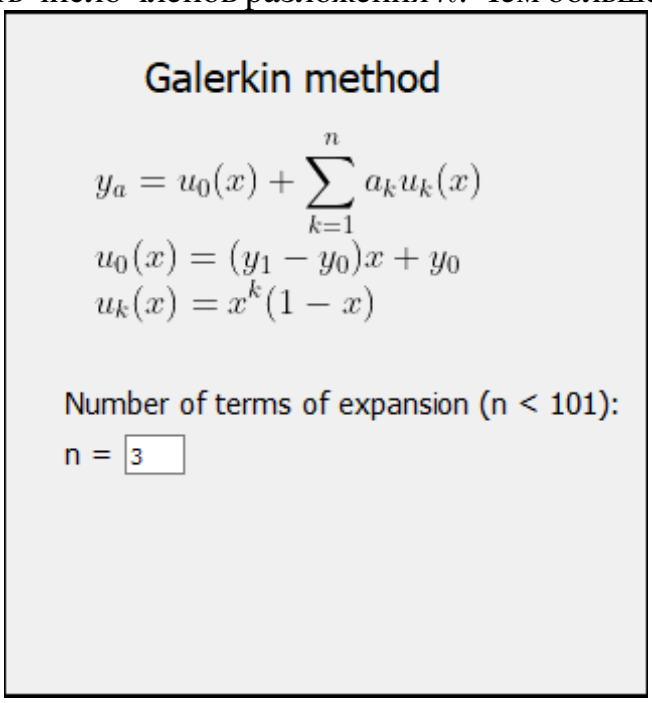


Рисунок 9 — Компонента Galerkin method.

будет выбрано, тем точнее будет результат. Реализация допускает выбирать от 1 до 100 членов разложения, однако на практике достаточно не более 5 членов для получения достаточно точного результата. Выбирать более 20 членов вообще не рекомендуется, так как в таком случае вычисление может производиться долго.

В правой части окна (рис. 10) выводятся результаты решения краевой задачи. Они состоят из графика приближенной функции на отрезке $[0,1]$ и места под оценку погрешности (в самом низу компоненты). В качестве оценки выводится норма невязки (см. раздел **??**). График строится с помощью библиотеки `matplotlib`, и к нему приложен стандартный набор инструментов этой библиотеки для работы с

