МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «КубГУ»)

Кафедра математических и компьютерных методов

КУРСОВАЯ РАБОТА

РЕШЕНИЕ И ИССЛЕДОВАНИЕ РЕШЕНИЯ ВНУТРЕННЕЙ ЗАДАЧИ ДИРИХЛЕ ДЛЯ УРАВНЕНИЯ ЛАПЛАСА В ОДНОСВЯЗНОЙ ОБЛАСТИ МЕТОДОМ БАЗИСНЫХ ПОТЕНЦИАЛОВ

Работу выполнил_____Д. А. Пасько
(подпись, дата)

Факультет математики и компьютерных наук_____курс 3
Направление 02.03.01 математика и компьютерные науки_____

Научный руководитель
доцент кафедры МиКМ,
канд. физ.-мат. наук_____А. А. Свидлов
(подпись, дата)

Нормоконтролер
преподаватель_____А. А. Цыбенко
(подпись, дата)

Краснодар 2018

СОДЕРЖАНИЕ

Введение ... 4
1 Необходимые сведения ... 6
1.1 Обозначения ... 6
1.2 Определение задачи ... 6
1.3 Задача аппроксимации ... 7
1.4 Суть метода базисных потенциалов ... 9
1.5 Сведение метода базисных потенциалов к комплексу простых задач ... 11
2 Программная реализация метода базисных потенциалов. Описание главных действующих алгоритмов ... 14
2.1 Чтение данных и работа с данными ... 14
2.1.1 Чтение с текстового файла ... 15
2.1.2 Генерация массива точек программным образом ... 16
2.2 Поиск решения ... 16
2.2.1 Нахождение криволинейного интеграла первого рода ... 17
2.2.2 Нахождение решений СЛАУ с симметрической матрицей ... 18
2.2.3 Вывод точности ... 19
2.3 Иллюстрация решения ... 20
2.4 Временная сложность алгоритма ... 21
3 Входные данные программы ... 22
3.1 Тестовые области ... 22
3.1.1 Параметризация границы круга ... 22
3.1.2 Параметризация треугольника ... 23
3.1.3 Параметризация границы квадрата ... 24
3.1.4 Параметризация «острия» ... 25
3.2 Тестовые граничные функции ... 25
3.3 Описание наиболее важных используемых классов и структур, глобальных переменных, некоторых функций и процедур ... 27
4 Исследование устойчивости приближённого решения и качества аппроксимации ... 31
4.1 Эффективность метода базисных потенциалов ... 31
4.2 Поиск метода решения СЛАУ, при котором приближённое решение покажет наилучшую устойчивость ... 38

4.2.1 Неустойчивость метода Гаусса 39
4.2.2 Неустойчивость метода наискорейшего спуска 42
4.2.3 Неустойчивость метода Холецкого 45
4.2.4 Неустойчивость гибридного метода 46
4.2.5 Попытка использования метода покоординатного спуска 51
4.2.6 Устойчивый метод 54
4.3 Поиск расположения базисных точек, обеспечивающего наилучшую аппроксимацию 57
4.3.1 Описание контуров, вблизи которых берутся базисные точки 57
4.3.2 Зависимость качества аппроксимации от кривых, вблизи которых берутся базисные точки 59
Заключение 63
Список использованных источников 64
Ссылки 65

Введение

Большое количество явлений современного мира люди уже умеют описывать математическими моделями, при этом оказывается, что немалая часть таких явлений сводится к краевым задачам математической физики всего нескольких категорий. За всяким физическим процессом стоит некоторая зависимость (функция), которую для каких-то целей требуется выразить с помощью уже известных функций.

Многие математические модели реальных физических процессов представляют из себя краевые задачи, в которых используются гармонические операторы; наиболее часто это верно для задач гидродинамики и квантовой физики, гармонические операторы присутствуют в уравнениях Лапласа, Пуассона, Гельмгольца, Шредингера, в уравнениях, описывающих стационарные процессы, включая установившееся распределение тепла, силу тяготения, потоки в устоявшихся течениях жидкости или же силы электрического взаимодействия. Решение таких задач можно строить разными методами, однако наиболее продуктивными из этих методов являются проекционные, приближённо выражающие решение задачи через проекцию на какое-то пространство функций: такие решения удобнее затем использовать в прикладных исследованиях изучаемых процессов, вдобавок они оказываются очень точными.

Относительно недавно (1960-е г., в работах Купрадзе В. Д., Алексидзе М. А. [13, 14]) класс проекционных методов пополнился методом базисных потенциалов, вариантом метода фундаментальных решений для краевых задач уравнения Лапласа, который с того времени постепенно модифицируется и считается, как минимум, вполне хорошим. Некоторыми преимуществами этого метода являются: высокая точность при небольшом числе базисных функций и возможность его приемлемой реализации на ЭВМ даже с сильно ограниченными системными ресурсами. Однако, при стремлении достичь крайне максимальной точности приближения мы сталкиваемся с несколькими важными задачами, которые ещё предстоит решить: во-первых, численный поиск приближения на ЭВМ ст*о*ит нам погрешностей в вычислении, поэтому от

выбора тех или иных методов интегрирования, решения систем и пр. зависит устойчивость нашего решения; далее, при всяких входных данных, отвечающих определённым условиям, мы будем получать так или иначе приближённое решение, однако до сих пор остаются неизвестными условия, при которых такие решения окажутся наилучшими приближениями. Именно эти задачи являются объектом моего исследования.

Постановка задачи. Найти численный метод, придающий наибольшую устойчивость приближённому решению задачи Дирихле для уравнения Лапласа методом базисных потенциалов; определить условия на расстановку базисных точек, при которых фиксированное количество точек обеспечит наилучшую аппроксимацию в сравнении с любыми другими расположениями того же числа точек.

1 Необходимые сведения

1.1 Обозначения

(x, y) – точка на плоскости с такими координатами; $x = (x_1, \dots, x_n)$ – вектор из пространства R^n; $x^m = (x_1, \dots, x_m)$ – m-мерный вектор; $\Delta = \sum_{i=1}^{n} \frac{\partial^2}{\partial^2 x_i}$ – оператор Лапласа; $L(f_1, \dots, f_m) = \sum_{i=1}^{m} \alpha_i f_i$, $\alpha_i \in R$ – множество всевозможных линейных комбинаций элементов $f_1, \dots, f_m$(линейная оболочка); $\varphi = \varphi(x)$ – граничная функция; $D_{c_i} = \frac{\partial}{\partial c_i}$ – частная производная по аргументу c_i; $L_2(\partial Q)$ – пространство функций, определённых на кривой ∂Q и интегрируемых с квадратом в смысле Лебега; $(\alpha_n, \alpha_m) = (\alpha_n, \alpha_m)_{L_2(\partial Q)} = \oint_{\partial Q} \alpha_n(x)\alpha_m(x)dl$ – скалярное произведение функций$(\alpha_n, \alpha_m) \in L_2(\partial Q)$; $Q^+ = R^2 \backslash (Q \cup \partial Q)$;$\varepsilon$ – некоторое очень малое число, большее нуля и меньшее 10^{-5}; а. п. – аналитическое продолжение функции на всю плоскость, используемое для упрощения программирования этой функции.

1.2 Определение задачи

Пусть на некотором множестве Q пространства R^n определена дважды непрерывно дифференцируемая функция $u(x_1, x_2, \dots, x_n)$. В таком случае оператор Лапласа Δ устанавливается действием на эту функцию следующим образом:

$$\Delta u(x_1, \dots, x_n) = \frac{\partial u^2}{\partial^2 x_1} + \dots + \frac{\partial u^2}{\partial^2 x_n} = \sum_{i=1}^{n} \frac{\partial u^2}{\partial^2 x_i}.$$

Тогда задача нахождения ядра такого оператора сводится к решению уравнения Лапласа

$$\Delta u(x_1, \dots, x_n) = 0,$$

которое, как известно, имеет целый класс решений, называемых гармоническими функциями; и чтобы выделить из множества решений какое-то одно, требуется наложить дополнительные условия. Как видно, уравнение Лапласа

относится к уравнениям эллиптического типа и описывает стационарный процесс, а потому для него могут быть определены только граничные условия; при этом задача нахождения решения уравнения Лапласа при граничных условиях первого рода, то есть вида

$$u(x_1, \dots, x_n)|_{x_1,\dots,x_n \epsilon \partial Q} = \varphi(x_1, \dots, x_n),$$

где ∂Q – граница области определения функции, называется задачей Дирихле для уравнения Лапласа.

В нашем случае требуется методом базисных потенциалов найти решение задачи Дирихле

$$\begin{cases} \Delta u|_Q = 0 \\ u|_{\partial Q} = \varphi \end{cases},$$

где множество Q – плоская односвязная область с кусочно-гладкой границей.

1.3 Задача аппроксимации

В общем случае задача аппроксимации состоит в следующем: требуется максимально хорошо приблизить некоторый элемент $x \in \mathrm{X}$ элементами пространства $F \subset \mathrm{X}, x \notin F$. Качество приближения оценивается по тому, насколько малым оказывается расстояние от элемента $x \in \mathrm{X}$ до элемента $u \in F$, наиболее близкого к x; само же это расстояние выражается формулой:

$$\rho(x, F) = \inf_{u \in F} \|x - u\|.$$

Если для какого-то $u_0 \in F$ инфинум достигается, то u_0 называется элементом наилучшего приближения. В нашем случае мы приближаем решение дифференциального уравнения известной системой функций, называемых базисными потенциалами; при этом неизбежно нужно знать ответы на первейшие вопросы теории аппроксимации: 1) найдётся ли на линейной оболочке системы базисных потенциалов элемент наилучшего приближения и 2) будет ли он единственным? На оба вопроса мы имеем положительные ответы; во-пер-

вых, на практике всегда приходится брать конечную систему функций, а потому решение дифференциального уравнения приближается конечномерным пространством, в котором элемент наилучшего приближения однозначно существует (доказательство можно посмотреть в [9] и [10]); во-вторых, при нахождении решения мы работаем с функциями из пространства $L_2(\partial Q)$, которое является строго нормированным, из чего следует единственность элемента наилучшего приближения; более этого, пространство $L_2(\partial Q)$ является гильбертовым, а потому наилучшим приближением элемента $x \in L_2(\partial Q)$ является его проекция на линейную оболочку $L(f_1, \dots, f_m)$ базисных потенциалов.

Однако, при решении реальных физических задач важно не только найти элемент наилучшего приближения в некоторой системе элементов, но и, собственно, выбрать такую систему, которая сможет обеспечивать безграничное возрастание точности при своём расширении. Иными словами, требуется работать с полными системами; система $S \subset L_2(\partial Q)$ является полной, если для любого элемента из $L_2(\partial Q)$ и для всякой наперёд заданной точности найдётся комбинация элементов из S, обеспечивающая данную точность; для практики полнота системы означает, что с ростом числа базисных потенциалов точность всегда будет улучшаться:

$$\left\| f - \sum_{s \in S_n} c_s(f) s \right\| \to 0, n \to \infty, S_1 \subset S_2 \subset \cdots \subset S_n \subset \cdots \subset S$$

Если же система функций не будет полной, мы никакими способами не сможем осуществить с её помощью приближение лучшее, чем некоторое. Именно поэтому в дальнейшем возникает вопрос о критериях полноты системы базисных потенциалов, то есть о том, как следует выбирать точки, чтобы обеспечивать безгранично хорошую аппроксимацию.

1.4 Суть метода базисных потенциалов

Как было сказано, функции, удовлетворяющие уравнению Лапласа, называются гармоническими; решение задачи Дирихле заключается в нахождении гармонической функции, которая на множестве точек границы исходной области совпадает с заданной функцией φ. Метод базисных потенциалов заключается в следующем: 1) в дополнении исходной области фиксируется последовательность точек, 2) каждой точке ставится в соответствие функция, являющаяся фундаментальным решением уравнения Лапласа, то есть гармоническая функция, 3) граничная функция φ аппроксимируется линейной комбинацией соответствующих функций; таким образом мы находим приближённое решение задачи Дирихле, которое удовлетворяет уравнению Лапласа и с некоторой степенью точности отличается от граничной функции, причём точность зависит как от самих выбранных точек, так и от их количества (ведь при вычислениях нельзя работать с бесконечными последовательностями). Теперь наша задача заключается в определении необходимых понятий и в теоретическом обосновании действенности описанного метода.

Ограниченная последовательность точек $z^m, m = 1, 2, \ldots$ будет называться ***базисной***, если она принадлежит области $Q^+ = R^2 \backslash (Q \cup \partial Q)$ и обеспечивает полноту системы функций, которые описываются далее; в общем случае система $z^m, m = 1, 2, \ldots$ должна быть множеством единственности потенциала простого слоя, что выполняется, например, когда точки $z^m, m = 1, 2, \ldots$ образуют границу любого выпуклого многогранника $M \supset Q$ (другие примеры и условия можно прочесть в [11]). В частности также, последовательность является базисной, если она отделена от границы и удовлетворяет условию единственности гармонических функций (то есть любые две гармонические функции, совпадающие на этих точках, совпадают тождественно).

Пусть базисная система точек определена. В таком случае рассмотрим определённую на ∂Q (учитывая, что Q – область в двумерном пространстве) систему функций

$$\alpha_m(y) = \alpha_m^+(y) = \ln\frac{1}{|z^m - y|} = \ln\frac{1}{\sqrt{\sum_{n=1}^{2}(z_n^m - y_n)^2}},\ z^m \in Q^+, m = 1, 2, \dots, y \in \partial Q,$$

которую назовём ***системой базисных потенциалов***. Эта система является линейно независимой и замкнутой в $L_2(\partial Q)$ [6], а поскольку само пространство $L_2(\partial Q)$ является сепарабельным и гильбертовым, то замкнутость в нём эквивалентна полноте [8]. Поскольку система базисных потенциалов является полной и линейно независимой, то всякую функцию из пространства $L_2(\partial Q)$ можно с любой степенью точности аппроксимировать линейной комбинацией конечного числа базисных потенциалов, то есть

$$\forall f \in L_2(\partial Q)\ \forall \varepsilon > 0\ \exists \sum_{m=1}^{N} c_m \alpha_m : \left\| f - \sum_{m=1}^{N} c_m \alpha_m \right\|_{L_2(\partial Q)} < \varepsilon.$$

Зная это, мы можем свести решение задачи Дирихле для уравнения Лапласа к нахождению приближенного решения по системе базисных потенциалов: из наперёд заданных условий $\begin{cases}\Delta u|_Q = 0 \\ u|_{\partial Q} = \varphi\end{cases}$ и заданных N точек требуется найти такую комбинацию коэффициентов $c_m, m = 1, 2, \dots$, чтобы функция $\varphi(x_1, \dots, x_n)$ приближалась максимально возможно, то есть

$$F(c) = \left\| \varphi - \sum_{m=1}^{N} c_m \alpha_m \right\|_{L_2(\partial Q)} \to \min.$$

Иначе говоря, мы будем искать проекцию функции φ на линейную оболочку конечного числа функций $\alpha_m, m = 1, 2, \dots, N$; таким образом, мы максимально приблизим граничную функцию φ; то, что найденная приближённая функция будет удовлетворять собственно уравнению Лапласа, следует из того, что она окажется линейной комбинацией фундаментальных решений этого уравнения.

Теперь же требуется свести описанную математическую задачу к системе наиболее простых, дабы начать создавать программу, её решающую.

1.5 Сведение метода базисных потенциалов к комплексу простых задач

Надо найти вектор c из предыдущего пункта. Так как пространство $L_2(\partial Q)$ – гильбертово, то норма в нём выражается через скалярное произведение:

$$\left\| \varphi - \sum_{m=1}^{N} c_m \alpha_m \right\|_{L_2(\partial Q)} = \sqrt{\left(\varphi - \sum_{m=1}^{N} c_m \alpha_m, \varphi - \sum_{m=1}^{N} c_m \alpha_m \right)_{L_2(\partial Q)}}$$

$$= \sqrt{(\varphi, \varphi) - 2 \sum_{m=1}^{N} c_m (\varphi, \alpha_m) + \sum_{n=1}^{N} c_n \sum_{m=1}^{N} c_m (\alpha_n, \alpha_m)}$$

Так как норма является неотрицательной по определению, то предыдущее выражение можно возвести в квадрат без риска потерять решения или приобрести новые:

$$\left\| \varphi - \sum_{m=1}^{N} c_m \alpha_m \right\|^2_{L_2(\partial Q)}$$

$$= (\varphi, \varphi)_{L_2(\partial Q)} - 2 \sum_{m=1}^{N} c_m (\varphi, \alpha_m)_{L_2(\partial Q)}$$

$$+ \sum_{n=1}^{N} c_n \sum_{m=1}^{N} c_m (\alpha_n, \alpha_m)_{L_2(\partial Q)}$$

$$= (\varphi, \varphi)_{L_2(\partial Q)} - 2 \sum_{m=1}^{N} c_m (\varphi, \alpha_m)_{L_2(\partial Q)}$$

$$+ (c_1, c_2, \dots, c_N) \begin{pmatrix} (\alpha_1, \alpha_1)_{L_2(\partial Q)} & \cdots & (\alpha_1, \alpha_N)_{L_2(\partial Q)} \\ \vdots & \ddots & \vdots \\ (\alpha_N, \alpha_1)_{L_2(\partial Q)} & \cdots & (\alpha_N, \alpha_N)_{L_2(\partial Q)} \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_N \end{pmatrix},$$

причём возникшая матрица Грама является невырожденной ввиду линейной независимости системы базисных потенциалов. Наша задача превратилась в минимизацию полученного квадратичного функционала.

Вычислим стационарные точки этого функционала из системы:

$$D_{c_i}F(c) = 0, i = 1, 2, \dots, N,$$

где

$$\begin{aligned}
D_{c_i}F(c) = D_{c_i}\Bigg(& (\varphi,\varphi)_{L_2(\partial Q)} - 2\sum_{m=1}^{N} c_m\,(\varphi,\alpha_m)_{L_2(\partial Q)} \\
& + \sum_{n=1}^{N} c_n \sum_{m=1}^{N} c_m\,(\alpha_n,\alpha_m)_{L_2(\partial Q)} \\
& = (\varphi,\varphi)_{L_2(\partial Q)} - 2\sum_{m=1}^{N} c_m\,(\varphi,\alpha_m)_{L_2(\partial Q)} + \sum_{n=1}^{N} c_n \sum_{m=1}^{N} c_m\,(\alpha_n,\alpha_m)\Bigg) \\
& = -2(\varphi,\alpha_i)_{L_2(\partial Q)} + \sum_{n=1}^{N} D_{c_i}(c_n) \sum_{m=1}^{N} c_m\,(\alpha_n,\alpha_m)_{L_2(\partial Q)} \\
& + \sum_{n=1}^{N} c_n\, D_{c_i} \sum_{m=1}^{N} c_m\,(\alpha_n,\alpha_m)_{L_2(\partial Q)} \\
& = -2(\varphi,\alpha_i)_{L_2(\partial Q)} + \sum_{m=1}^{N} c_m\,(\alpha_i,\alpha_m)_{L_2(\partial Q)} + \sum_{n=1}^{N} c_n\,(\alpha_n,\alpha_i)_{L_2(\partial Q)} \\
& = 2\left(\sum_{m=1}^{N} c_m\,(\alpha_i,\alpha_m)_{L_2(\partial Q)} - (\varphi,\alpha_i)_{L_2(\partial Q)}\right)
\end{aligned}$$

Такая система сводится к виду

$$\Gamma c = \begin{pmatrix} (\varphi,\alpha_1)_{L_2(\partial Q)} \\ \vdots \\ (\varphi,\alpha_N)_{L_2(\partial Q)} \end{pmatrix},$$

где $c = (c_1, c_2, \dots, c_N)$, Γ – матрица Грама нашей системы. Так как эта матрица является невырожденной, система имеет единственное решение; поскольку

второй дифференциал функции $F(c)$ является положительно определённой квадратичной формой с той же матрицей Грама, то в точке решения нашей системы функция $F(c)$ действительно имеет минимум.

Таким образом, мы свели решение задачи Дирихле для уравнения Лапласа $\begin{cases}\Delta u|_Q = 0\\ u|_{\partial Q} = \varphi\end{cases}$ к нахождению приближённого решения $u^N = \sum_{m=1}^{N} c_m\,\alpha_m$, где вектор коэффициентов $c = (c_1, c_2, \dots, c_N)$ является решением системы

$$\begin{pmatrix} (\alpha_1,\alpha_1)_{L_2(\partial Q)} & \cdots & (\alpha_1,\alpha_N)_{L_2(\partial Q)} \\ \vdots & \ddots & \vdots \\ (\alpha_N,\alpha_1)_{L_2(\partial Q)} & \cdots & (\alpha_N,\alpha_N)_{L_2(\partial Q)} \end{pmatrix}\begin{pmatrix} c_1 \\ \vdots \\ c_N \end{pmatrix} = \begin{pmatrix} (\varphi,\alpha_1)_{L_2(\partial Q)} \\ \vdots \\ (\varphi,\alpha_N)_{L_2(\partial Q)} \end{pmatrix},$$

в которой

$$\alpha_m = \alpha_m(x) = \ln\frac{1}{\sqrt{\sum_{n=1}^{2}(z_n^m - x_n)^2}},\ z^m \in R^2\backslash(Q\cup\partial Q), x\in\partial Q,$$

$$(\alpha_n,\alpha_m)_{L_2(\partial Q)} = \oint_{\partial Q}\alpha_n(x)\alpha_m(x)dl,$$

$$(\varphi,\alpha_n)_{L_2(\partial Q)} = \oint_{\partial Q}\alpha_n(x)\,\varphi(x)dl\,, m,n = 1,2,\dots,N$$

Осталось написать программу, которая будет оптимально решать исходную систему линейных алгебраических уравнений, выводить точность найденного решения задачи и создавать графическую иллюстрацию. Дополнительно придётся находить значения криволинейных интегралов первого рода и делать программу универсальной, то есть не нуждающейся в серьёзных изменениях при смене граничной функции φ или области Q.

2 Программная реализация метода базисных потенциалов. Описание главных действующих алгоритмов

Программа нахождения решения задачи Дирихле для уравнения теплопроводности пишется на языке C++ и состоит из следующих ***этапов***:

1) ***Чтение данных и работа с этими данными***. На этом этапе выбирается вариант граничной функции φ, граница области Q и считываются координаты точек плоскости (базисных точек), затем из считанного множества точек плоскости удаляются повторяющиеся элементы в случае их наличия; предполагается, что все точки принадлежат дополнению области Q.

2) ***Поиск решения задачи***. Каждой точке ставится в соответствие функция потенциала, считаются элементы матрицы Грама, решается система линейных алгебраических уравнений с целью нахождения коэффициентов проекции функции φ на линейную комбинацию базисных потенциалов, после этого считается и выводится точность полученного решения.

3) ***Иллюстрация решения***. При помощи графических возможностей языка C++ найденная приближённая функция изображается на рисунке вместе с заранее известным точным решением задачи Дирихле.

Далее проводится подробное описание каждого из этапов.

2.1 Чтение данных и работа с данными

Прежде чем описывать процесс чтения данных (в основном точек), естественно определить некоторую структуру, которая содержит всю информацию, связанную с этими точками, включая и функцию потенциала, соответствующую выбранной точке. Пусть изначально такая структура будет иметь вид:

```
struct basp{
     double x;//координаты точки в двумерном пространстве
     double y;
```

```
    //функция базисного потенциала от точки z, соответствующая
исходной базисной точке
    double potfunc(basp z){
        return log(1/sqrt((z.x-x)*(z.x-x)+(z.y-y)*(z.y-y)));
    }
    //расстояние между точками
    double eudistance(basp z,basp w){
        return sqrt((z.x-w.x)*(z.x-w.x)+(z.y-w.y)*(z.y-w.y));
    }

}
```

2.1.1 Чтение с текстового файла

С текстового файла in.txt считываются строки пояснений и все числовые значения, которые в нём есть; первые два числовых значения – соответствующие номера для области Q и граничной функции φ (последние определены в самом коде), следующие числа – координаты базисных точек; пусть количество числовых значений К, тогда результатом целочисленного деления $\left\lfloor \frac{К-2}{2} \right\rfloor$ выясняется мощность N множества точек плоскости, то есть количество задаваемых точек; далее эти точки снова считываются и сохраняются в массив уже определённых структур `basp`.

Далее производится ***проверка на совпадение введённых точек*** с целью убрать возможную линейную зависимость потенциальных функций; если же этого не сделать, наша задача и вовсе не обязана будет иметь решений, покуда задача Дирихле имеет единственное решение. Алгоритм проверки на совпадение точек таков: точки сортируются по первой координате, потом полученный массив разбивается на участки, где имеются точки с одинаковой первой координатой, далее каждый такой участок сортируется по второй координате, а затем все точки пробегаются; если у соседних точек обе координаты равны, одна из точек удаляется, а пробег продолжается, покуда все точки не будут пройдены. В конечном итоге получается окончательный массив точек плоскости; если мощность исходного множества уменьшится, это зафиксируется. Для

сортировки используется библиотечная функция `sort()`с наиболее оптимальной сложностью.

2.1.2 Генерация массива точек программным образом

Если же количество базисных точек должно стать большим настолько, что их неразумно записывать в файл, то в этом случае будет использоваться функция генерации таких точек. Идея заключается в следующем: во-первых, в плоскости R^2берётся замкнутая кривая, близкая к границе ∂Q области, в которой решается дифференциальное уравнение (при этом для простоты будут браться кривые, подобные ∂Q, причём сама ∂Q может быть одной из них), затем отрезок, образом которого она является, разбивается точками на фиксированное количество равных частей, после в малых окрестностях образов этих точек (дабы не все они лежали на кривой) случайно выбираются такие точки, которые лежат во внешности области Q; так мы получаем массив точек, равномерно располагающихся в окрестности выбранной кривой; но для удобства последующего исследования куда лучше сделать ещё и так, чтобы для любой мощности и всякое взятое подмножество нашего множества представляло собой точки, равномерно располагающиеся во всей окрестности кривой. Последнюю задачу можно решить простым «перемешиванием» массива, которое даст результат не многим худший, нежели если попытаться расставить точки на кривой по строгим и сложным алгоритмам.

2.2 Поиск решения

Как мы установили, поиск приближённого решения задачи Дирихле для уравнения Лапласа заключается в решении системы линейных уравнений

$$\begin{pmatrix} (\alpha_1, \alpha_1)_{L_2(\partial Q)} & \cdots & (\alpha_1, \alpha_N)_{L_2(\partial Q)} \\ \vdots & \ddots & \vdots \\ (\alpha_N, \alpha_1)_{L_2(\partial Q)} & \cdots & (\alpha_N, \alpha_N)_{L_2(\partial Q)} \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_N \end{pmatrix} = \begin{pmatrix} (\varphi, \alpha_1)_{L_2(\partial Q)} \\ \vdots \\ (\varphi, \alpha_N)_{L_2(\partial Q)} \end{pmatrix}.$$

При этом возникает дополнительная задача нахождения скалярного произведения функций из пространства $L_2(\partial Q)$, которое представляет из себя криволинейный интеграл первого рода по границе нашей области:

$$(\alpha_n, \alpha_m)_{L_2(\partial Q)} = \oint_{\partial Q} \alpha_n(x)\alpha_m(x)dl$$

Составим алгоритмы для решения каждой из поставленных задач.

2.2.1 Нахождение криволинейного интеграла первого рода

Пусть граница ∂Q задаётся кривой $L\big(x(t), y(t)\big), a \le t < b$. В таком случае для определённой на этой кривой функции $f(z) = f(x, y)$ криволинейный интеграл первого рода может быть вычислен приближённо:

$$\oint_L f(z)dl = \sum_{k=0}^{M-1} f\left(x\left(\frac{t_{k+1} + t_k}{2}\right), y\left(\frac{t_{k+1} + t_k}{2}\right)\right)$$
$$\times \sqrt{\big(x(t_{k+1}) - x(t_k)\big)^2 + \big(y(t_{k+1}) - y(t_k)\big)^2}, t_{k+1} - t_k = \frac{b-a}{M}$$
$$= h$$

при достаточно большом M.

Если использовать метод трапеций, то точность получится большей. Этот метод выражается в формуле:

$$\oint_L f(z)dl = \sum_{k=1}^{M} \frac{f\big(x(t_{k-1}), y(t_{k-1})\big) + f\big(x(t_k), y(t_k)\big)}{2}$$
$$\times \sqrt{\big(x(t_{k-1}) - x(t_k)\big)^2 + \big(y(t_{k-1}) - y(t_k)\big)^2}, t_{k+1} - t_k = \frac{b-a}{M}$$

Реализовать же такой метод можно, создав класс функций, задающих кривые, причём основным методом этого класса должно быть возвращение точки (x, y) по заданному параметру t. Выглядеть это будет примерно так:

```
class curve {//класс кривых
private:
    double a;//начальное значение параметра
```

```
    double b;//конечное значение параметра
public:
    double h=(b-a)/M;//значение шага для этой кривой
    basp transfer(double t){//возврат точки на кривой по значе-
нию параметра
        basp point;
        point.x=3*cos(t);
        point.y=sin(t);
        return point;
    }
}
```

2.2.2 Нахождение решений СЛАУ с симметрической матрицей

Так как матрица Грама Г является симметрической, то для решения уравнения $\Gamma x = b$ разумнее пользоваться методами, показывающими на симметрических матрицах б**о**льшую экономичность и точность в сравнении с общими методами; по крайней мере, изначально поведение решения является неизвестным, поэтому пока что я лишь предполагаю, что методы для симметрических матриц дадут наиболее устойчивые решение дифференциального уравнения. К одному из них относится метод квадратного корня (метод Холецкого), который заключается в следующем. Всякую симметрическую матрицу Г можно разложить на произведение двух взаимно транспонированных треугольных матриц:

$$\Gamma = \mathrm{T}^{\mathrm{T}}\mathrm{T},$$

где

$$\mathrm{T} = \begin{pmatrix} t_{11} & \cdots & t_{1N} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & t_{NN} \end{pmatrix}, \mathrm{T}^{\mathrm{T}} = \begin{pmatrix} t_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ t_{NN} & \cdots & t_{1N} \end{pmatrix}.$$

В таком случае после перемножения матриц выводятся формулы для коэффициентов матрицы Т через коэффициенты матрицы $\Gamma = [a_{ij}]$:

$$\begin{cases} t_{11} = \sqrt{a_{11}} \\ t_{1j} = \dfrac{a_{1j}}{t_{11}}, 1 < j \le N \\ t_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} t_{ki}^2}, 1 < i \le N \\ t_{ij} = \dfrac{a_{ij} - \sum_{k=1}^{i-1} t_{ki} t_{kj}}{t_{ii}}, i < j \\ t_{ij} = 0, i > j \end{cases}$$

Поскольку $\Gamma = \mathrm{T}^{\mathrm{T}}\mathrm{T}$, то система $\Gamma x = b$ сводится к виду:

$$\begin{cases} \mathrm{T}^{\mathrm{T}} y = b \\ \mathrm{T} x = y \end{cases}.$$

Решая эту систему, мы придём к рекуррентным отношениям:

$$\begin{cases} y_1 = \dfrac{b_1}{t_{11}} \\ y_i = \dfrac{b_i - \sum_{k=1}^{i-1} t_{ki} y_k}{t_{ii}}, 1 < i \le N \\ x_N = \dfrac{y_N}{t_{NN}} \\ x_i = \dfrac{y_i - \sum_{k=i+1}^{N} t_{ik} x_k}{t_{ii}}, N > i \ge 1 \end{cases}$$

2.2.3 Вывод точности

После нахождения решения в виде $\sum_{m=1}^{N} c_m \alpha_m$, где α_m – известные базисные потенциалы, c_m – найденные константы, **в выходной файл для показания точности аппроксимации будет выведено расстояние между приближённым решением и функцией φ**, равное

$$\rho\left(\sum_{m=1}^{N} c_m\,\alpha_m, \varphi\right) = \left\|\varphi - \sum_{m=1}^{N} c_m\,\alpha_m\right\|_{L_2(\partial Q)} = \sqrt{\oint_{\partial Q}\left(\sum_{m=1}^{N} c_m\,\alpha_m - \varphi\right)^2 dl}$$

$$= \left((\varphi,\varphi)_{L_2(\partial Q)} - 2\sum_{m=1}^{N} c_m\,(\varphi,\alpha_m)_{L_2(\partial Q)} \right.$$

$$\left. + (c_1, c_2, \dots, c_N)\begin{pmatrix} (\alpha_1,\alpha_1)_{L_2(\partial Q)} & \cdots & (\alpha_1,\alpha_N)_{L_2(\partial Q)} \\ \vdots & \ddots & \vdots \\ (\alpha_N,\alpha_1)_{L_2(\partial Q)} & \cdots & (\alpha_N,\alpha_N)_{L_2(\partial Q)} \end{pmatrix}\begin{pmatrix} c_1 \\ \vdots \\ c_N \end{pmatrix} \right)^{\frac{1}{2}}$$

$$= \left((\varphi,\varphi)_{L_2(\partial Q)} - 2(c_1, c_2, \dots, c_N)\begin{pmatrix} (\varphi,\alpha_1)_{L_2(\partial Q)} \\ \vdots \\ (\varphi,\alpha_N)_{L_2(\partial Q)} \end{pmatrix} \right.$$

$$\left. + (c_1, c_2, \dots, c_N)\begin{pmatrix} (\varphi,\alpha_1)_{L_2(\partial Q)} \\ \vdots \\ (\varphi,\alpha_N)_{L_2(\partial Q)} \end{pmatrix} \right)^{\frac{1}{2}}$$

$$= \left((\varphi,\varphi)_{L_2(\partial Q)} - (c_1, c_2, \dots, c_N)\begin{pmatrix} (\varphi,\alpha_1)_{L_2(\partial Q)} \\ \vdots \\ (\varphi,\alpha_N)_{L_2(\partial Q)} \end{pmatrix} \right)^{\frac{1}{2}}$$

2.3 Иллюстрация решения

С помощью библиотеки “Graph.h” в трёхмерном пространстве рисуется график исходной граничной функции и найденной приближённой функции. Чтобы рисование графика было удобным и автоматизированным, необходимо приближённую функцию рассматривать как метод некоторого класса, который является наследственным по отношению к описанным ранее.

Кроме иллюстрации самого решения, программа будет иллюстрировать графики (число базисных потенциалов)-(качество приближения), (близость

выбранной кривой к границе исходной области)-(качество приближения при фиксированном числе базисных точек).

2.4 Временная сложность алгоритма

Пусть N_1– количество считанных точек. Тогда сортировка этих точек и отсеивание повторяющихся занимает $O(N_1 \times \lg(N_1))$ операций. В результате мы получаем массив неповторяющихся базисных точек размерности $N_2 \leq N_1$. Далее происходит заполнение системы линейных уравнений, для чего требуется найти $N_2 \times (N_2 + 1)$ криволинейных интегралов; вычисление каждого интеграла через M шагов будет занимать $O(M)$ операций; следовательно, заполнение системы стоит нам $O(MN_2^2)$ операций. Решение этой системы методом Холецкого будет занимать $O(N_2^3)$ операций. В таком случае общая сложность алгоритма равняется:

$$O(N_1 \times \lg(N_1)) + O(MN_2^2) + O(N_2^3) =$$
$$= O(N_1 \times lg(N_1)) + N_2^2\big(O(M) + O(N_2)\big)$$
$$= \begin{cases} O(MN_2^2), M \geq N_2, N_1 \times lg(N_1) \leq N_2^2 \\ O(N_2^3), M < N_2 \end{cases}$$

3 Входные данные программы

Создание необходимых классов представило большую сложность ввиду требований знания языка на высоком уровне. Здесь будут описаны основные классы созданной программы, а также области и граничные функции, на которых проводится тестирование. Затем будет рассказано о результатах тестирования при разных областях, разных граничных функциях, разном числе и расположении базисных точек.

3.1 Тестовые области

В качестве тестовых областей будут использоваться простейшие: круг, внутренность треугольника, внутренность квадрата. Несмотря на их простоту, не так просто задать параметризацию границ этих областей так, чтобы ей можно было пользоваться в программе. Но придётся это сделать.

3.1.1 Параметризация границы круга

Как известно, параметризация окружности с центром в начале координат и радиусом a может выглядеть так:

$$\begin{cases} x(t) = a\cos t\,, t \in (0, 2\pi] \\ y(t) = a\sin t\,, t \in (0, 2\pi] \end{cases}.$$

В таком случае окружность радиуса, например, 3 будет как на рисунке 1.1.

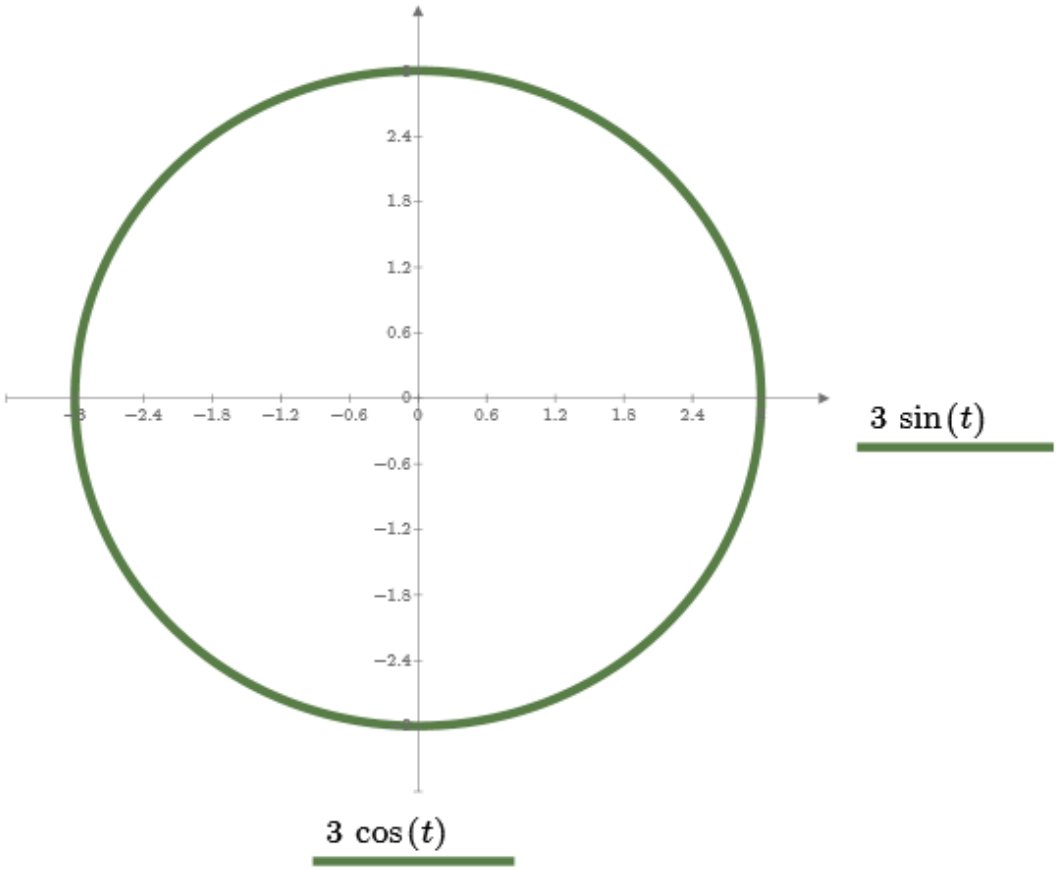

Рисунок 1.1 Выполненный в Mathcad Prime график окружности с центром в начале координат и радиусом 3, параметризованной известным образом.

3.1.2 Параметризация треугольника

Пусть нам нужно параметризовать равносторонний треугольник со стороной 4. Удобнее будет, если одна из его вершин совпадает с началом координат, а одна из сторон лежит на оси координат. Например, подойдёт такая параметризация треугольника с вершинами в точках (0,0), (4,0), $(2,2\sqrt{3})$, которую можно вывести с помощью известного уравнения прямой, проходящей через две точки[1]:

$$x(t) = \begin{cases} x = t, t \in [0,4] \\ x = 12 - 2t, t \in [4,6]' \end{cases}$$

$$y(t) = \begin{cases} y = t\sqrt{3}, t \in [0,2] \\ y = -t\sqrt{3} + 4\sqrt{3}, t \in [2,4] \\ y = 0, t \in [4,6] \end{cases}.$$

[1] $\frac{x - x_{ki}}{(x_{kj} - x_{ki})} = \frac{y - y_{ki}}{(y_{kj} - y_{ki})}$

В результате получим треугольник, изображённый на рисунке 1.2.