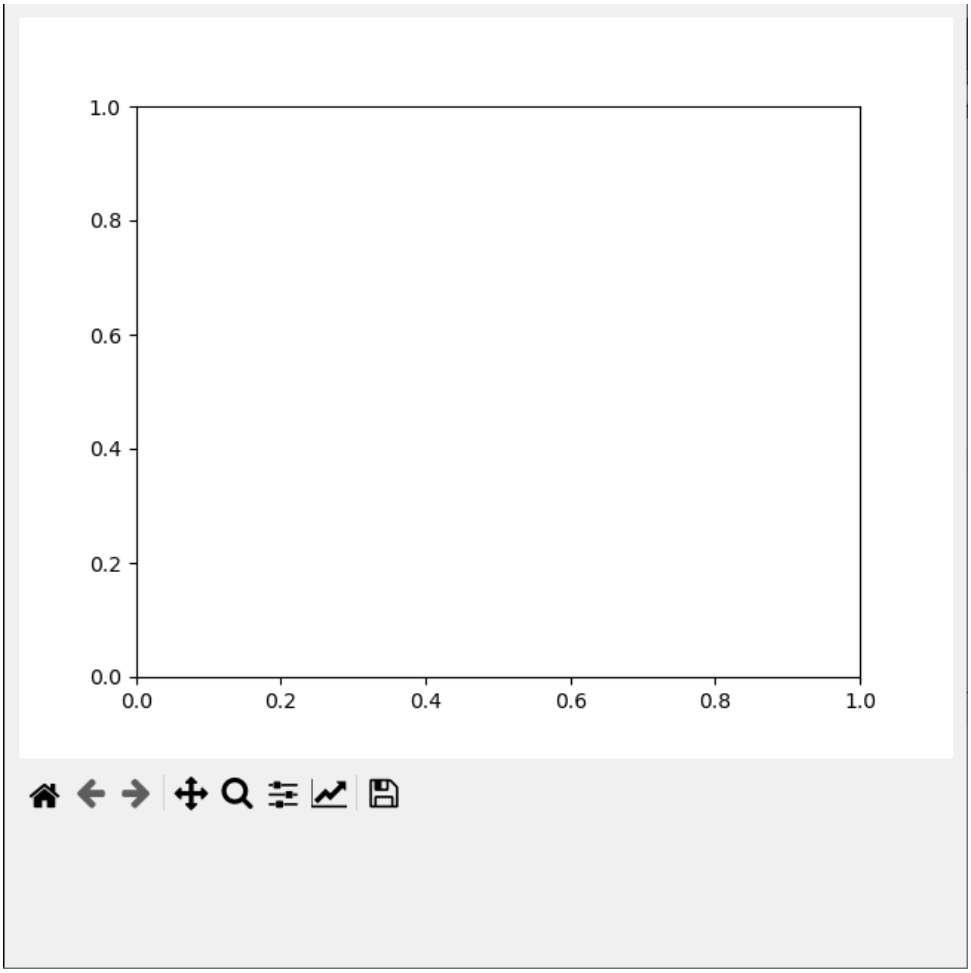


Рисунок 10 — Компонента вывода результатов.

графиком. Поэтому можно свободно увеличивать и двигать график, менять масштабы осей и другие действия, привычные при работе с графиками, построенными с помощью данной библиотеки.

В последней компоненте (рис. 11) представлен функционал для сохранения, загрузки и удаления сохраненных результатов. По умолчанию кнопка `Add` является

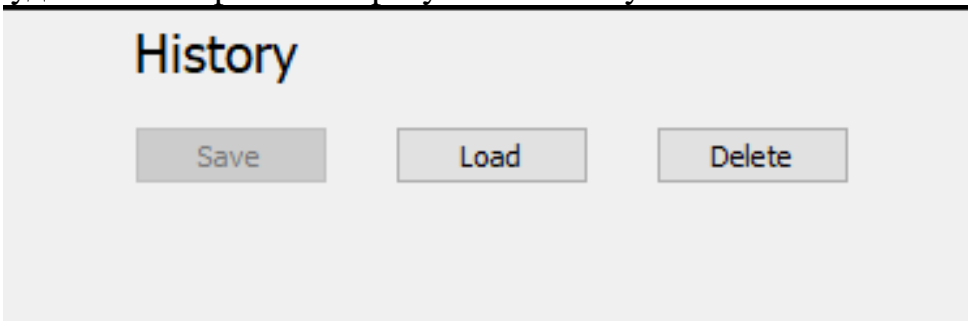


Рисунок 11 — Компонента History.

недоступной. Она разблокируется как только будет выполнено вычисление какой-либо задачи. При ее нажатии открывается окно (рис. 12), предлагающее ввести

название фигуры в текстовое поле и сохранить текущий результат. После этого в том же окне появится сообщение об успешном сохранении, а введенное название отобразится над графиком полученной функции.

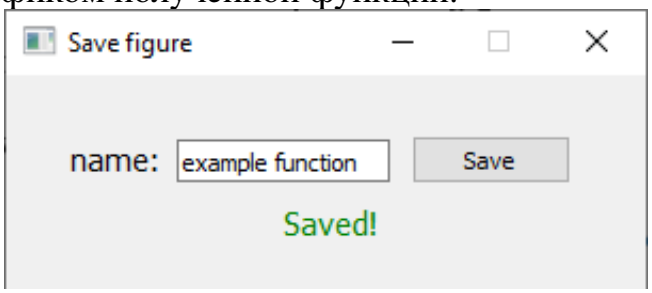


Рисунок 12 — Окно сохранения.

Кнопки `Load` и `Delete` открывают окна, в которых можно загрузить на экран сохраненный результат или же удалить его (рис. 14). Для этого нужно выбрать нужную фигуру из списка и нажать на кнопки `Load` или `Delete` соответственно. Загрузка и удаление также сопровождаются сообщениями об успехе. Причем при

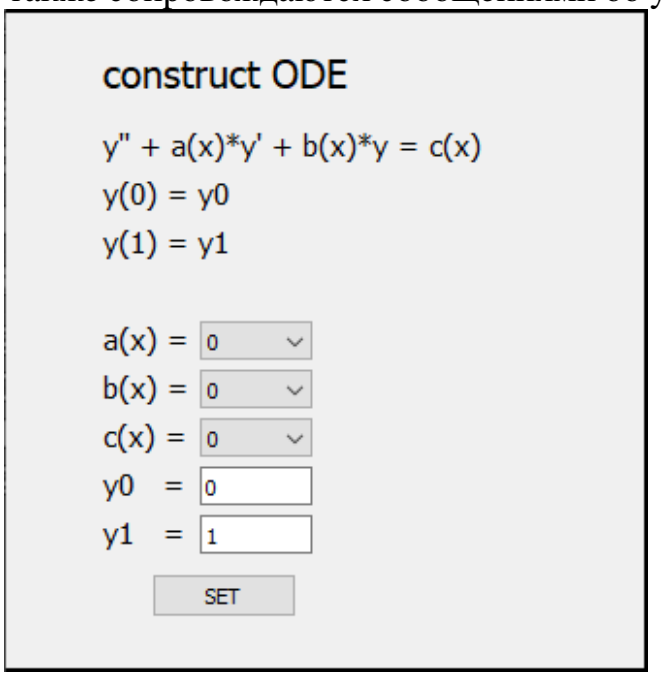
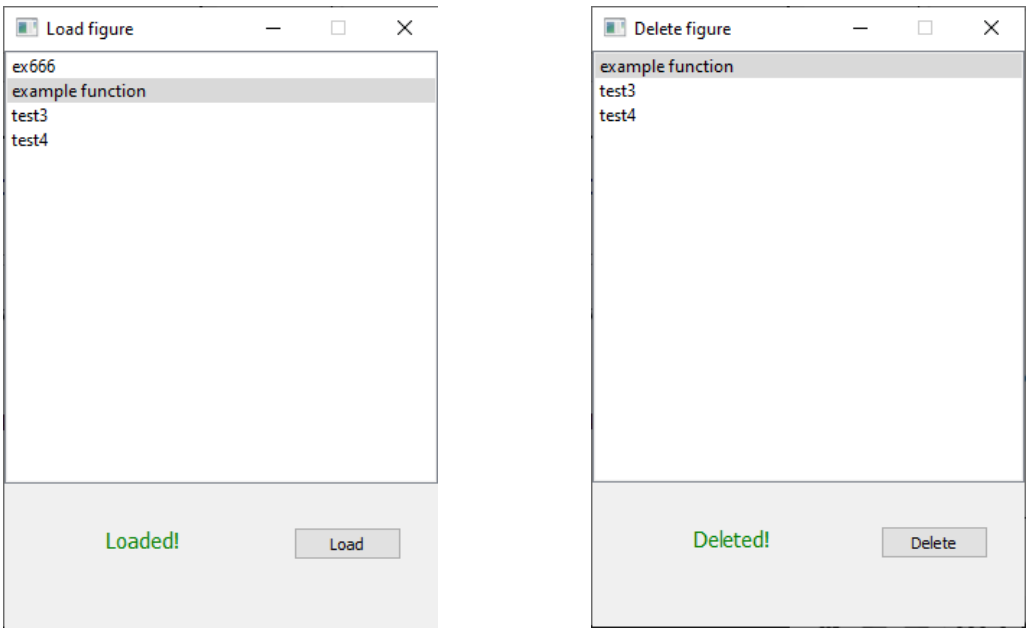


Рисунок 13 — Компонента construct ODE.

загрузке фигуры помимо вывода графика и оценки погрешности также приводится к соответствующему виду краевая задача и все заданные параметры. Стоит отметить, что информация о графике также сохраняется, поэтому никаких повторных вычислений при загрузке не производится.

В итоге при успешном введении данных и решении задачи мы получаем нужный нам результат (см. рис 15).



а) Окно загрузки. б) Окно удаления.

Рисунок 14 — Окна загрузки и удаления.