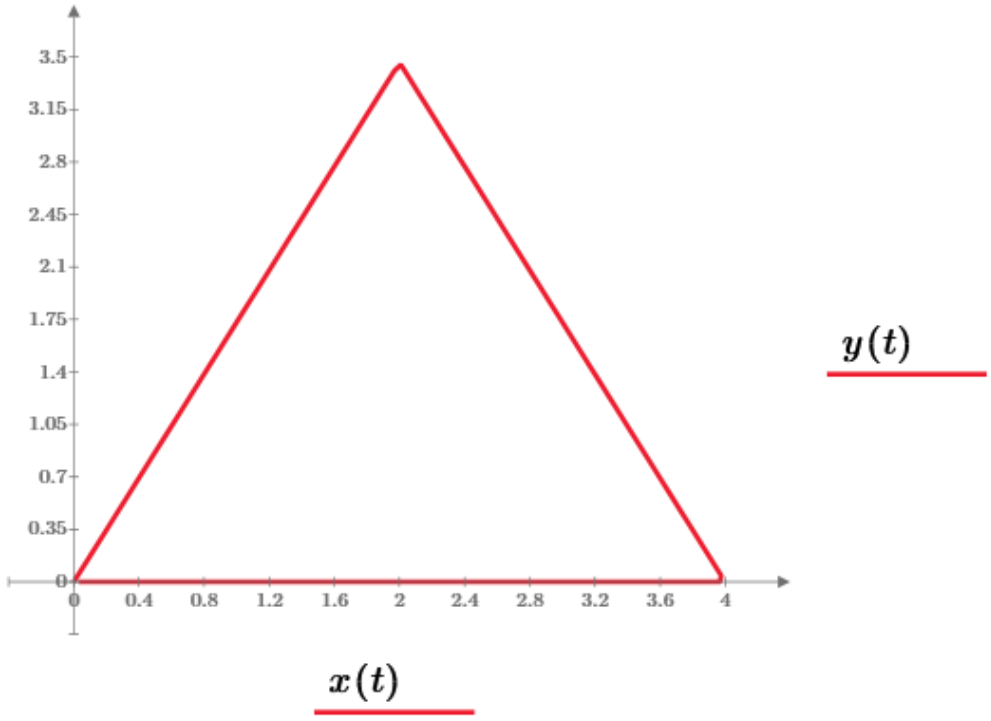

Рисунок 1.2. Выполненный в Mathcad Prime график равностороннего треугольника со стороной 4, параметризованного известным образом.

3.1.3 Параметризация границы квадрата

По полной аналогии с предыдущим пунктом зададим параметризацию квадрата:

$$x(t) = \begin{cases} x = t, t \in [0,4] \\ x = 4, t \in [4,8] \\ x = 12 - t, t \in [8,12]' \\ x = 0, t \in [12,16] \end{cases}$$

$$y(t) = \begin{cases} y = 0, t \in [0,4] \\ y = t - 4, t \in [4,8] \\ y = 4, t \in [8,12] \\ y = 16 - t, t \in [12,16] \end{cases}.$$

Результат изображён на рисунке 1.3.

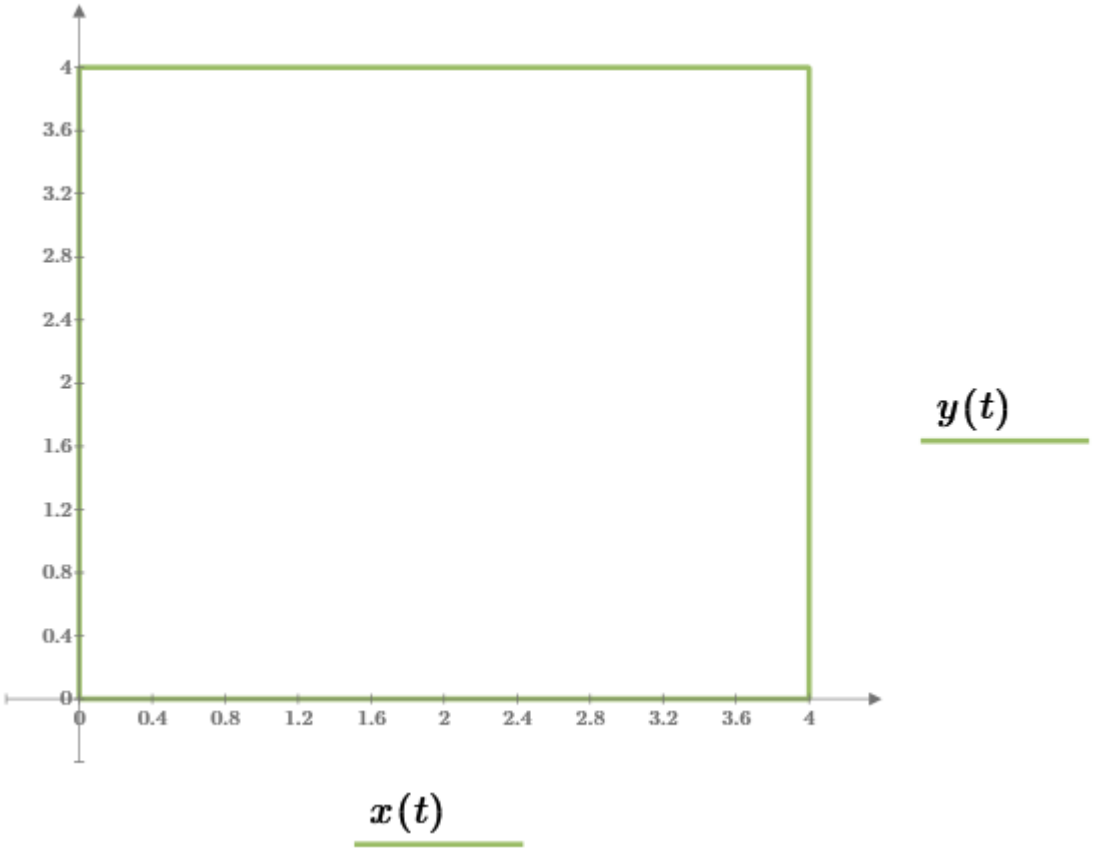

Рисунок 1.3. Выполненный в Mathcad Prime график квадрата со стороной 4, параметризованного известным образом.

3.1.4 Параметризация «острия»

Последней областью является внутренность «острия», граница которого представляет кусочно-гладкий контур $S = S_1 \cup S_2 \cup S_3$, где

$$\begin{cases} S_1 = \{(x,y): y = 0, x \in [0,1]\} \\ S_2 = \left\{(x,y): y = \sqrt{1-x^2}, x \in [0.5,1]\right\} \\ S_3 = \left\{(x,y): y = \sqrt{1-(x-1)^2}, x \in [0,0.5]\right\} \end{cases}$$

3.2 Тестовые граничные функции

Наша внутренняя задача Дирихле

$$\begin{cases} \Delta u|_Q = 0 \\ u|_{\partial Q} = \varphi \end{cases}$$

разрешима единственным образом в тех случаях, когда $\varphi \in L_2(\partial Q)$[7]. В качестве граничных функций возьмём следующие:

$$1)\ \varphi(z) = \varphi(x,y) = \cos x \cos y\,,$$

$$2)\ \varphi(x,y) = \sin y\,,$$

$$3)\ \varphi(x,y) = x^2 + 4,$$

$$4)\ \varphi(x,y) = const,$$

$$5)\ \varphi(x,y) = \ln\frac{1}{\sqrt{(z_2^1 - y)^2 + (z_1^1 - x)^2}},$$

$$6)\ \varphi(x,y) = \ln(|xy| + 1) + 2x\,,$$

$$7)\ \varphi(x,y) = \begin{cases} \cos 2x * \sec yx\,, \cos xy \neq 0 \\ 1, \cos xy = 0 \end{cases}$$

$$8)\ \varphi(x,y) = \begin{cases} 0, (x,y) \in S_1 \\ \frac{1}{2}, (x,y) \in S_2 \\ -\frac{1}{2}, (x,y) \in S_3 \end{cases}$$

$$\xrightarrow{\text{а. п.}} \begin{cases} 0, -\frac{2\pi}{3} \leq \arg\left(\left(x - \frac{1}{2}\right) + i\left(y - \frac{\sqrt{3}}{2}\right)\right) \leq -\frac{\pi}{3} \\ \frac{1}{2}, -\frac{\pi}{3} \leq \arg\left(\left(x - \frac{1}{2}\right) + i\left(y - \frac{\sqrt{3}}{2}\right)\right) \leq \frac{\pi}{2} \\ -\frac{1}{2}, \frac{\pi}{2} \leq \arg\left(\left(x - \frac{1}{2}\right) + i\left(y - \frac{\sqrt{3}}{2}\right)\right) \leq \frac{4\pi}{3} \\ \arg(x + iy) = \operatorname{arctg}\left(\frac{y}{x}\right) + \operatorname{sign}(|x| - x) \times \pi \times \operatorname{sign}(y) \end{cases}.$$

Выбранные таким образом функции на границах указанных областей будут кусочно-гладкими, иногда будут иметь разрывы первого рода, иногда – устранимые разрывы.

3.3 Описание наиболее важных используемых классов и структур, глобальных переменных, некоторых функций и процедур

Согласно ***Бьярне Страуструпу***, “цель программиста – выразить вычисления, причём это должно быть сделано **1)** правильно, **2)** просто, **3)** эффективно”[2].

Именно поэтому код программы должен быть снабжён множественными комментариями, переменные должны быть названы удобными именами, а сама программа, желательно, должна быть разбита на большое число отдельных функций и процедур, чтобы легче было проводить её отладку, и читать, и понимать; если же программа должна решать крупные задачи и не требовать крупных изменений для решения аналогичных задач, в ней не обойтись без ***классов***. Если общий алгоритм программы требовал пояснений, то отдельные функции в них не нуждаются; тем не менее, общий алгоритм представляет собой взаимодействие множества небольших алгоритмов, каждый из которых требует определённого количества времени и системных ресурсов, а потому оптимизация каждого из них – дело существенное, сложное, но важное.

В программе определена **структура «базисная точка»** с полями координат точки на плоскости:

```
struct basispoint {
          double x;//координаты точки в двумерном пространстве
          double y;
          //функция базисного потенциала в точке z, соответствующая исходной базисной точке
          double potentialf(basispoint z) {
               return log(1 / sqrt((z.x - x)*(z.x - x) + (z.y - y)*(z.y - y)));
          }
     };
```

Также в структуре определён метод, сопоставляющий точке функцию базисного потенциала, что реализуется формулой

[2] «Программирование: принципы и практика с использованием C++»

$$\alpha_m(y) = \ln\frac{1}{|z^m - y|} = \ln\frac{1}{\sqrt{\sum_{n=1}^{2}(z_n^m - y_n)^2}}$$

В программе реализуется **класс кривых**; к этому классу будет относиться граница области, в которой решается дифференциальное уравнение, и кривая, в окрестности которой берутся базисные точки; в этом же классе содержится метод, находящий криволинейный интеграл первого рода от определённой на кривой функции $\varphi(i,j) = \alpha_i \times \alpha_j$.

```
class curve {
private:
	double(*u)(double);//параметризации координат
	double(*v)(double);
	double h; //значение шага для этой кривой
	int M;//число шагов
	void geth(int MM) {
		M = MM;
		h = (b - a) / M; //присвоение шагу конкретного значения
	}
public:
	basispoint transfer(double t) {//возврат точки на кривой по значению параметра
	}
	double a;//начальное значение параметра
	double b;//конечное значение параметра
	curve(double a0, double b0, double(*uu)(double), double(*vv)(double)) {//конструктор
	}
	curve() {//простейший конструктор
		a = 0; b = 0;
	}

	double firstkind(int i, int j) {//вычисление криволинейного интеграла первого рода по этой кривой от функции fpow(int i,int j,basispoint z)
```

```
    }
};
```

В программе используется **класс систем линейных уравнений**, в которых содержатся разные методы решения этих систем как методы класса:

```
//класс СЛАУ с методами их решения
class systems {
private:
    int n;//размерность системы
public:
    double **a, *b, *x;//указатели на матрицу и векторы (свободный и решения)

    void make(int k) {//создание двумерного и одномерных динамических массиво с заданной размерностью
        n = k;
        a = new double *[n];
        for (int i = 0; i < n; i++)
            a[i] = new double[n];
        b = new double[n];
        x = new double[n];
    }

    void Gauss(int t) {////метод Гаусса}

    void Holets() { //решение уравнения Ax=b методом Холецкого, присвоение вектору x значений решения}

    void Jak() {//метод Якоби}
                                …
};
```

Перечислим **основные глобальные переменные** программы:

```
basispoint *mas;//указатель на массив точек плоскости
curve mycurve;//граница области, в которой решается диф. уравнение
```

```
systems mysystem;//система, которую придётся решить
double(*fi)(basispoint);//указатель на граничную функцию
int circle, gf, N;//номер области, граничной функции и количество базисных точек
const int kgf = 2;//количество граничных функций на одну кривую
const double eps = 0.0001;//используемая в некоторых местах погрешность
```

Так как код программы получился крайне объёмным, многие её функции и процедуры пришлось выделять в отдельные пространства имён, что может усложнить понимание кода. **Главная** функция программы имеет вид:

```
int main()
{
    setlocale(LC_ALL, "Russian");

    desigion(400);//заполнение массива из файла (0) или при генерировании (>0), решение и вывод решения
    illustrating(fi, mas, mysystem.x, mycurve, 1);// график граничной функции и приближения
    fixity();//график зависимости погрешности аппроксимации от числа базисных точек
    quality(100);//график зависимости погрешности от кривой, возле которой берутся базисные точки
    return 0;
}
```

4 Исследование устойчивости приближённого решения и качества аппроксимации

Тестирование готовой программы позволяет изучать свойства приближённого решения, качество его аппроксимации, устойчивость. Такое тестирование позволяет приходить к выводам, к которым не придёшь никаким другим способом. Проведя тестирование написанной мной программы, я пришёл к выводу, что 1) метод базисных потенциалов действительно является очень эффективным даже при малом числе базисных функций, 2) в зависимости от способа решения СЛАУ приближённое решение нашей задачи показывает разную степень неустойчивости, 3) качество аппроксимации улучшается по мере приближения базисных точек к границе области, в которой решается дифференциальное уравнение. Исследуем каждый из трёх пунктов подробнее.

4.1 Эффективность метода базисных потенциалов

Возьмём за границу области окружность радиуса 0.5 и базисные точки будем выбирать вблизи окружности радиуса 0.7, причём обе окружности являются концентрическими. За граничные функции возьмём в первом случае $\varphi(x, y) = \cos x \cos y$, во втором, к примеру, $\varphi(x, y) = \sin y$; покажем, что метод базисных потенциалов является крайне действенным даже при небольшом числе базисных функций, как то 3, 5, 8. Тестирование показало, что:

1) Разница приближённого решения с граничной функцией при числе базисных функций 3 равна 0.370718 для функции $\varphi(x, y) = \cos x \cos y$ и 0.212332 для $\varphi(x, y) = \sin y$. Этому соответствуют два графика (в плоскости $t \to \varphi(x(t), y(t))$, причём далее граничная функция обозначается зелёным, приближённое решение – фиолетовым; если на рисунке видна только фиолетовая кривая, то приближение настолько хорошее, что погрешности аппроксимации не видно):

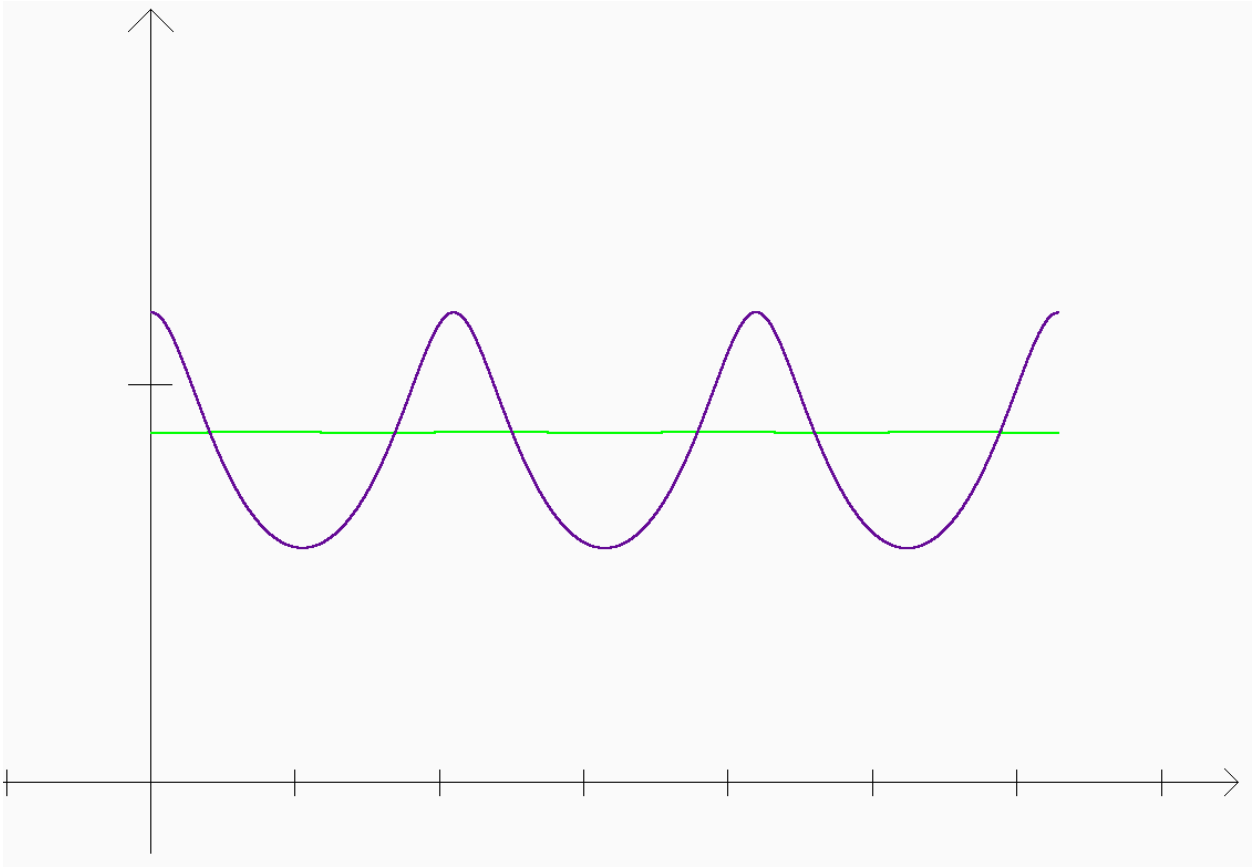

Рисунок 2.1. График приближения для первой граничной функции при трёх базисных функциях

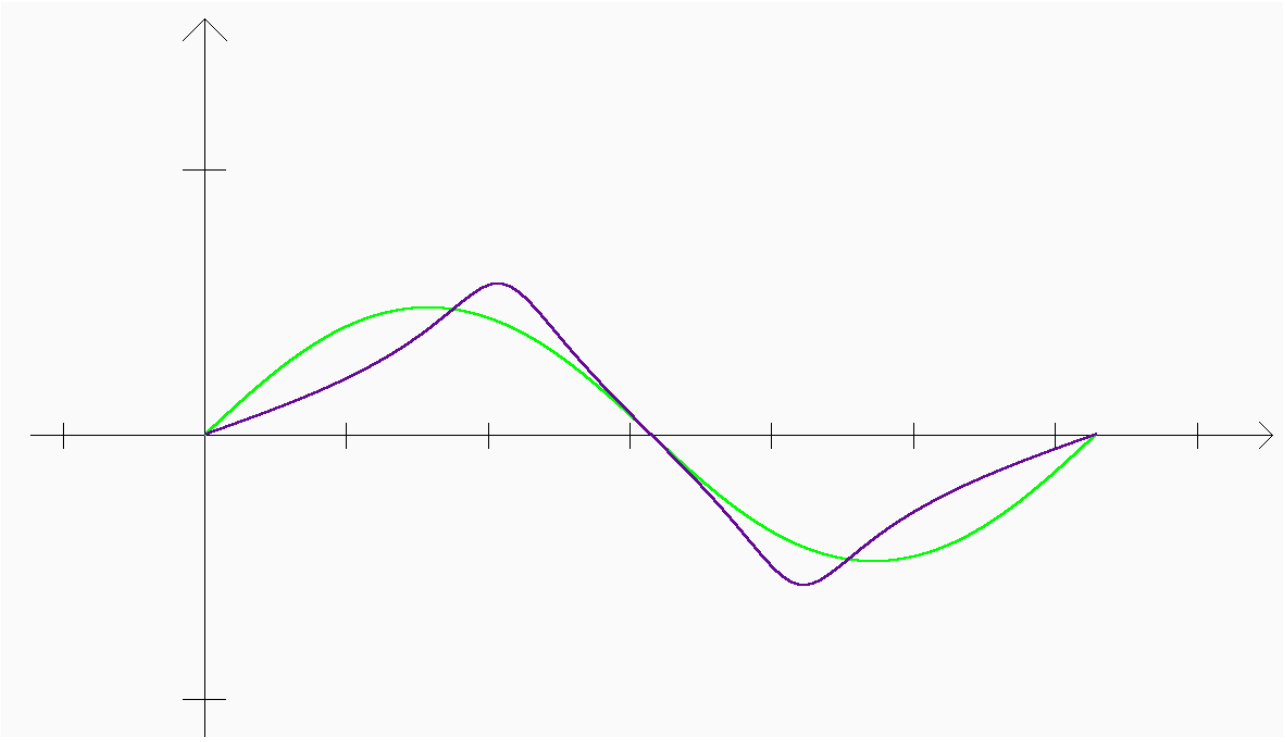

Рисунок 2.2. График приближённого решения для второй функции при трёх базисных функциях