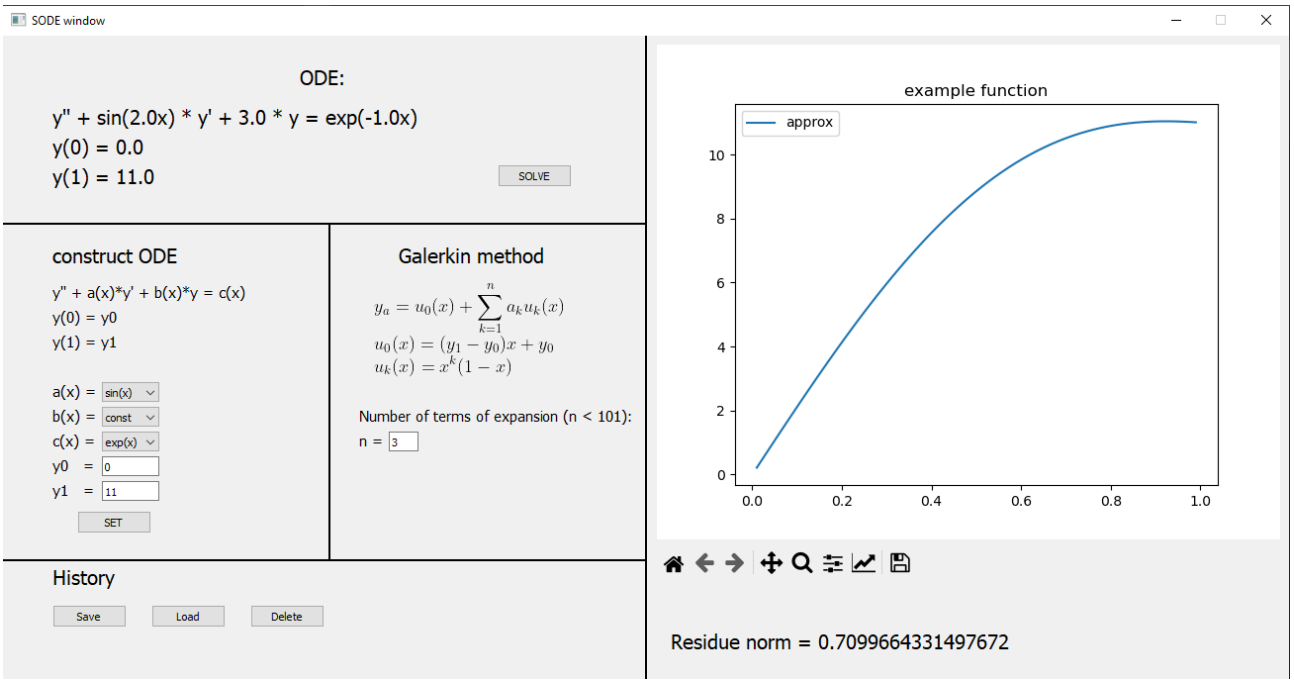


Рисунок 15 — Результат работы программы.

4.4 Неожиданные результаты

Дифференциальные уравнения — вообще говоря довольно сложный объект. Решать их в общем виде до сих пор никто не умеет. И даже в таких условно про-

стеньких примерах, как линейное дифференциальное уравнение второго порядка с переменными коэффициентами, решение может быть безуспешным. Традиционный метод Галеркина не лишен таких проблем, поэтому некторые виды дифференциальных уравнений могут быть не решены. Особенно этим грешит использование степенных функций (в особенности отрицательных и дробных степеней). Например, попытка решить дифференциальное уравнение $y'' + \frac{y'}{x} + x^{-2.3}y = x^5$ методом, заложенным в приложении, не дает точного результата (см. рис. 16). Видно, что даже при выборе 50 членов норма невязки слишком высока. Тем не менее, было ре-