2) Для пяти граничных функции имеем погрешности 0.11501 и 0.0586778 соответственно. Им соответствуют графики:

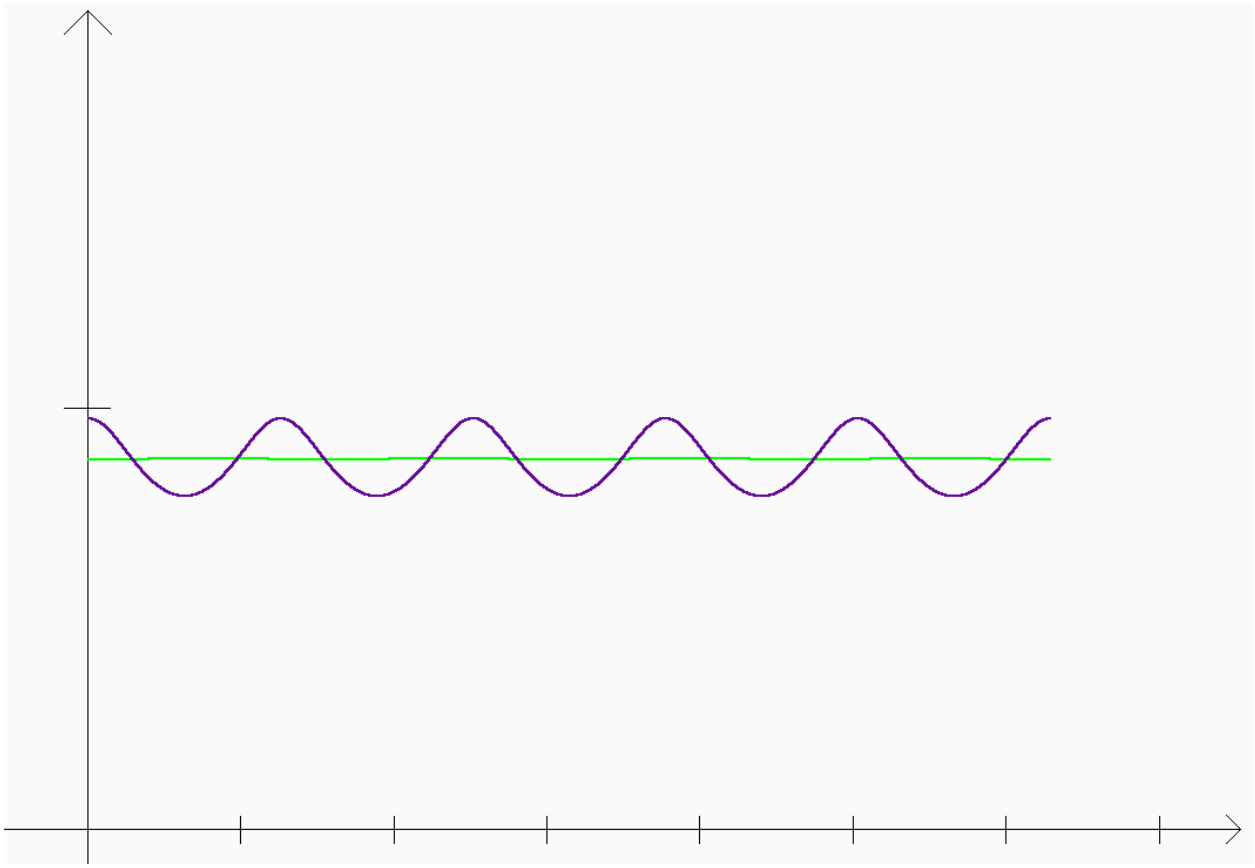

Рисунок 2.3. График приближённого решения для первой функции при пяти базисных функциях

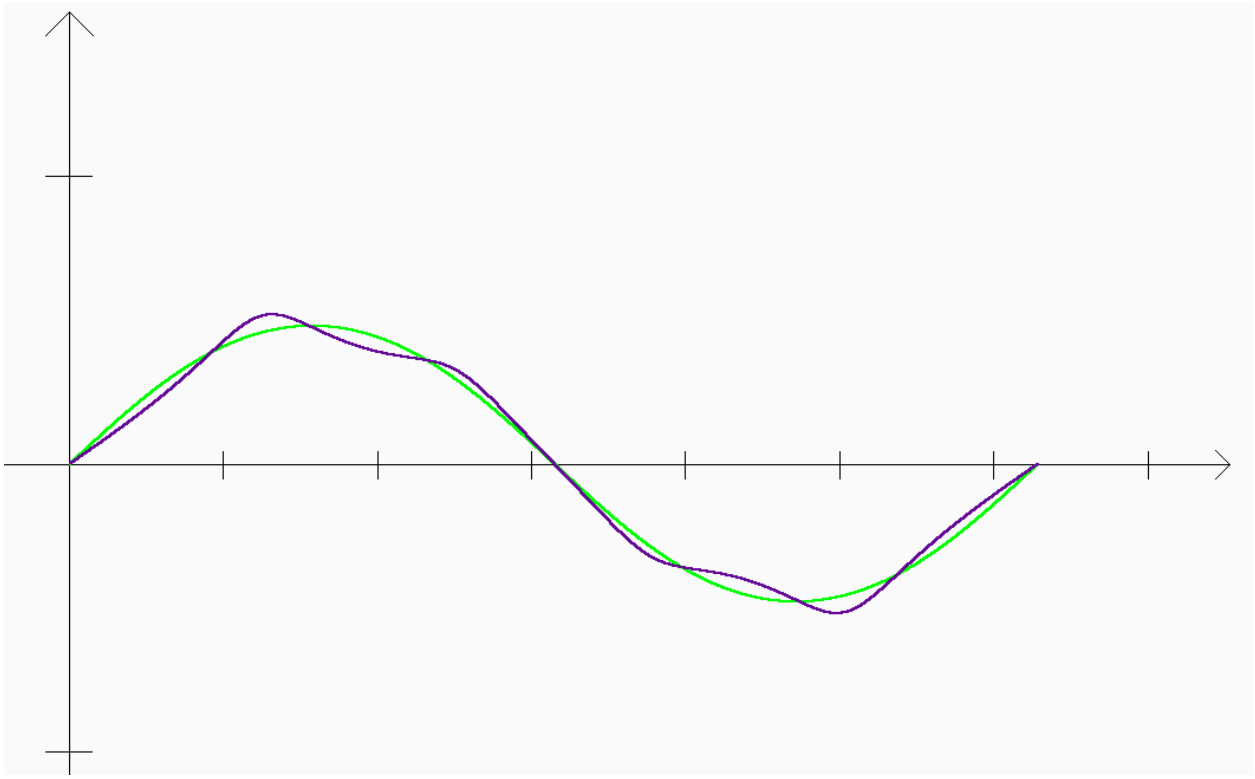

Рисунок 2.4. График приближённого решения для второй функции при пяти базисных функциях

3) Для восьми граничных функций имеем погрешности 0.0261588 и 0.0125437 и графики:

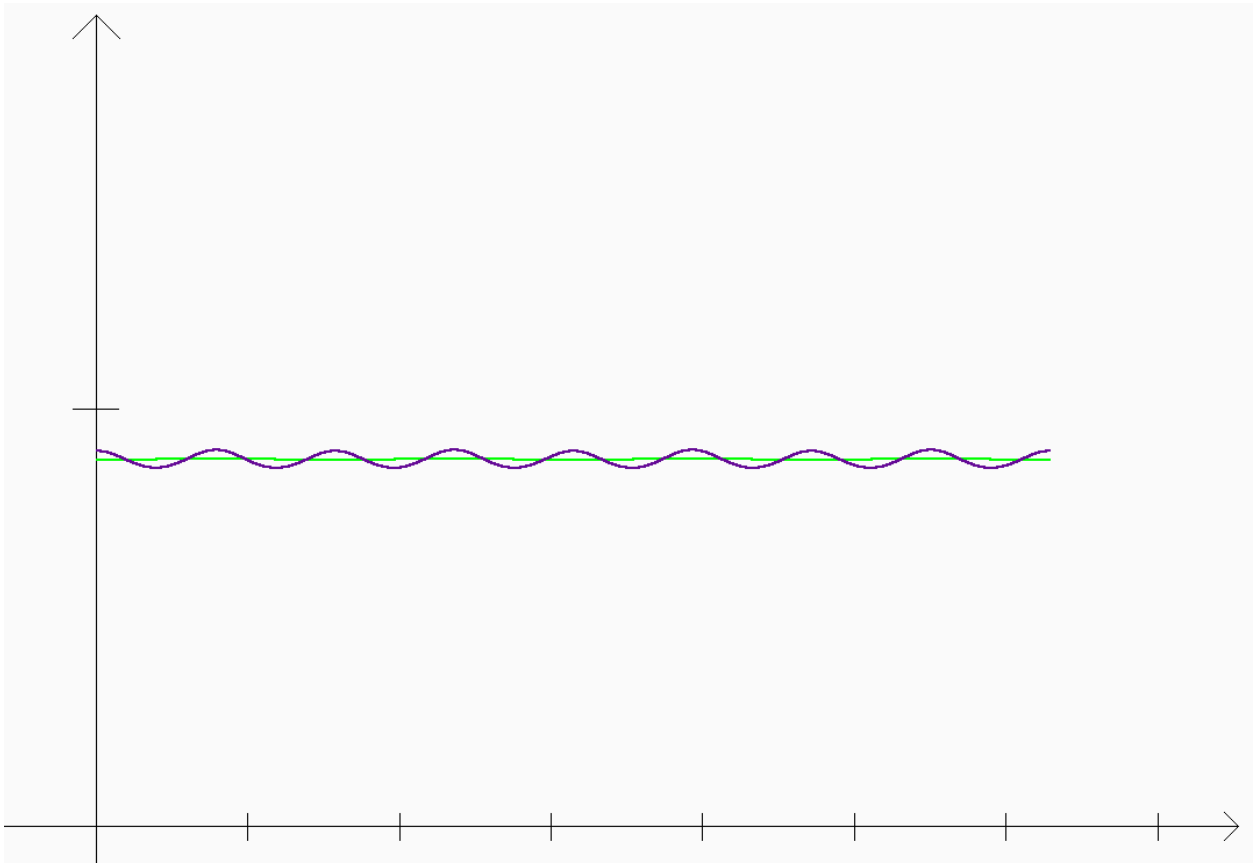

Рисунок 2.5. График приближённого решения для первой функции при восьми базисных функциях

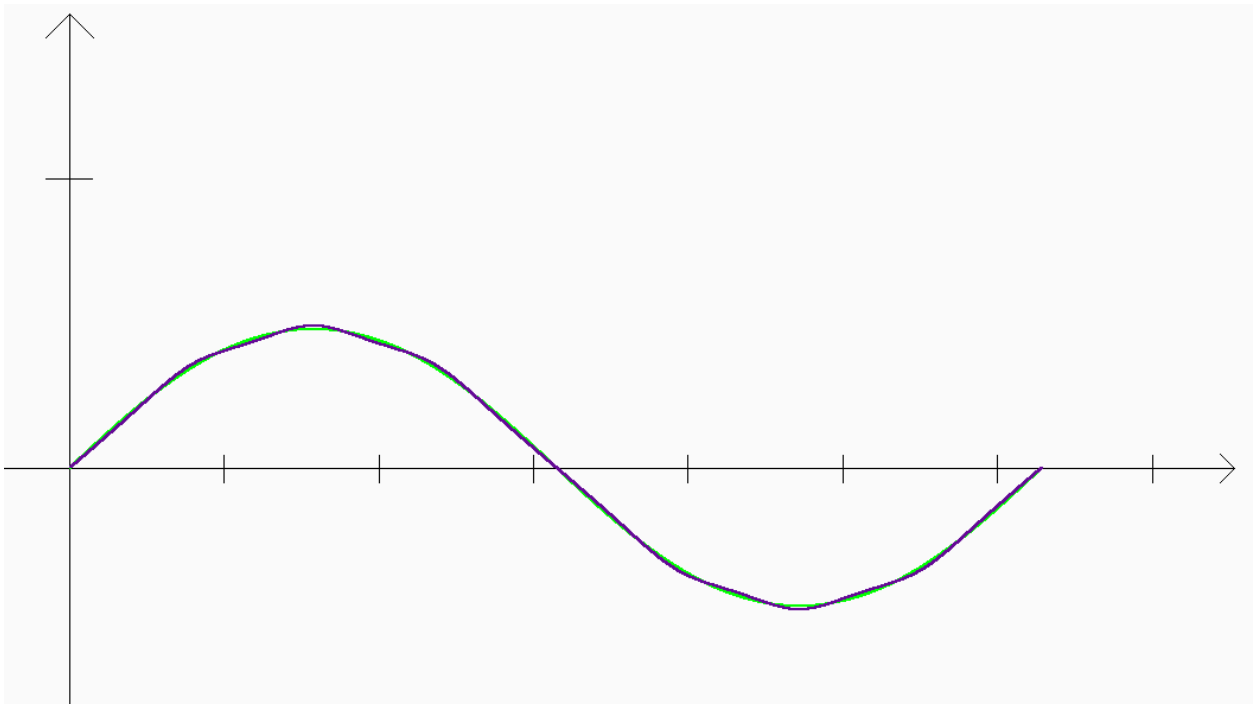

Рисунок 2.6. График приближённого решения для второй функции при восьми базисных функциях

Из показанных результатов видно, что метод базисных потенциалов обеспечивает хорошую сходимость уже при малом числе базисных функций.

Это останется верным и для, например, кусочно-гладких функций, чьи графики тоже можно изобразить. Следующие рисунки показывают приближение других граничных функций, заданных на квадрате со стороной 0.5 (его параметризация описана в 3.1.3) системой базисных потенциалов из 3-7-ми потенциалов:

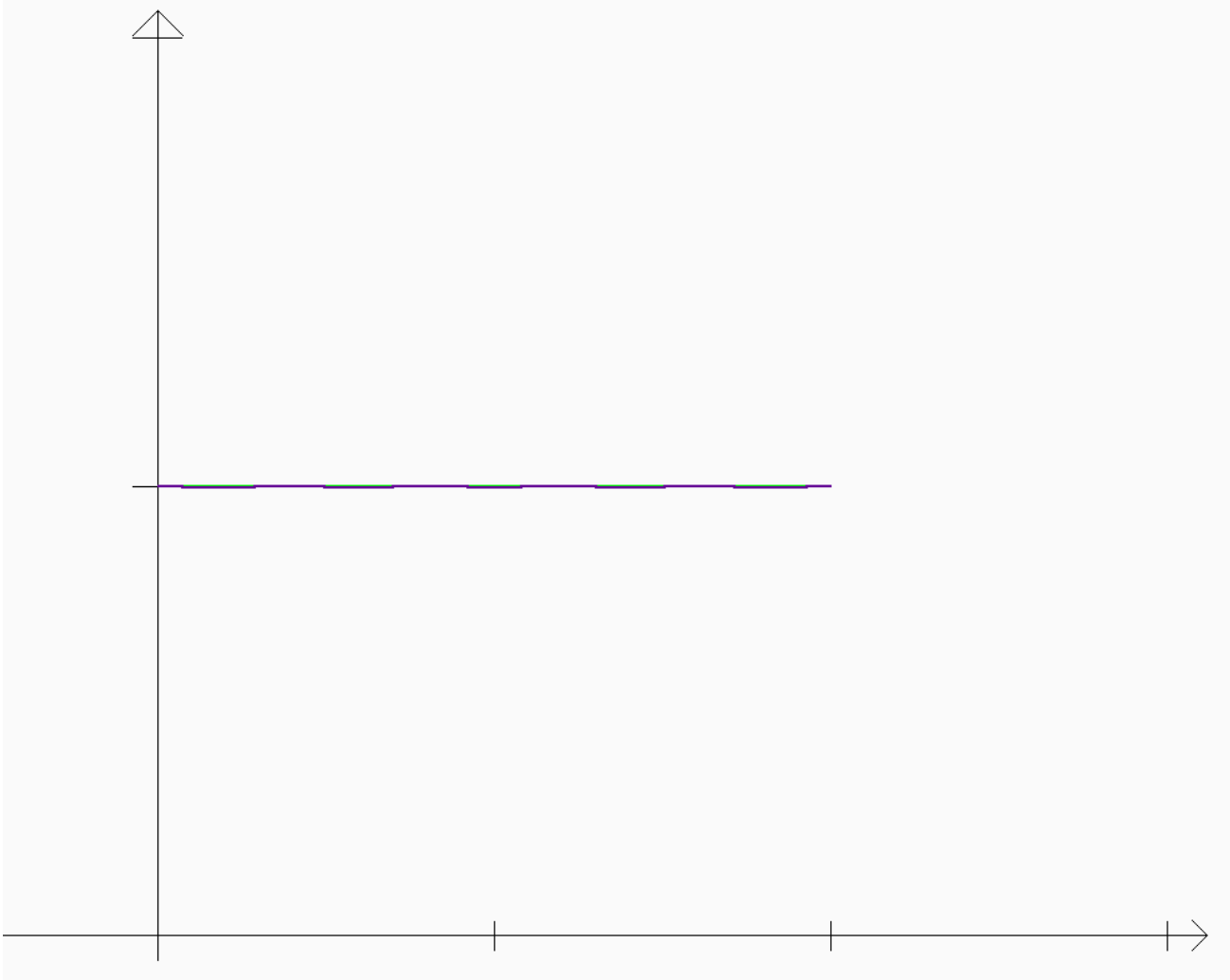

Рисунок 2.7 График приближённого решения для 4-й функции (константы, равной единице) при семи базисных функциях

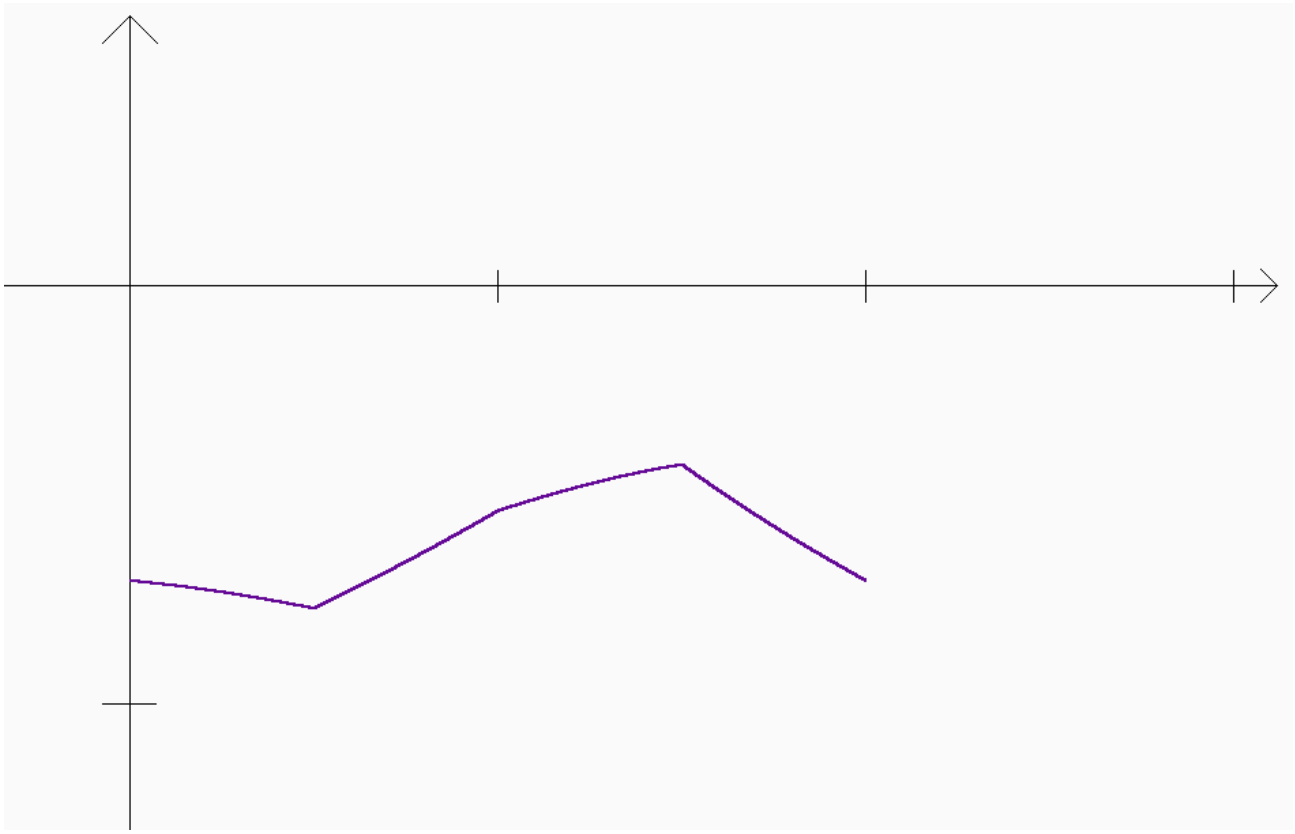

Рисунок 2.8 График приближённого решения для 5-й функции при семи базисных функциях

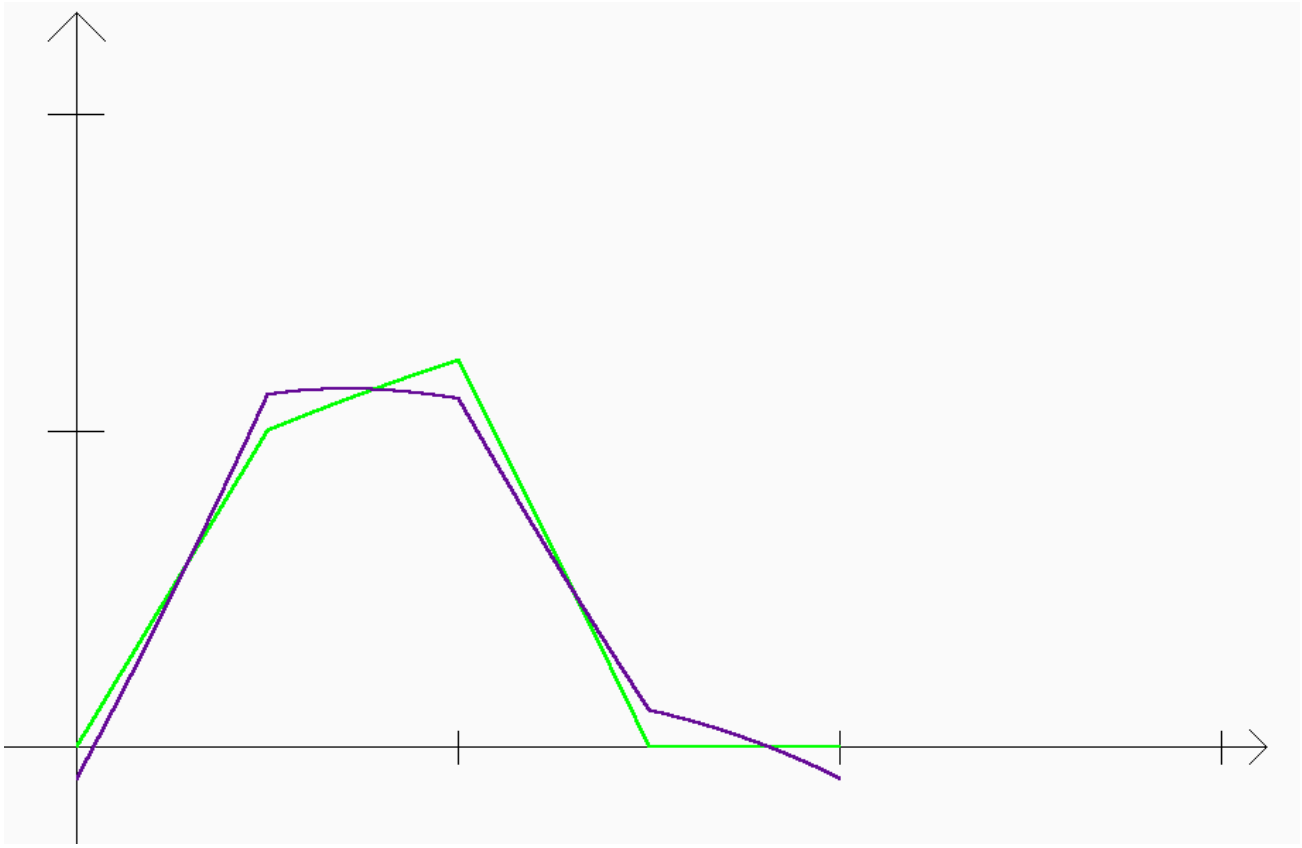

Рисунок 2.9 График приближённого решения для 6-й функции при трёх базисных функциях

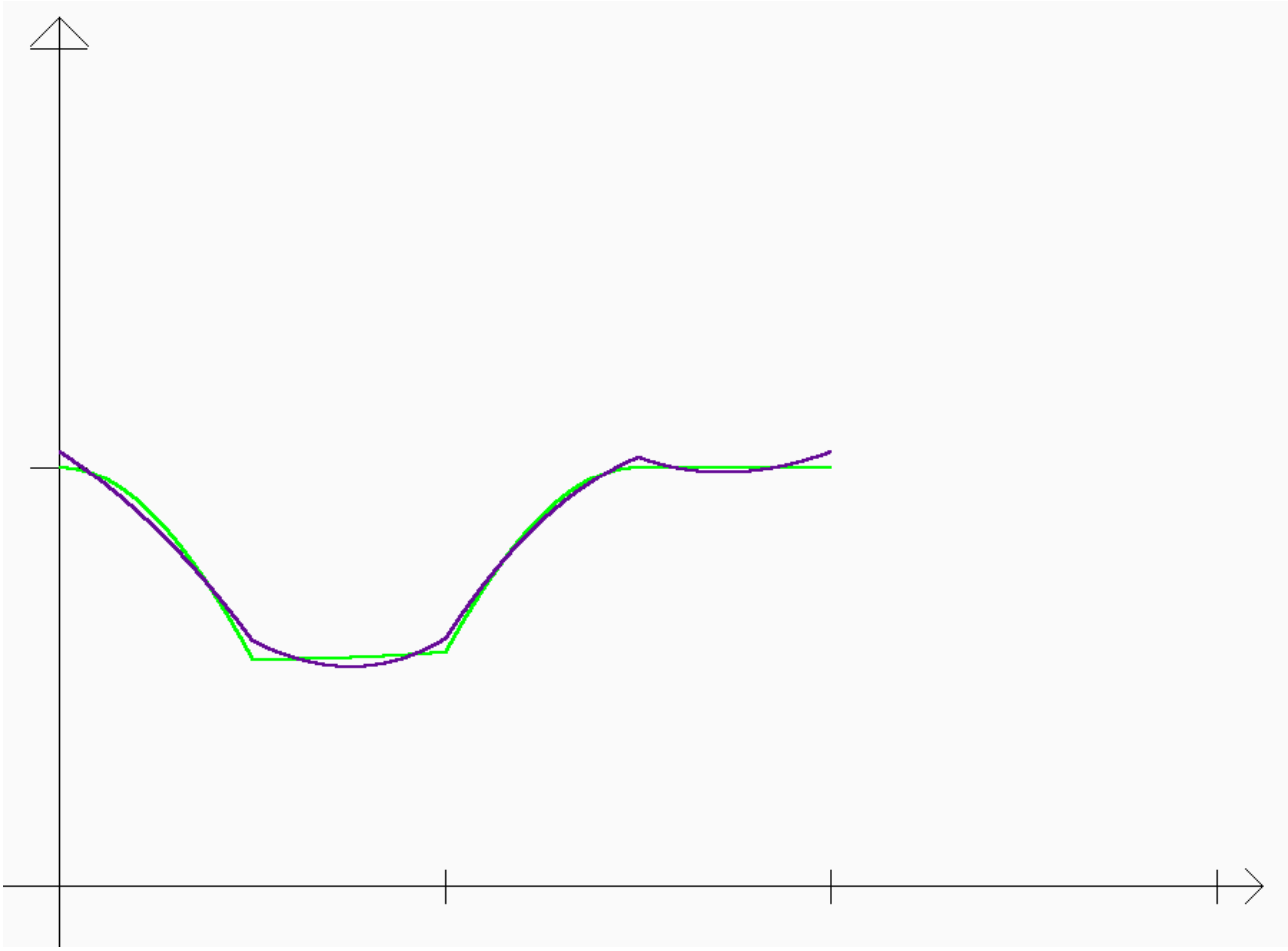

Рисунок 2.10 График приближённого решения для 7-й функции при пяти базисных функциях

Для функций с разрывами первого рода приходится брать большее число базисных потенциалов, но точность аппроксимации остаётся высокой:

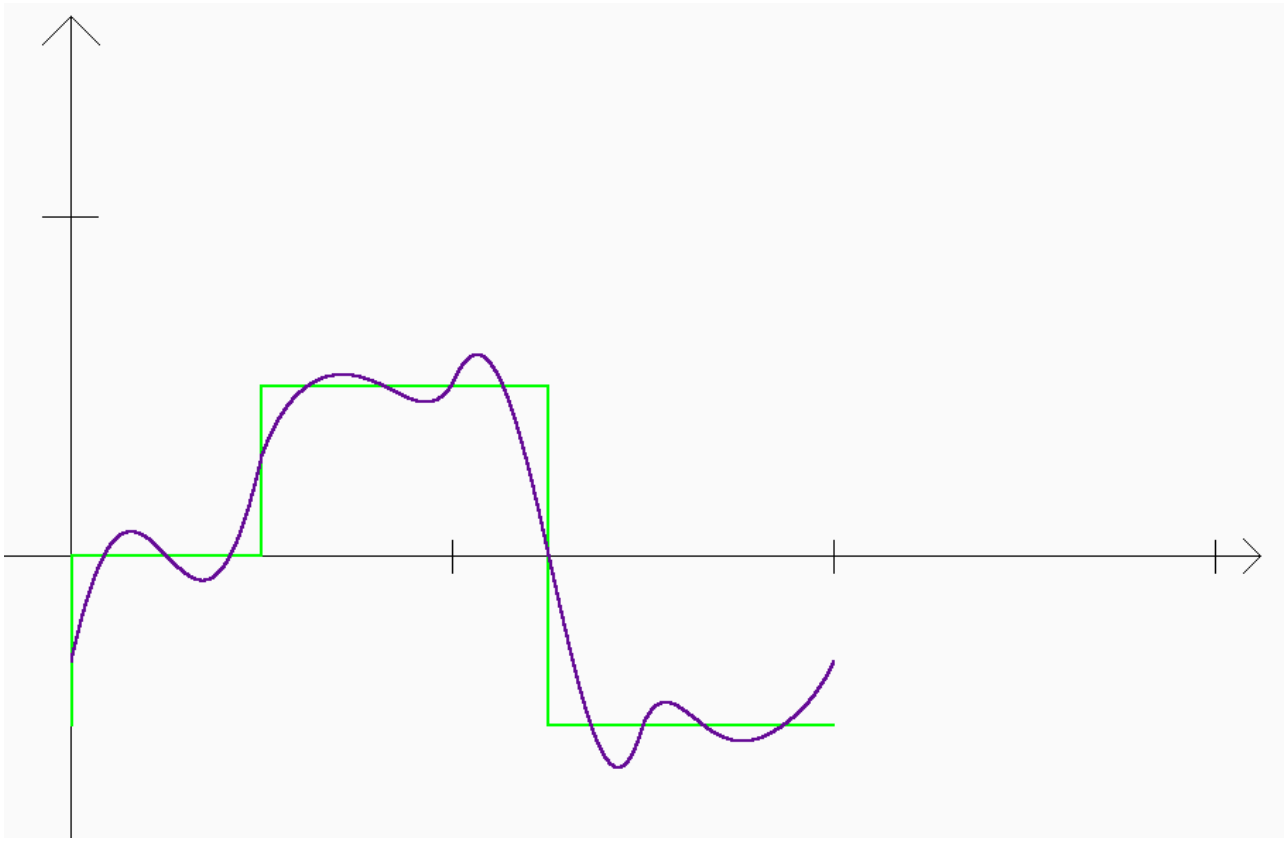

Рисунок 2.11 График приближённого решения для 8-й функции при 13-ти базисных функциях

4.2 Поиск метода решения СЛАУ, при котором приближённое решение покажет наилучшую устойчивость

Решая систему получающихся линейных алгебраических уравнений любым рабочим методом, мы получим достаточно точное решение; однако далеко не все методы обеспечат устойчивое решение исходного дифференциального уравнения, то есть такое решение, которое не сильно изменится при небольшом изменении начальных условий; если же решение любого уравнения окажется неустойчивым, то оно не годится для практических целей. Понять неустойчивость приближённого решения нашего дифференциального уравнения можно по поведению качества аппроксимации в зависимости от количества базисных точек: так как система базисных потенциалов является полной, с ростом числа базисных функций должно происходить монотонное уменьшение погрешности (улучшение аппроксимации), но если решение является неустойчивым, на графике (число базисных потенциалов)-(погрешность) будут наблюдаться скачки вверх.

Руководствуясь целью найти метод решения СЛАУ, обеспечивающий устойчивость решения дифференциального уравнения, мы рассмотрим классические методы Гаусса, Холецкого, Якоби, наискорейшего спуска; возможно, для устойчивого решения задачи Дирихле придётся разрабатывать собственный метод. Каждый из методов показывает настолько хорошую сходимость, что график (число базисных потенциалов)-(погрешность) придётся выводить в логарифмической шкале; задачей является исследование устойчивости перечисленных методов.

4.2.1 Неустойчивость метода Гаусса

Получившаяся система с матрицей Грама хорошо решается методом Гаусса, поскольку является симметрической и имеет небольшое диагональное преобладание, так что нет необходимости даже выбирать ведущие элементы. Именно поэтому даже при 200 базисных функциях метод Гаусса даёт невязку 1.43452e-013. Но решение дифференциального уравнения оказывается всё более и более неустойчивым при увеличении числа базисных функций. Убедиться в этом можно, взглянув на графики 3.1, 3.2, 3.3, 3.4 (берётся решение при входных данных: ∂Q – окружность радиуса 0.5 с центром в начале координат, $\varphi(x, y) = \sin y$, базисные точки берутся вблизи окружности с тем же центром и радиусом 0.7):

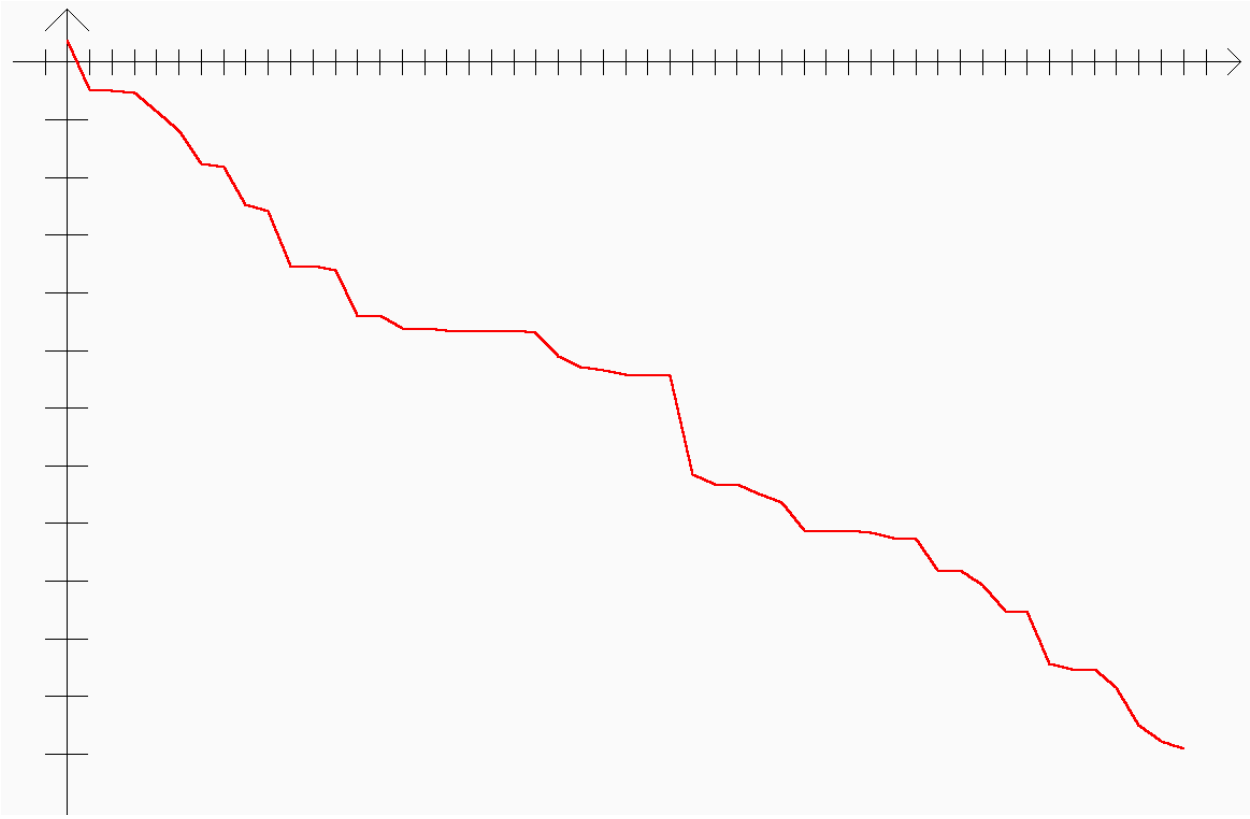

Рисунок 3.1 График зависимости качества аппроксимации от количества базисных функций при использовании метода Гаусса и максимальном числе базисных функций 50

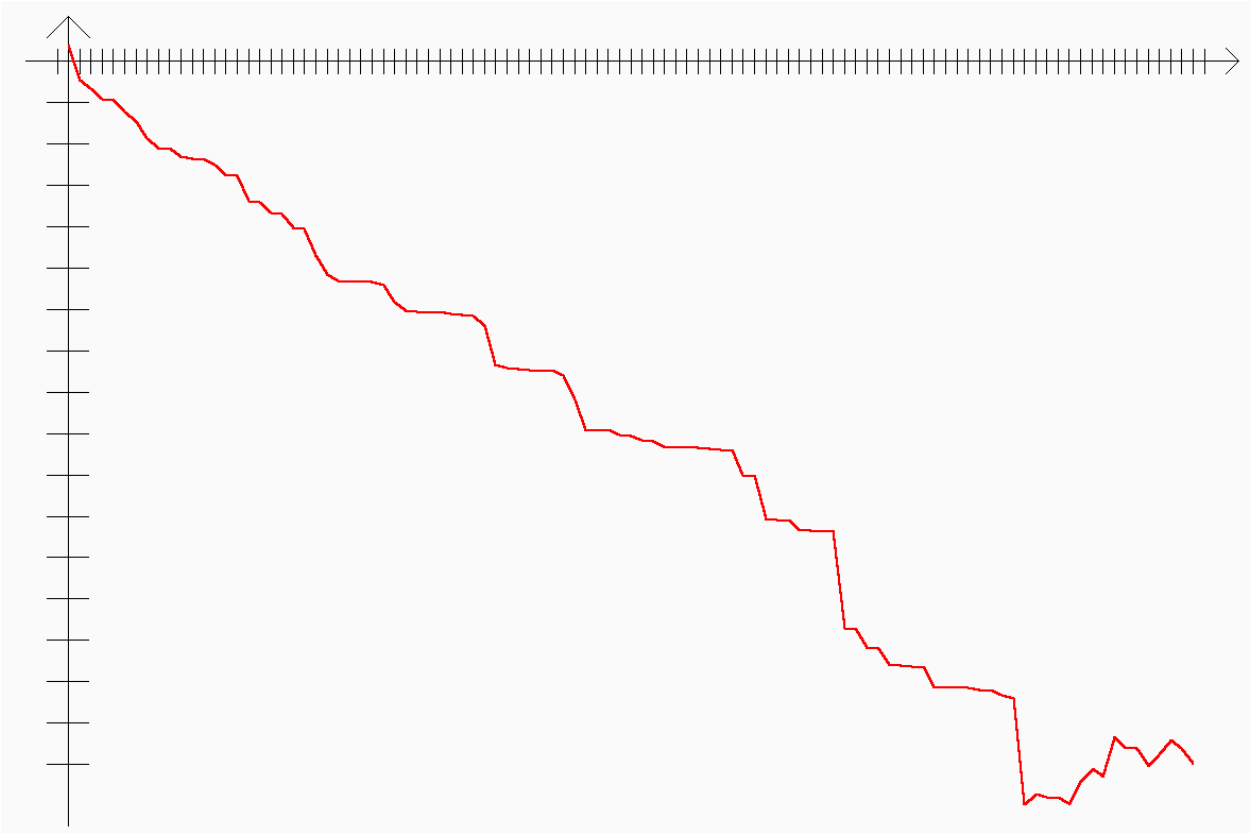

Рисунок 3.2 График зависимости качества аппроксимации от количества базисных функций при использовании метода Гаусса и максимальном числе базисных функций 100