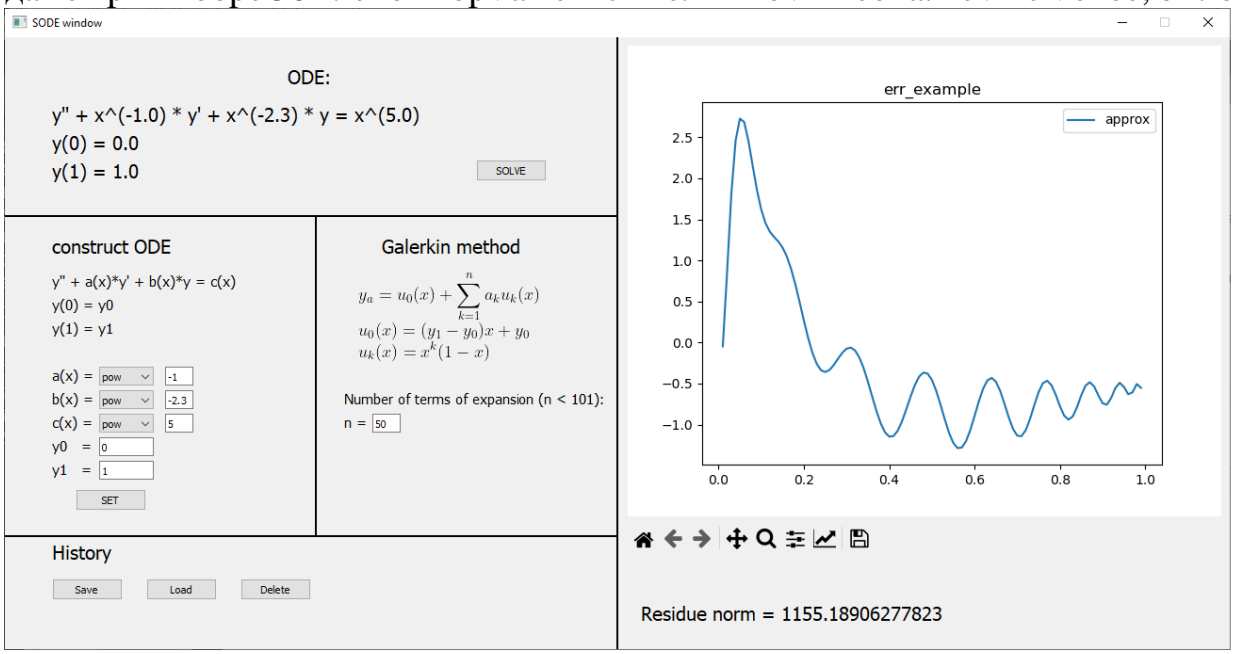


Рисунок 16 — Расхождение вычисления.

шено оставить степенную функцию в приложении, потому что для исследователя полезно понимать, какие уравнения решить данным методом не удается.

4.5 Некоторые детали реализации

Для реализации математических программ часто приходится работать с функциями. Хочется уметь легко их складывать, умножать, брать композицию, линейные комбинации и многое другое, иными словами, работать с ними как с обычными объектами. Такой весьма удобный синтаксис для работы с функциями присущ функциональным языкам, особенно синтаксис языка `Haskell` позволяет проделывать все эти действия без какого-либо труда и с минимальными затратами

Листинг 1: Простая комбинация функций на Python

```
f = lambda x: h(x) + g(x) * w(x)
```

Листинг 2: Интеграл с переменным верхним пределом на Python

```
from scipy.integrate import quad
from math import exp
F = lambda x: quad(lambda y: exp(-y**2/2), 0, x)
```

символов в коде. Оно и понятно, ведь в большинстве функциональных языков единственными объектами и выступают сами функции!

Однако язык Python лишен таких привелегий. Конечно, функции и тут являются самостоятельными объектами, но над ними не определены совершенно никакие арифметические операции, и для того чтобы создать функцию f, ведущую себя как-то так: $f(x) = h(x) + g(x) * w(x)$, приходится писать что-то вроде такого (см. листинг 1), или, например, для интеграла с переменным верхним пределом $F(x) = \int\limits_0^x e^{-\frac{\xi^2}{2}} d\xi$ код выглядел бы весьма неуклюже (см. 2). А для более сложных комбинаций функций число вложенных «лямбд» может зашкаливать.