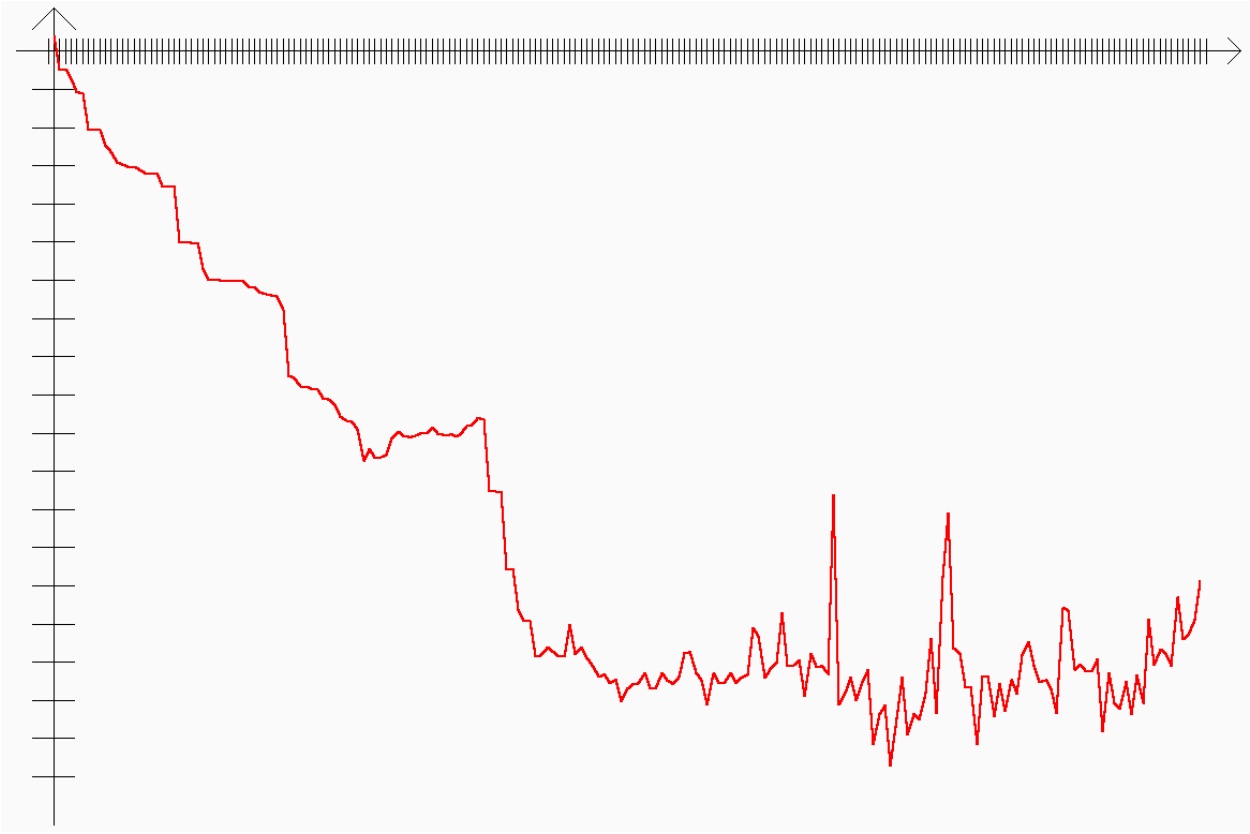

Рисунок 3.3 График зависимости качества аппроксимации от количества базисных функций при использовании метода Гаусса и максимальном числе базисных функций 200

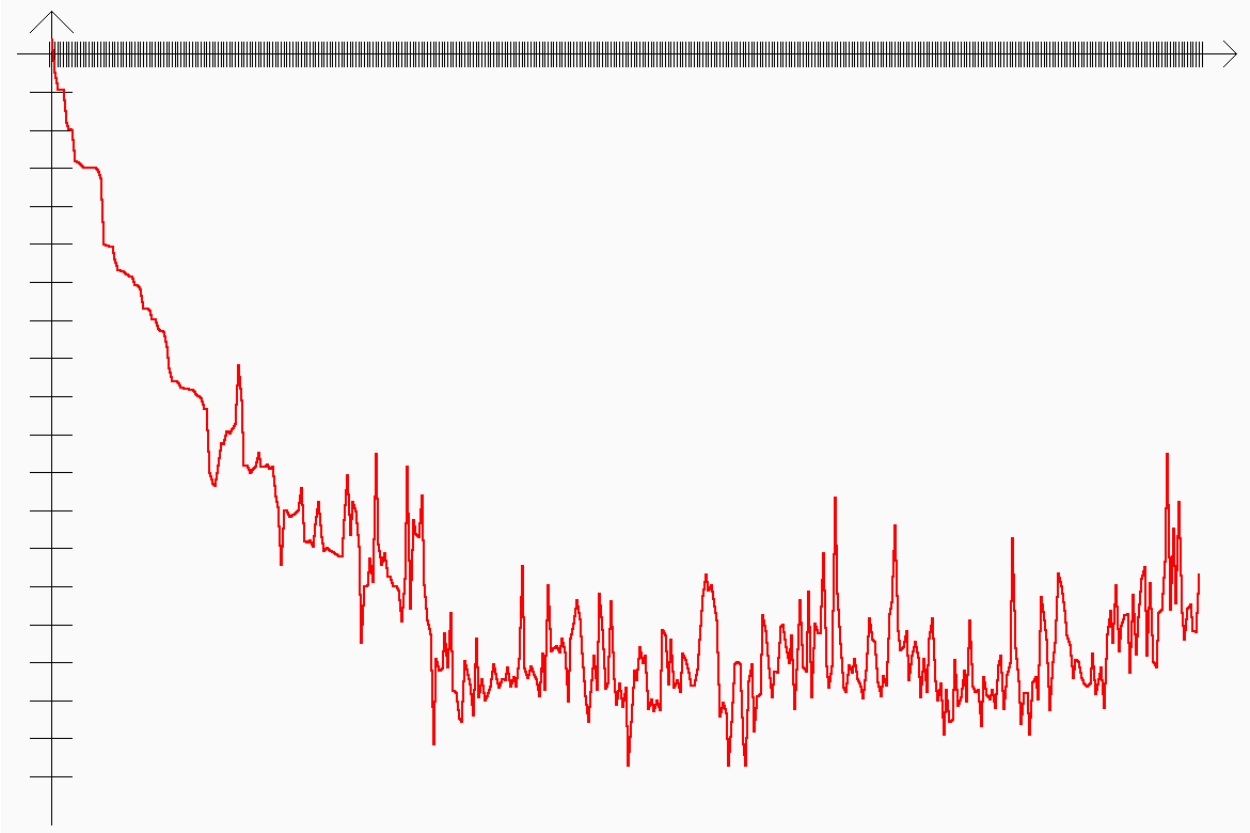

Рисунок 3.4 График зависимости качества аппроксимации от количества базисных функций при использовании метода Гаусса и максимальном числе базисных функций 400

4.2.2 Неустойчивость метода наискорейшего спуска

При точно таких же тестах выясняется неустойчивость (явного) метода наискорейшего спуска при решении нашего дифференциального уравнения. Без углубления в доказательства и выводы формул напомню, что этот метод является итерационным и может быть выражен формулой

$$x^{k+1} = x^k - \frac{(r^k, r^k)}{(Ar^k, r^k)} r^k, x^0 = b$$

где $r^k = Ax^k - b$, b – свободный вектор СЛАУ, A – матрица СЛАУ. При этом итерации выполняются до тех пор, пока не будет достигнута определённая заранее точность, однако для достижения точности, близкой к точности метода Гаусса, при больших размерностях системы метод начинает работать значительно дольше.

Тестирование на тех же кривых и той же самой функции $\varphi(x, y) = \sin y$ показывает, что при увеличении числа базисных функций точность аппроксимации улучшается крайне медленно, при числе базисных функций уже в 150 программа начинает работать в десятки раз дольше, нежели в слу-

чае использования метода Гаусса, а также наблюдается неустойчивость немного другого характера:

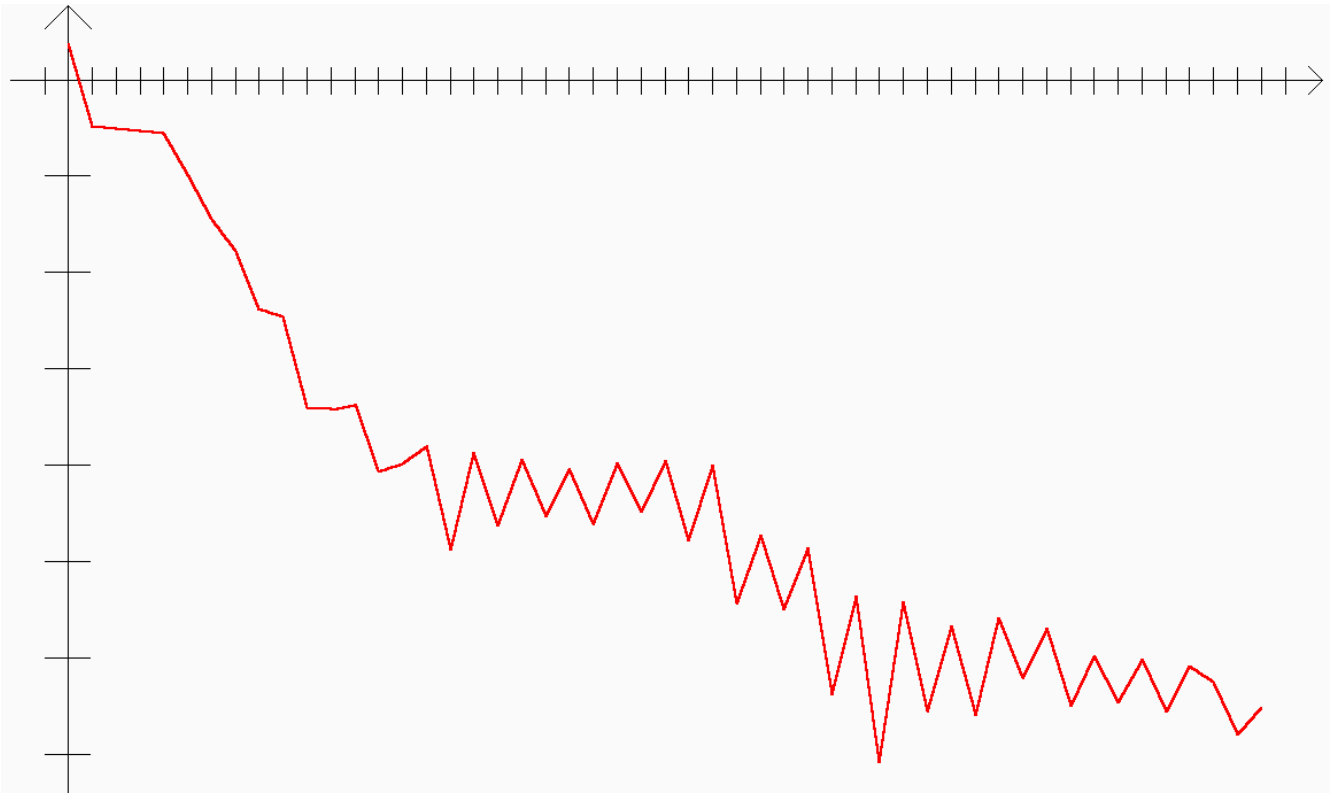

Рисунок 3.5 График зависимости качества аппроксимации от количества базисных функций при использовании метода наискорейшего спуска и максимальном числе базисных функций 50

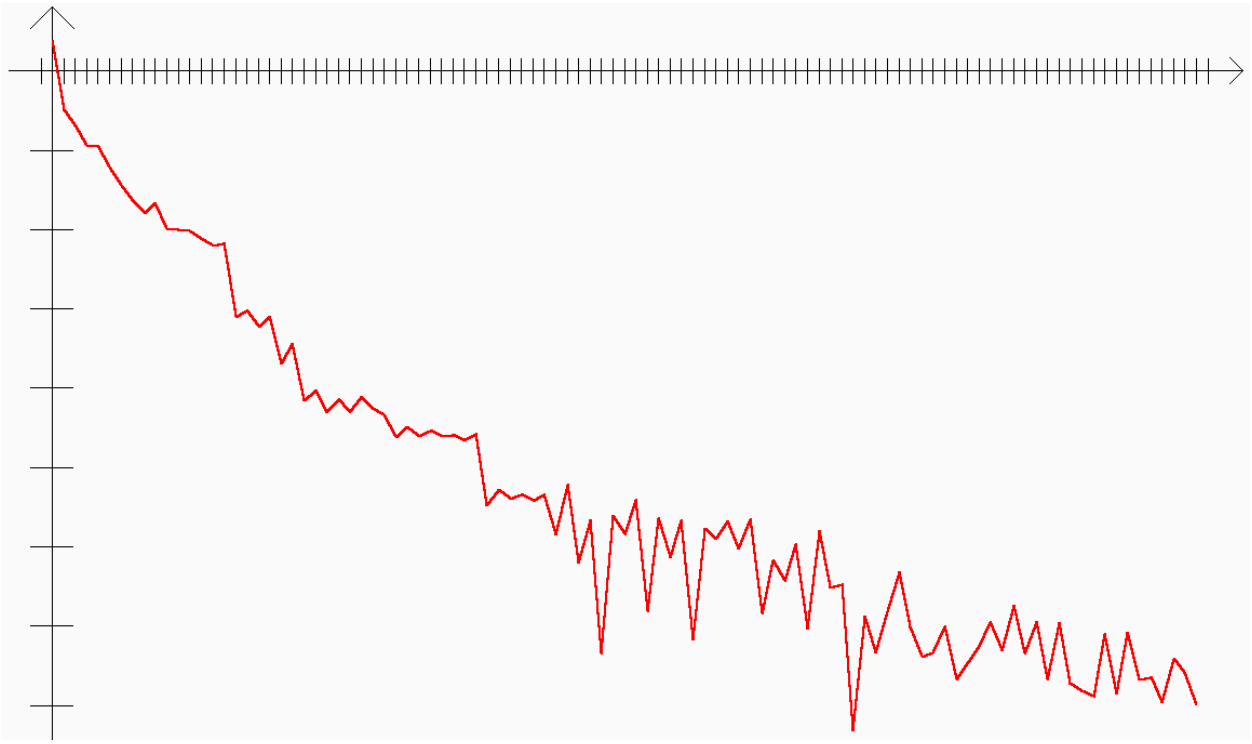

Рисунок 3.6 График зависимости качества аппроксимации от количества базисных функций при использовании метода наискорейшего спуска и максимальном числе базисных функций 100

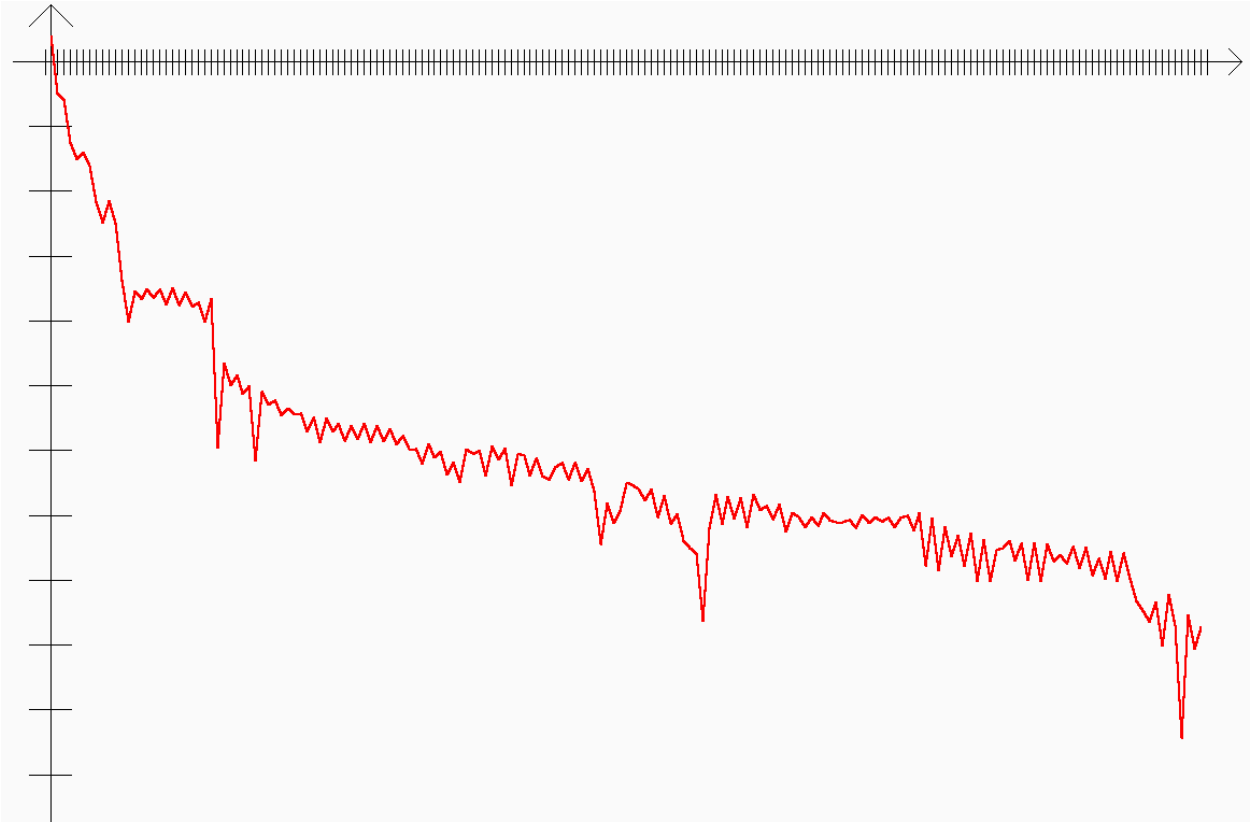

Рисунок 3.7 График зависимости качества аппроксимации от количества базисных функций при использовании метода наискорейшего спуска и максимальном числе базисных функций 180

При методе Якоби ситуация ещё хуже:

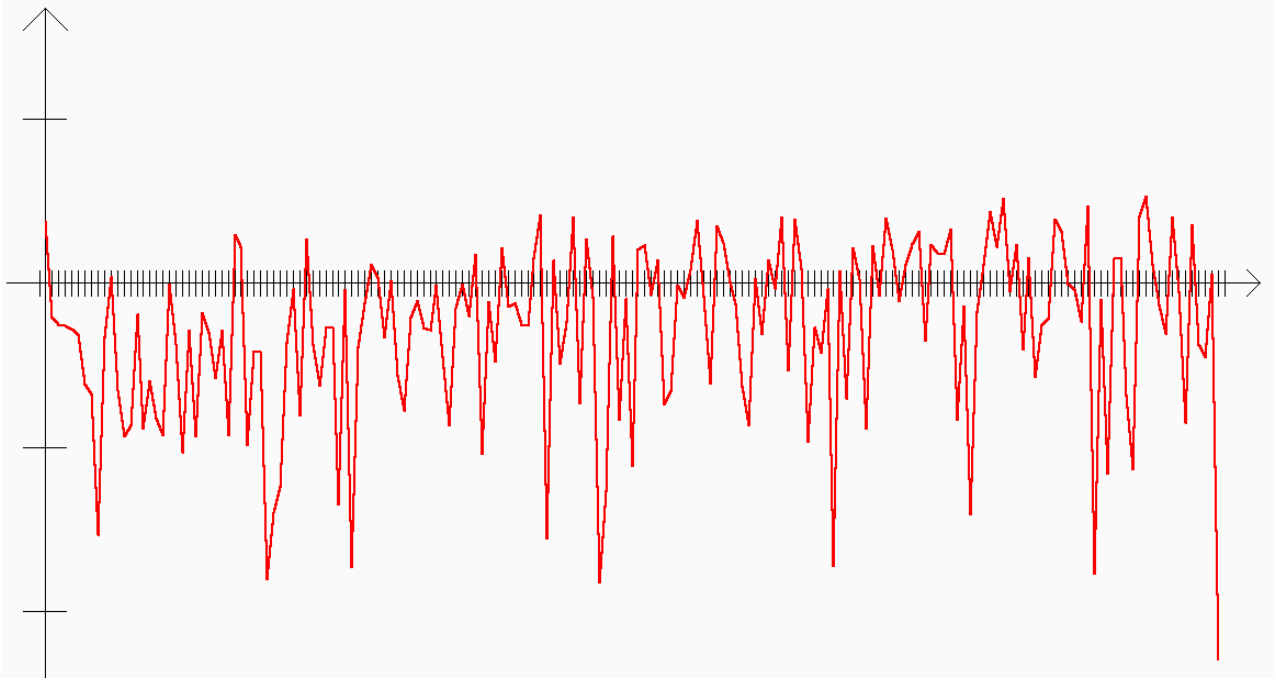

Рисунок 3.8 График зависимости качества аппроксимации от количества базисных функций при использовании метода Якоби и максимальном числе базисных функций 180

4.2.3 Неустойчивость метода Холецкого

Вопреки ожиданиям, метод Холецкого оказался самым худшим из всех рассмотренных. Уже с системы 15×15 он показывает стремительный рост невязки вплоть до бесконечности; для системы 60×60 невязка равна 19664.9, для системы 90×90 – уже 7.33145e+013; возможно, это связано с тем, что при росте размерности системы в матрице Грама элементы становятся всё более похожими друг на друга и матрица приближается к вырожденной.