С этой проблемой я столкнулся в процессе исследования метода Галёркина и при реализации приложения. Дабы облегчить задачу, был реализован удобный интерфейс, позволяющий совершать необходимые операции над функциями «без посредников». Иными словами, был реализован вспомогательный класс вещественнозначной функции, носящий название RF (сокращенно от *RealFunction*). Частичное описание данного класса можно найти в приложении А (3).

По сути данный класс представляет из себя некую «обертку» для стандартного объекта функции языка Python. Конструктор класса принимает «обычную» функцию (возможно, заданную через lambda), и возвращает ее же, но уже упакованную в «обертку» RF. Теперь с функцией можно делать много полезного: складывать, умножать, брать композиции, интегрировать и брать линейные комбинации. Разумеется, большинство методов реализованы через «магические методы» языка Python.

Листинг 3: Примеры использования класса RF

```
sin = RF.sin()
cos = RF.cos()
exp = RF.exp()
x2 = 2*RF.id()                  # f(x) = 2*x
g = sin**2 + cos**2             # то же самое, что и RF.const(1)
h = -exp**2/2
F = lambda x: h.integrate(0,x)  # почти нормальное распределение
w = sin @ cos + 2*exp @ x2 + 4  # какая-то чушь
G = F @ w @ F @ w               # боюсь представить, как это выглядит
                                # аналитически
xk = RF(lambda k: RF.pow(k))    # f(x) = x^k
a = [1, 1, 1/2, 1/6, 1/24]
pols = [xk(k) for k in range(5)]
e = RF.linComb(a, pols)         # почти то же самое, что и RF.exp()
```

Метод `call` обеспечивает возможность вызова функции-представителя класса `RF` как совершенно обычной функции. Методы `add`, `mul`, `sub`, `truediv` и `pow` позволяют совершать арифметические операции сложения, умножения, вычитания и деления, и даже не только над двумя функциями, но и над функцией и числом, а также операцию возведения функции в степень действительного числа. Метод `matmul` обеспечивает возможность брать композицию двух функций с использованием оператора `@`, появившегося в языке `Python` в версии 3.5. Метод `integrate` реализует интегрирование функций по отрезку, а метод `linComb` — взятие линейной комбинации набора функций с заданным набором числовых коэффициентов (особенно полезный метод для получения пробного решения в методе Галёркина). Также в класс включены некоторые стандартные функции, полный список можно увидеть в приложении.

Некоторые примеры использования класса `RF` представлены в листинге 3.

Легко видеть, что теперь даже самые устрашающие комбинации функций можно записать читабельным языком и в несколько строчек. Методы интегрирования и линейной комбинации использовались особенно часто. Первая — для вычисления норм и скалярных произведений, вторая — для получения функций приближенного решения (разложения в ряд по полной системе) и вычисления невязок. Также весь этот инструментарий использовался для проверки работоспособности метода Галёркина путем вычисления реальной абсолютной погрешности

решения. Сравнивались они с теми решениями, что выдавал `WolframAlpha`, а у него порой были такие страшные результаты вроде вложенных (иногда и кратное число раз!) интегралов с переменных верхним пределом (разумеется, не без сопровождающего оркестра в виде множественных арифметических операций и операции композиции функций). Иными словами, данный интерфейс значительно облегчил мне жизнь.

Таким образом, данный класс позволяет совершать множество полезных комбинаций над функциями, что сильно облегчило написание кода в процессе исследования алгоритмов метода Галёркина и реализации приложения. Данный класс можно сильно расширить, добавив много других полезных математических конструкций в случае необходимости. Я лишь реализовал те, которые мне были необходимы для работы над темой данного дипломного проекта.

ЗАКЛЮЧЕНИЕ

Метод Галёркина весьма эффективен для решения различных типов, к тому же несложен в реализации. Главная проблема этого метода заключается в том, что сложно разработать какой-то общий алгоритм, покрывающий большой класс задач. В результате исследования дифференциальных уравнений второго порядка выяснилось, что разработанный алгоритм работает не всегда: для некоторых функциональных коэффициентов (особенно со степенными функциями) алгоритм приводит к расходимости. Это вовсе не значит, что метод Галёркина для таких уравнений не годится, просто вероятно нужно подобрать более удобное окружение для его использования. Выбор полной системы функций, выбор скалярного произведения, выбор весовых функций, все это влияет на точность и сходимость. Тем не менее, той или иной формулировкой метода Галёркина можно решить *почти любой* тип задач.