Ни на какую устойчивость подобного метода, судя по росту невязки, и надеяться не следует: и метод показывает неустойчивость уже на первых шагах, а после 40-ка базисных функций качество аппроксимации начинает стремиться к бесконечности (рисунки 3.9, 3.10).

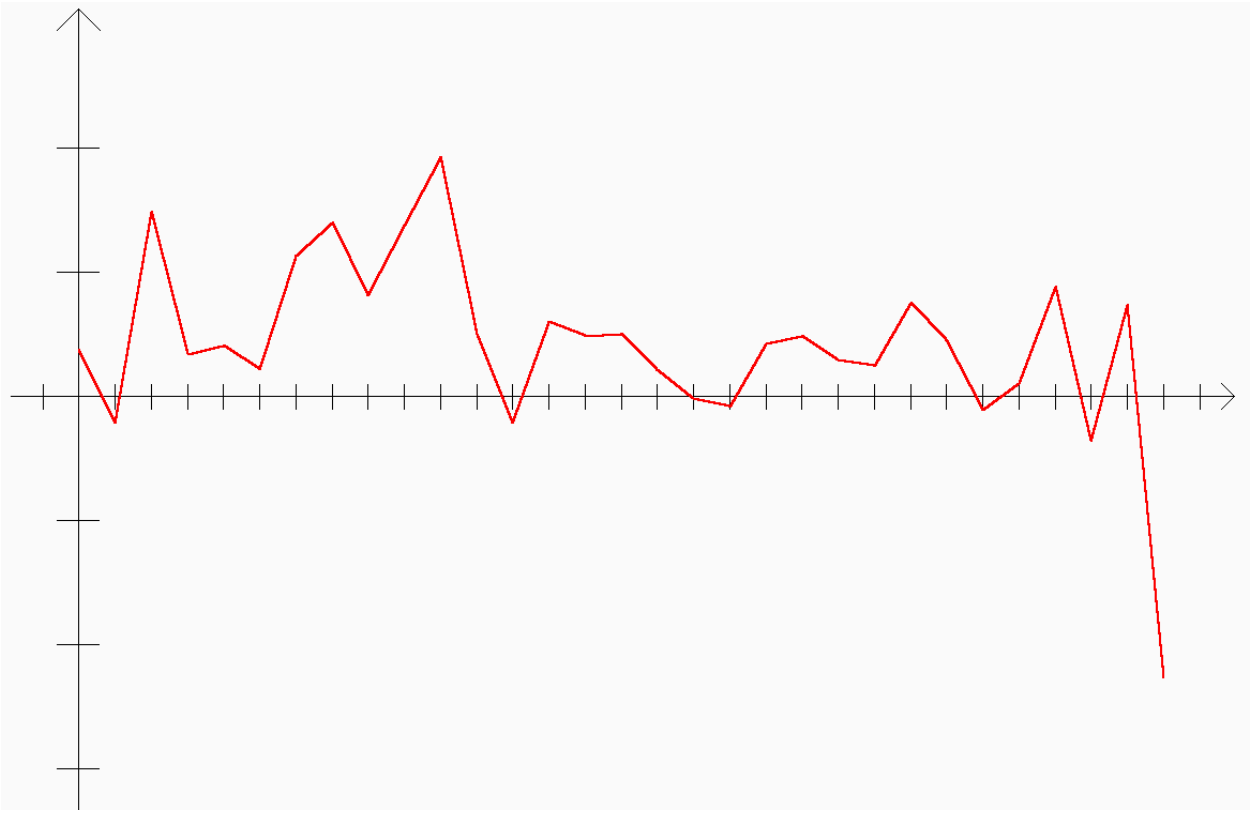

Рисунок 3.9 График зависимости качества аппроксимации от количества базисных функций при использовании метода Холецкого и максимальном числе базисных функций 30

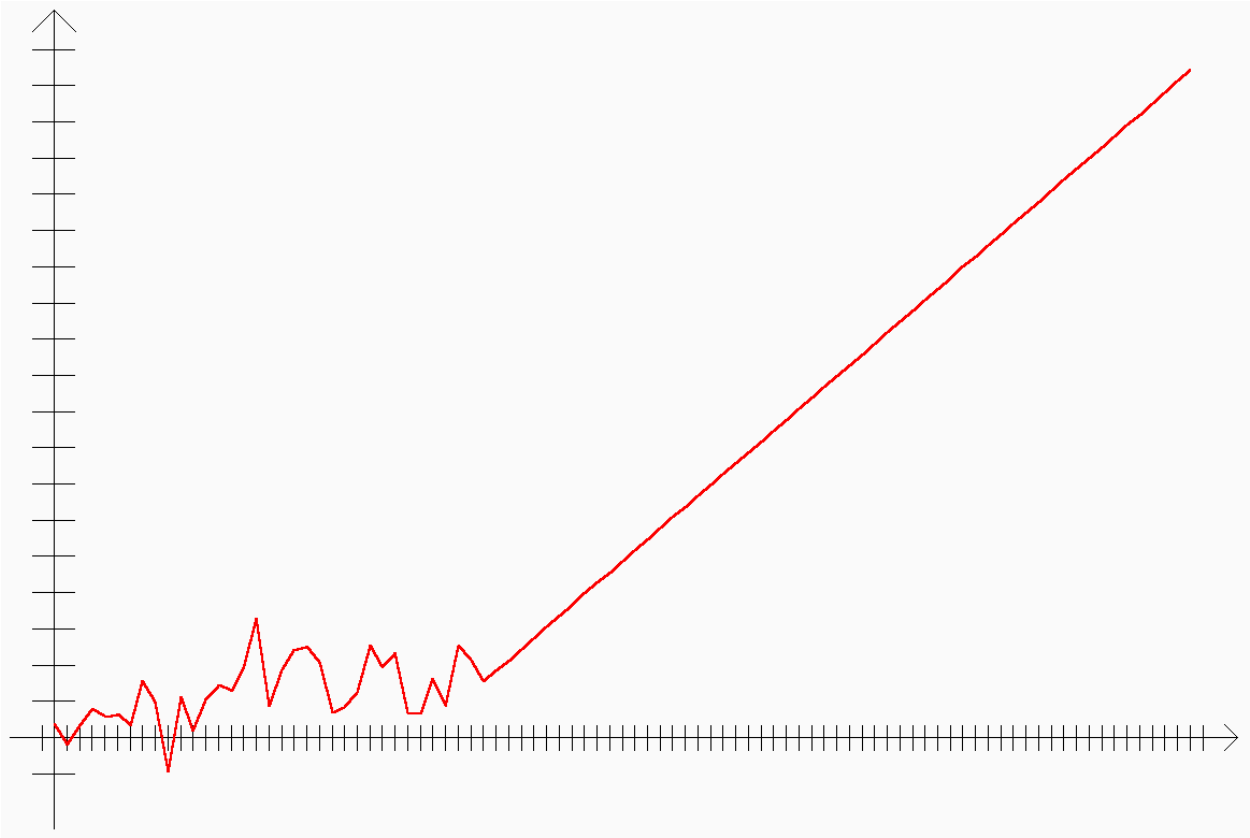

Рисунок 3.10 График зависимости качества аппроксимации от количества базисных функций при использовании метода Холецкого и максимальном числе базисных функций 90

4.2.4 Неустойчивость гибридного метода

При реальном решении на ЭВМ систем линейных уравнений любым методом неизбежно возникают ошибки округления, которые приводят к тому, что полученное решение оказывается не совсем точным; например, при использовании метода Гаусса невязка (погрешность) при числе базисных функций 300 равна 3.1942e-013 (что, впрочем, достаточно мало); в том же случае метод Холецкого показывает невязку (ожидаемо) 8.25305e+077; при этом методу наискорейшего спуска для достижения невязки как в методе Гаусса требуется 5-10 минут работы. Ясно, что из неточного решения системы никак не получится максимально точное приближение, поэтому естественно предположить, что и всякие «неустойчивости» получаемого приближения имеют связь именно с неточными решениями системы, а потому и устранение этих неточностей должно положительно сказаться на качестве аппроксимации.

Требуется узнать поведение качества аппроксимации при максимально возможном приближении. Так как метод Гаусса для получающейся матрицы Грама уже сам по себе показывается достаточно малую невязку, то не нужно испытывать новые методы, а следует лишь дополнить метод Гаусса так, чтобы повысить его точность до машинного нуля. Это легко сделать, просто взяв за начальное приближение системы её решение методом Гаусса, из которого методом наискорейшего спуска будут производиться итерации до тех пор, пока невязка не станет равной «нулю». Тестирование показывает, что и при использовании такого метода, который назовём гибридным, неустойчивость сохраняется, что видно из рисунков 3.11, 3.12, 3.13 (взяты те же данные, что и предыдущих пунктах):

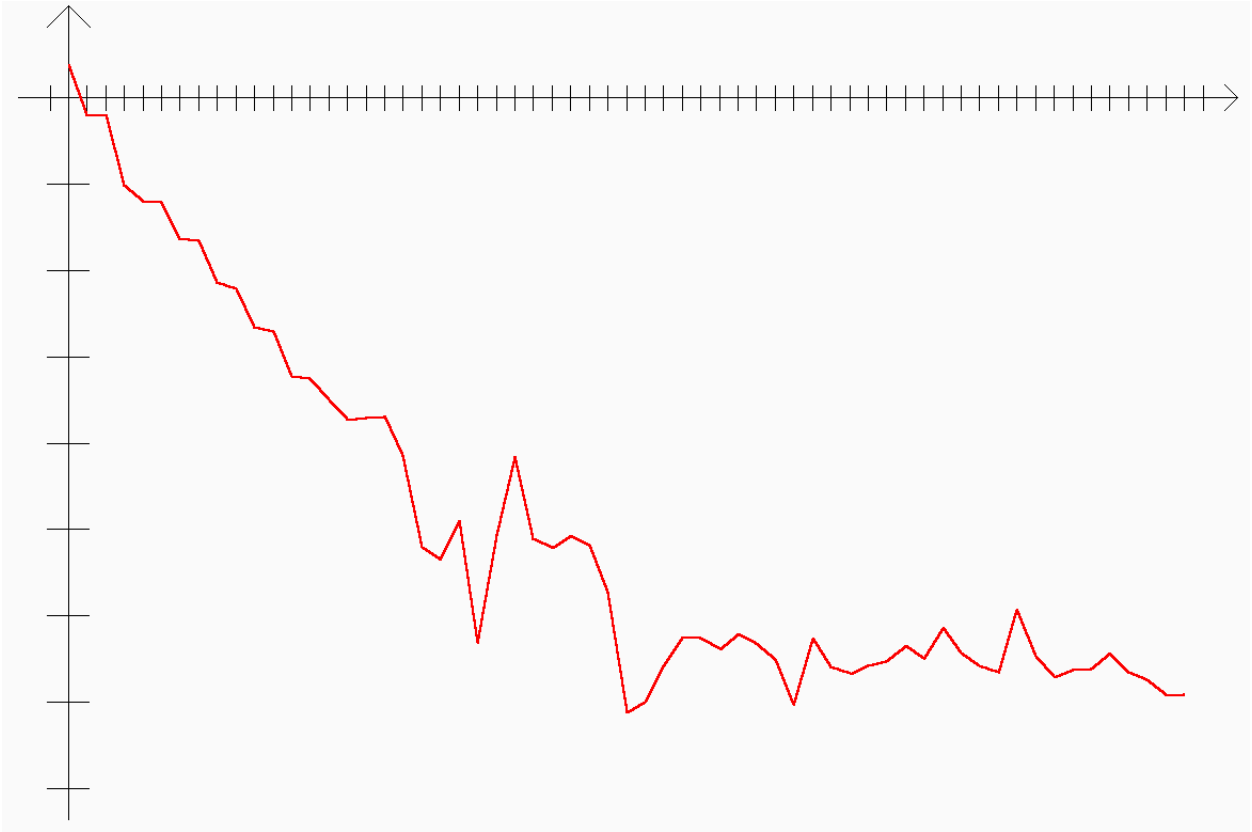

Рисунок 3.11 График зависимости качества аппроксимации от количества базисных функций при использовании гибридного метода и максимальном числе базисных функций 60

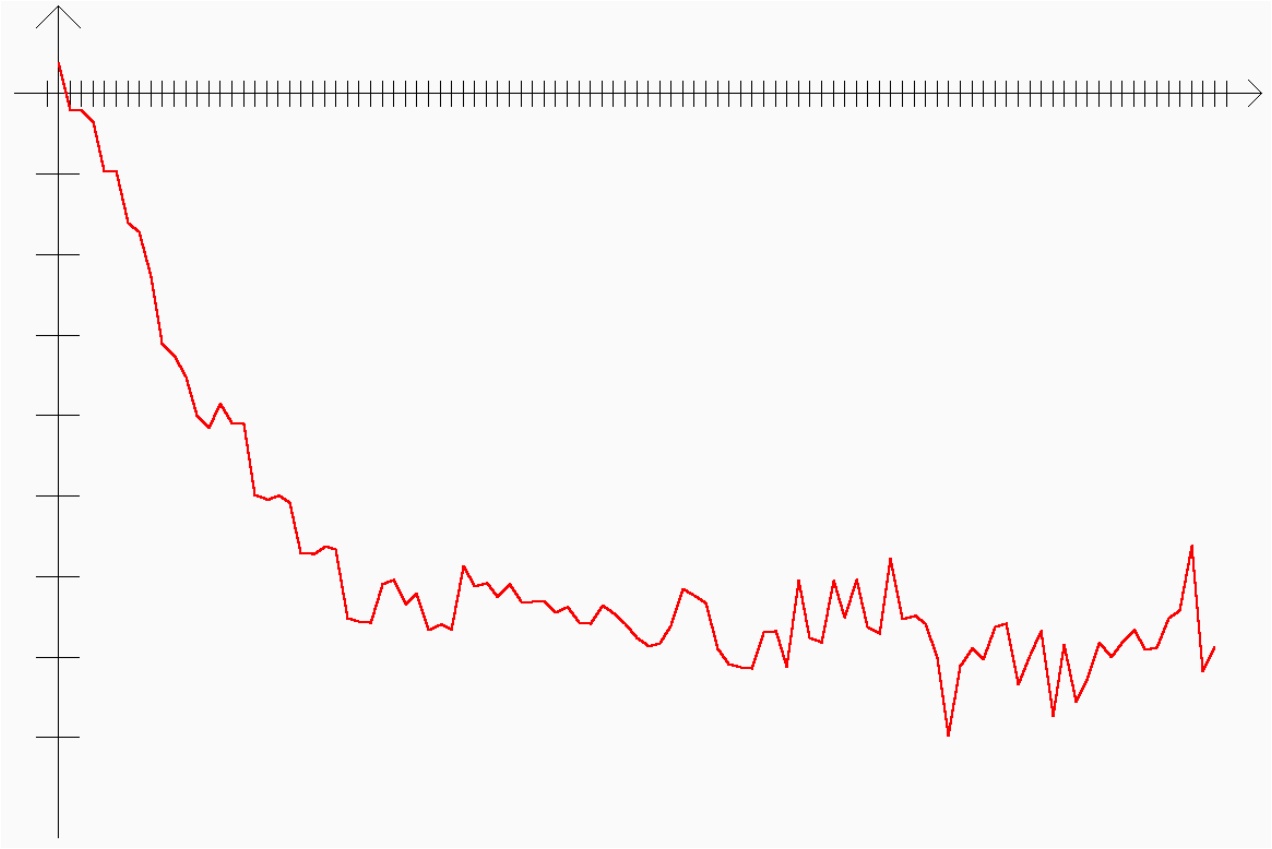

Рисунок 3.12 График зависимости качества аппроксимации от количества базисных функций при использовании гибридного метода и максимальном числе базисных функций 100

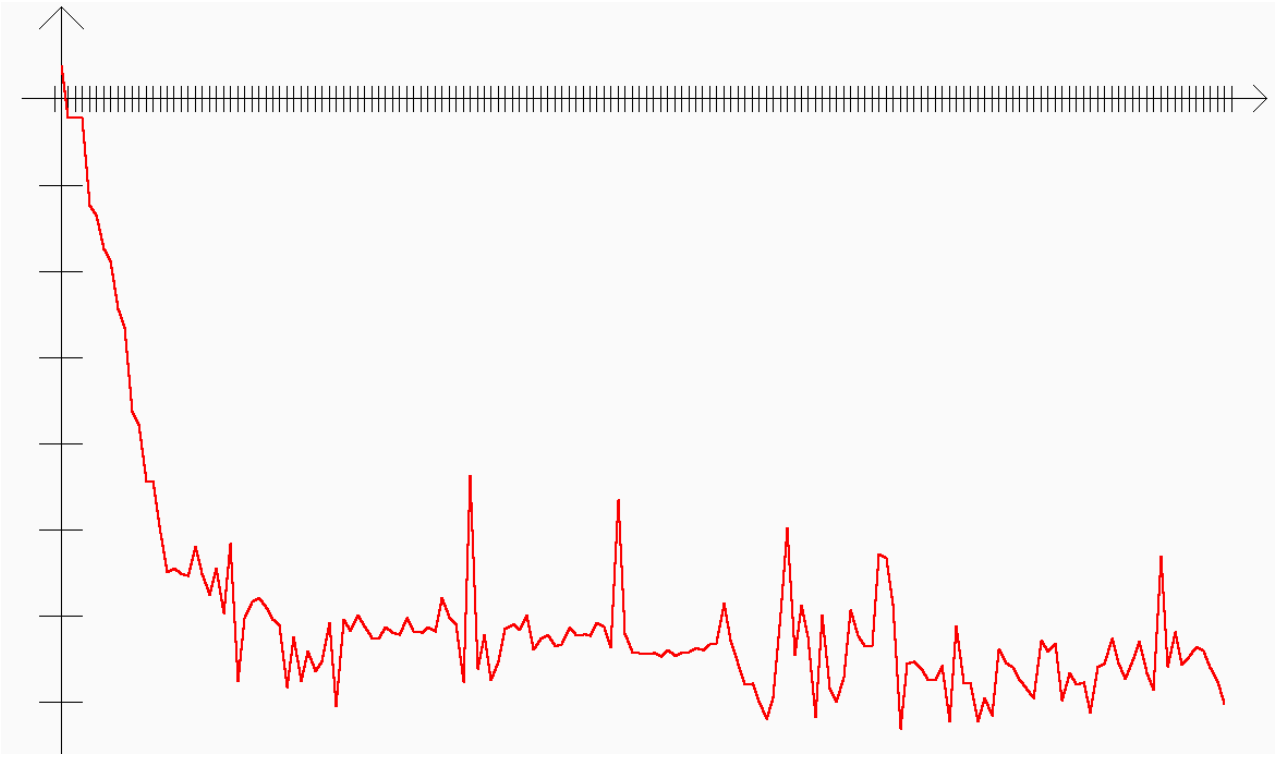

Рисунок 3.13 График зависимости качества аппроксимации от количества базисных функций при использовании гибридного метода и максимальном числе базисных функций 160

Если сравнивать между собой гибридный метод и метод Гаусса, то из рисунка 3.14 видно, что поначалу они оба обеспечивают одинаковую аппроксимацию, затем одновременно начинают показывать неустойчивость, причём примерно в равном количестве случаев гибридный метод аппроксимирует лучше метода Гаусса и наоборот.