Важной особенностью традиционного метода Галёркина, способствовавшей его широкому распространению, являяется возможность достижения высокой точности при небольшом числе членов в пробном решении. Эта особенность метода усиливает необходимость такого выбора пробных функций, при котором использовалась бы предварительно полученная информация о характере ожидаемого решения.

Напоследок хотелось бы сказать о возможных путях дальнейшего развития данного проекта. Во-первых, это решение вопроса о том, для какого класса дифференциальных уравнений алгоритмы, представленные в разделах 3.1 и 3.2 точно сходятся. Это бы позволило трезво взглянуть на ситуацию и понять, какие модификации требуются для расширения этого класса задач. Во-вторых, это вывод алгоритмов для более сложных видов задач, в частности, заданных дифференциальными уравнениями более высокого порядка, а также дифференциальными уравнениями в частных производных. Наконц, расширение возможностей приложения. В особенности, набор возможных функциональных коэффициентов в нем довольно беден и хотелось бы иметь возможность задавать более сложные комбинации функций. В идеале в этом должна помочь разработка парсера арифметических выражений, позволяющего исследователю вводить коэффициенты с клавиатуры.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Репозиторий с бланком LaTeX для выпускных квалификационных работ. — URL: https://github.com/TonitaN/Blanks-4LaTeX.
2. ПРИЛОЖЕНИЕ 1 к положению «О нормоконтроле, размещении текстов в электронно-библиотечной системе и проверке на объем заимствования выпускных квалификационных работ бакалавров». — URL: https://mf.bmstu.ru/info/uu/ot/norm%5C_docs/docs/polozhenie%5C_normcontrol%5C_pril1.pdf.
3. Городенцев А. Л., лекции по геометрии для студентов НМУ. — URL: http://gorod.bogomolov-lab.ru/ps/stud/geom_ru/1617/lec_total.pdf.
4. Канатников А. Н. Дифференциальные уравнения в частных производных. — Москва : МГТУ им. Н. Э. Баумана, 2021.
5. Воронов А. А. Теория автоматического управления. Ч.I. Теория линейных систем автоматического управления. — Москва : Высш. шк., 1986.
6. Солодовников В. В., Дмитриев А. Н., Егупов Н. Спектральные методы расчета и проектирования систем управления. — Москва : Машиностроение, 1986.
7. Галёркин Б. Г. // Вестник инженеров. — 1915. — № 19. — С. 897—908.
8. Флетчер К. Численные методы на основе метода Галёркина. — Москва : Мир, 1988.
9. Finlayson B. A. The method of weighted residuals and variational principles. — Нью Йорк : Academic Press, 1972.
10. Исходный код на Github. — URL: https://github.com/NotThatWay/Diploma.
11. Официальная страница языка Python. — URL: https://www.python.org/.

ПРИЛОЖЕНИЕ А

Листинг 3: Класс RF для введения арифметических и других операций над функциями.

```
from scipy.integrate import quad
import math as m
class RF:
    def __init__(self, f):
        self.f = f
    def __call__(self, x):
        return self.f(x)
    def __pow__(self, a):
        return RF(lambda x: self.f(x)**a)
    def __add__(self, other):
        if isinstance(other, RF):
            return RF(lambda x: self.f(x) + other.f(x))
        else:
            return RF(lambda x: self.f(x) + other)
    def __sub__(self, other):
        return self + (-other)
    def __mul__(self, other):
    if isinstance(other, RF):
        return RF(lambda x: self.f(x) * other.f(x))
    else:
        return RF(lambda x: self.f(x) * other)
    def __truediv__(self, other):
        return RF(lambda x: self.f(x) / other.f(x))
    def __matmul__(self, other):
       return RF(lambda x: self.f(other.f(x)))
    def integrate(self, a, b):
       return quad(self, a, b)[0]
    def linComb(cs, fs):
       curf = RF.const(0)
       for i in range(len(cs)):
           curf = curf + (cs[i] * fs[i])
       return curf
```