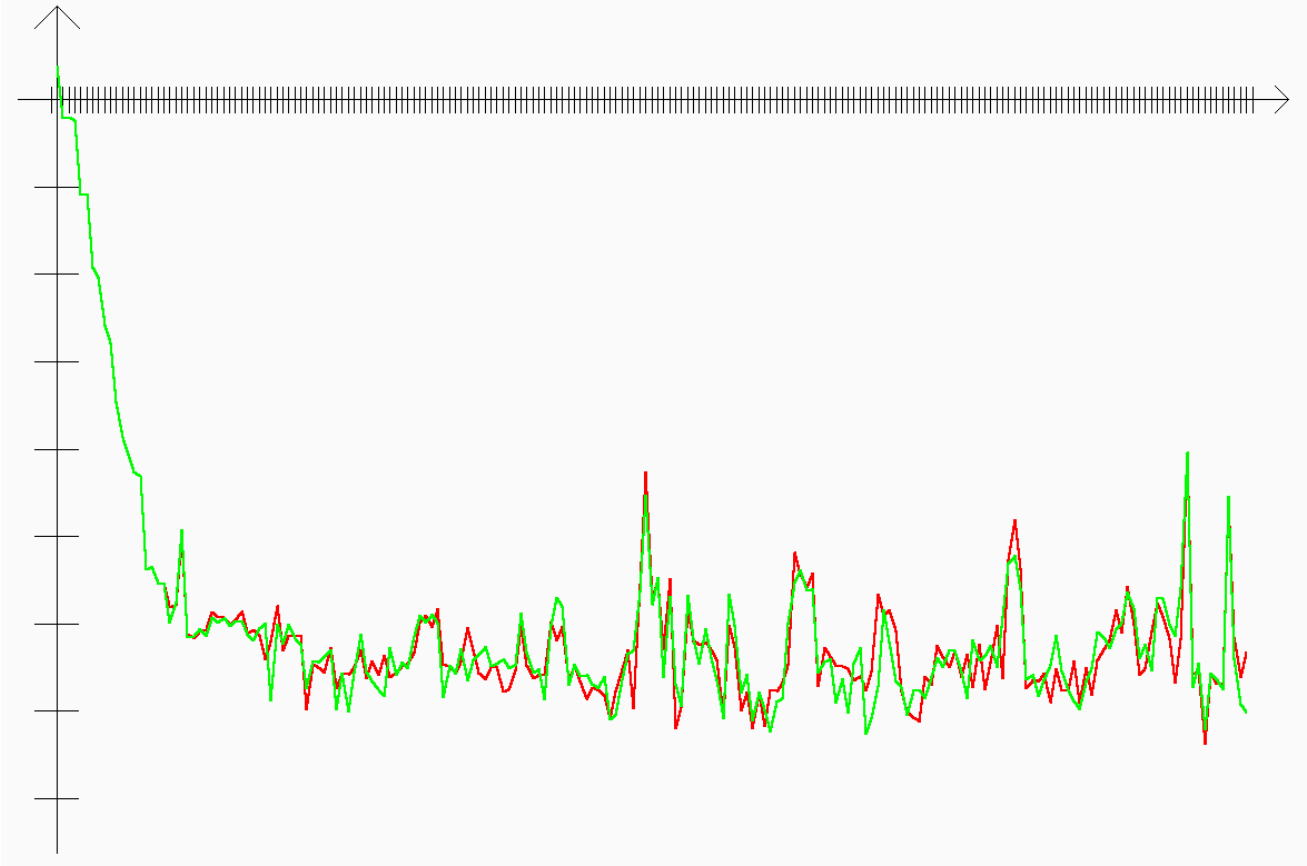

Рисунок 3.14 Сравнительный график зависимости качества аппроксимации от количества базисных функций при использовании гибридного метода (зелёный) и метода Гаусса (красный) и максимальном числе базисных функций 200

Если в качестве набора $c_m, m = 1, 2, \dots, M$ взять такой из решений СЛАУ методом Гаусса и гибридным методом, при котором качество аппроксимации лучше, то неустойчивость всё равно останется значительной (рисунок 3.15).

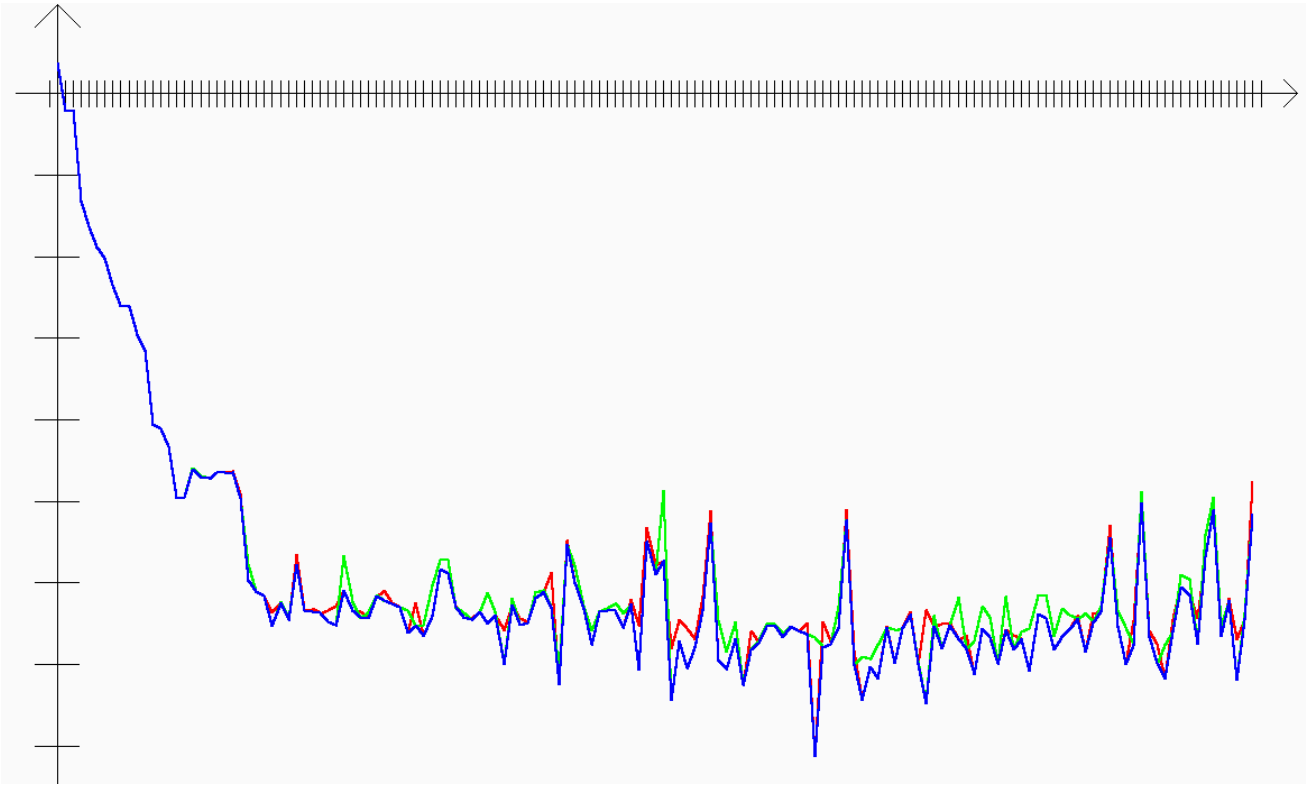

Рисунок 3.15 Сравнительный график зависимости качества аппроксимации от количества базисных функций при использовании гибридного метода (зелёный) и метода Гаусса (красный) и максимальном числе базисных функций 150. Синим цветом обозначен минимум по результатам двух предыдущих методов

Отсюда можно сделать вывод, что даже при максимально точном решении СЛАУ в пределах машинных возможностей минимизация функции

$$F(c) = \left\| \varphi - \sum_{m=1}^{N} c_m \, \alpha_m \right\|_{L_2(\partial Q)}$$

даёт сбои, поскольку результаты тестирования показывают, что при многих значениях $M < N$ имеет место неравенство

$$\left\| \varphi - \sum_{m=1}^{M} c_m \, \alpha_m \right\|_{L_2(\partial Q)} < \left\| \varphi - \sum_{m=1}^{M+1} c_m \, \alpha_m \right\|_{L_2(\partial Q)}$$

хотя вполне очевиден такой набор коэффициентов $c_m, m = 1, 2, \dots, M+1$, обеспечивающий аппроксимацию лучше того набора, который находится программой, а именно:

$$\left\| \varphi - \sum_{m=1}^{M} c_m \alpha_m - 0 \times \alpha_{M+1} \right\|_{L_2(\partial Q)} = \left\| \varphi - \sum_{m=1}^{M} c_m \alpha_m \right\|_{L_2(\partial Q)}$$

$$< \left\| \varphi - \sum_{m=1}^{M+1} c_m \alpha_m \right\|_{L_2(\partial Q)}$$

Значит, если при максимально возможном точном решении СЛАУ неустойчивость остаётся, то единственным оставшимся выходом для её устранения может быть только варьирование получившихся коэффициентов в таком направлении, чтобы аппроксимация стала минимальной. Теория при этом говорит, что получившаяся последовательность приближений должна, по крайней мере, не возрастать.

4.2.5 Попытка использования метода покоординатного спуска

Допустим, гибридным методом был найден набор чисел $c_m, m = 1, 2, \dots, N$, являющийся максимально точным решением получившейся СЛАУ; однако, в самом деле, этот набор не обязан обеспечивать максимально хорошую аппроксимацию хотя бы из-за неизбежных неточностей при вычислении скалярных произведений (интегралов), то есть решение системы

$$\begin{pmatrix} (\alpha_1, \alpha_1)_{L_2(\partial Q)} & \cdots & (\alpha_1, \alpha_N)_{L_2(\partial Q)} \\ \vdots & \ddots & \vdots \\ (\alpha_N, \alpha_1)_{L_2(\partial Q)} & \cdots & (\alpha_N, \alpha_N)_{L_2(\partial Q)} \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_N \end{pmatrix} = \begin{pmatrix} (\varphi, \alpha_1)_{L_2(\partial Q)} \\ \vdots \\ (\varphi, \alpha_N)_{L_2(\partial Q)} \end{pmatrix}$$

является максимально точным, но сама система лишь приближённо является той, какой должна быть. Отсюда можно сделать выводы, что некоторая корректировка набора $c_m, m = 1, 2, \dots, N$ оказывается не только полезной, но и обязательной, если при аппроксимации пытаться добиться максимальной точности и устойчивости; это понятно и из вывода, сделанного в предыдущем пункте. Можно осуществить такую корректировку при помощи метода покоординатного спуска, который состоит в следующем ([15]).

Найденный вектор $(c_1, c_2, \dots, c_N)$ берётся за начальное приближение; после этого для каждого $c_k, 1 \leq k \leq N$ последовательно выполняется задача минимизации функции

$$F(c_k) = \left\| \varphi - \sum_{m=1,m\neq k}^{N} c_m\,\alpha_m - c_k\alpha_k \right\|_{L_2(\partial Q)}$$

$$= \left(\varphi - \sum_{m=1,m\neq k}^{N} c_m\,\alpha_m - c_k\alpha_k, \varphi - \sum_{m=1,m\neq k}^{N} c_m\,\alpha_m - c_k\alpha_k \right)^{\frac{1}{2}}_{L_2(\partial Q)}$$

$$= \Bigg((\varphi,\varphi) - 2\sum_{m=1,m\neq k}^{N} c_m\,(\varphi,\alpha_m)$$

$$+ \sum_{n=1,n\neq k}^{N} c_n \sum_{m=1,m\neq k}^{N} c_m\,(\alpha_n,\alpha_m) + 2\sum_{m=1,m\neq k}^{N} c_m\,c_k(\alpha_k,\alpha_m)$$

$$- 2c_k(\alpha_k,\varphi) + c_k^2(\alpha_k,\alpha_k) \Bigg)^{\frac{1}{2}}$$

при этом полученный в конце новый вектор $(c_1, c_2, \dots, c_N)$ теоретически гарантировано обеспечит лучшую аппроксимацию, а потому и устойчивость (устранив «скачки»).

Итак, реализуем задачу минимизации функции

$$F(c_k)^2 = (\varphi,\varphi) - 2\sum_{m=1,m\neq k}^{N} c_m\,(\varphi,\alpha_m) + \sum_{n=1,n\neq k}^{N} c_n \sum_{m=1,m\neq k}^{N} c_m\,(\alpha_n,\alpha_m)$$

$$+ 2\sum_{m=1,m\neq k}^{N} c_m\,c_k(\alpha_k,\alpha_m) - 2c_k(\alpha_k,\varphi) + c_k^2(\alpha_k,\alpha_k)$$

(возведение в квадрат не повлияет на решения, поскольку функция принимает исключительно неотрицательные значения). Стационарные точки:

$$D_{c_k}F(c_k) = 0 \leftrightarrow 2\sum_{m=1,m\neq k}^{N} c_m\,(\alpha_k,\alpha_m) - 2(\alpha_k,\varphi) + 2c_k(\alpha_k,\alpha_k) = 0 \rightarrow c_k$$

$$= \frac{(\alpha_k,\varphi) - \sum_{m=1,m\neq k}^{N} c_m\,(\alpha_k,\alpha_m)}{(\alpha_k,\alpha_k)}$$

Очевидно, что найденная стационарная точка является точкой минимума, поскольку при переходе через неё производная функции меняет знак с отрицательного на положительный

$$D_{c_k}F(c_k \pm \varepsilon) = 2\sum_{m=1,m\neq k}^{N} c_m\,(\alpha_k, \alpha_m) - 2(\alpha_k, \varphi) + 2(c_k \pm \varepsilon) \times (\alpha_k, \alpha_k)$$

$$= \pm 2\varepsilon(\alpha_k, \alpha_k) = \pm\omega, \omega > 0$$

Однако, при тестировании метода покоординатного спуска выяснилось, что его использование вовсе необязательно минимизирует функцию, если в точке начального приближения её значения уже близки к нулю. Иными словами, если за начальное приближение взять результат работы какого-либо предыдущего метода (то есть вектор $(c_1, c_2, \dots, c_N)$ такой, что $0 < F(c) \leq \varepsilon$), то для получающегося вектора $\tau = (\tau_1, \tau_2, \dots, \tau_N)$ вовсе не обязательно будет выполняться $0 < F(\tau) \leq F(c)$, даже после многократного применения метода покоординатной минимизации (рисунок 3.16); иногда, впрочем, случается, что $F(\tau) \leq F(c)$, но при этом не никакой речи о каком-то монотонном улучшении качества аппроксимации не идёт, откуда и не получится устойчивости.

```
Точность аппроксимации при числе функций 145 =
до использования покоординатной минимизации:    2.90477e-007
после использования покоординатной минимизации 1 раз:   3.73571e-007
после использования покоординатной минимизации 2 раз:   1.97405e-007
после использования покоординатной минимизации 3 раз:   3.87e-007
после использования покоординатной минимизации 4 раз:   5.81909e-008
после использования покоординатной минимизации 5 раз:   1.49384e-007
Точность аппроксимации при числе функций 146 =
до использования покоординатной минимизации:    5.9614e-007
после использования покоординатной минимизации 1 раз:   2.88656e-007
после использования покоординатной минимизации 2 раз:   3.47072e-007
после использования покоординатной минимизации 3 раз:   4.78116e-007
после использования покоординатной минимизации 4 раз:   6.6989e-007
после использования покоординатной минимизации 5 раз:   4.97351e-007
Точность аппроксимации при числе функций 147 =
до использования покоординатной минимизации:    6.40923e-008
после использования покоординатной минимизации 1 раз:   1.90828e-007
после использования покоординатной минимизации 2 раз:   1.666e-007
после использования покоординатной минимизации 3 раз:   1.07712e-007
после использования покоординатной минимизации 4 раз:   4.59949e-007
после использования покоординатной минимизации 5 раз:   4.0604e-007
Точность аппроксимации при числе функций 148 =
до использования покоординатной минимизации:    4.23898e-007
после использования покоординатной минимизации 1 раз:   2.75973e-007
после использования покоординатной минимизации 2 раз:   1.33488e-007
```

Рисунок 3.16 Зависимость качества аппроксимации до и после многократного использования метода покоординатной минимизации. Из рисунка видно, что многократное использования покоординатной минимизации не только не обеспечивает монотонное убывание, но и само убывание не гарантирует

4.2.6 Устойчивый метод

Модификация и слияние нескольких описанных ранее методов привела к созданию устойчивого метода решение исходного дифференциального уравнения; к слову, он никак не привязан к системе базисных потенциалов и должен эффективно работать для любой полной системы функций. Назовём этот метод ультра-гибридным. Вот в чём он заключается.

Допустим, требуется минимизировать функционал

$$F(c^M) = \left\| \varphi - \sum_{m=1}^{M} c_m \, \alpha_m \right\|_{L_2(\partial Q)}$$

причём его минимизация должна удовлетворять условию

$$F(c^{M-1}) \geq F(c^M) \geq F(c^{M+1}), M \geq 2$$

Такую минимизацию можно осуществить по алгоритму:

Шаг 1. Найти вектор $c^M = (c_1, c_2, \dots, c_M)$, решив известную СЛАУ описанным ранее гибридным методом.

Шаг 2. Если $F(c^{M-1}) < F(c^M)$, то (применить покоординатную минимизацию для последнего элемента)

$$c^M = \left(c_1^{M-1}, c_2^{M-1}, \dots, c_{M-1}^{M-1}, \frac{(\alpha_M, \varphi) - \sum_{m=1}^{M-1} c_m (\alpha_M, \alpha_m)}{(\alpha_M, \alpha_M)}\right)$$

Шаг 3. Если и в таком случае $F(c^{M-1}) < F(c^M)$, то

$$c^M = (c_1^{M-1}, c_2^{M-1}, \dots, c_{M-1}^{M-1}, 0) \to F(c^{M-1}) = F(c^M)$$

Аппроксимативные свойства этого метода, очевидно, не хуже, чем у гибридного (рисунок 3.19); при этом устойчивость гарантирована (рисунки 3.17, 3.18).

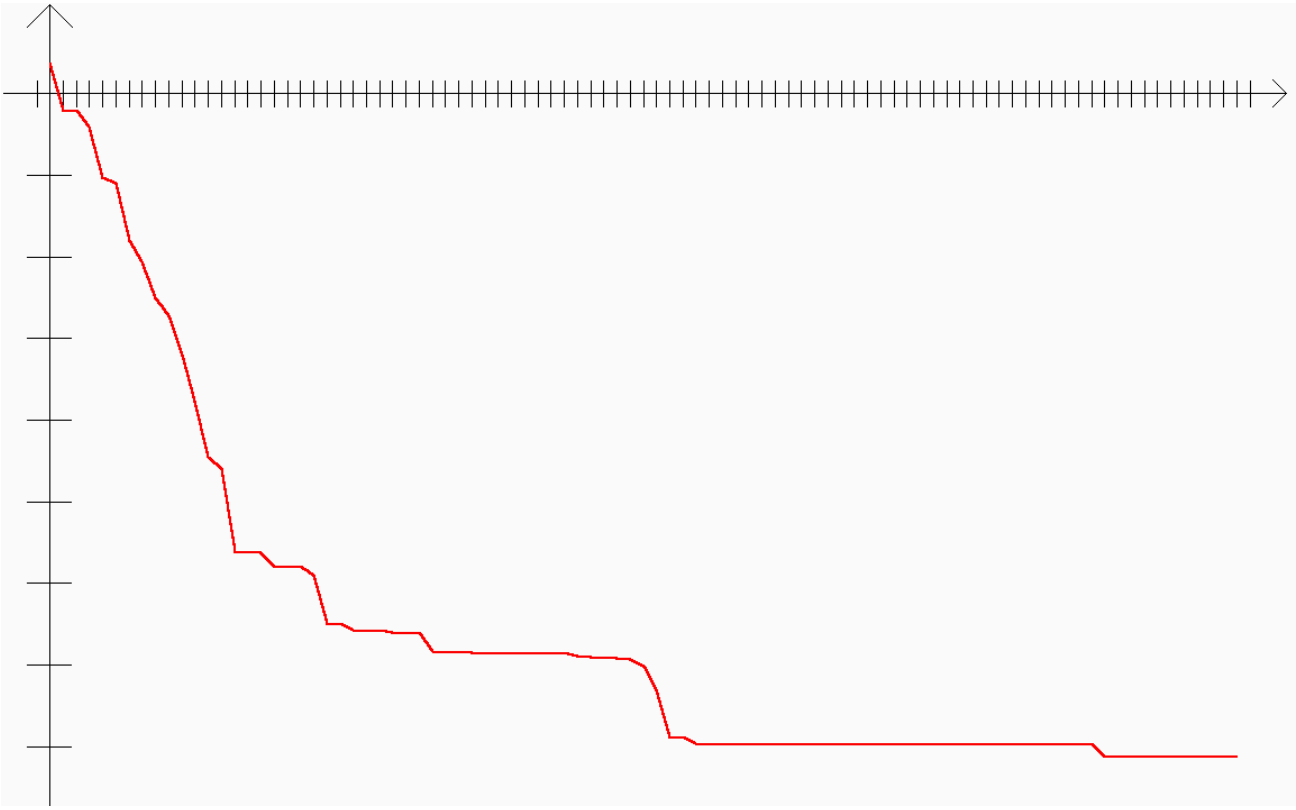

Рисунок 3.17 График зависимости качества аппроксимации граничной функции 2 от количества базисных функций при использовании ультра-гибридного метода и максимальном числе базисных функций 90

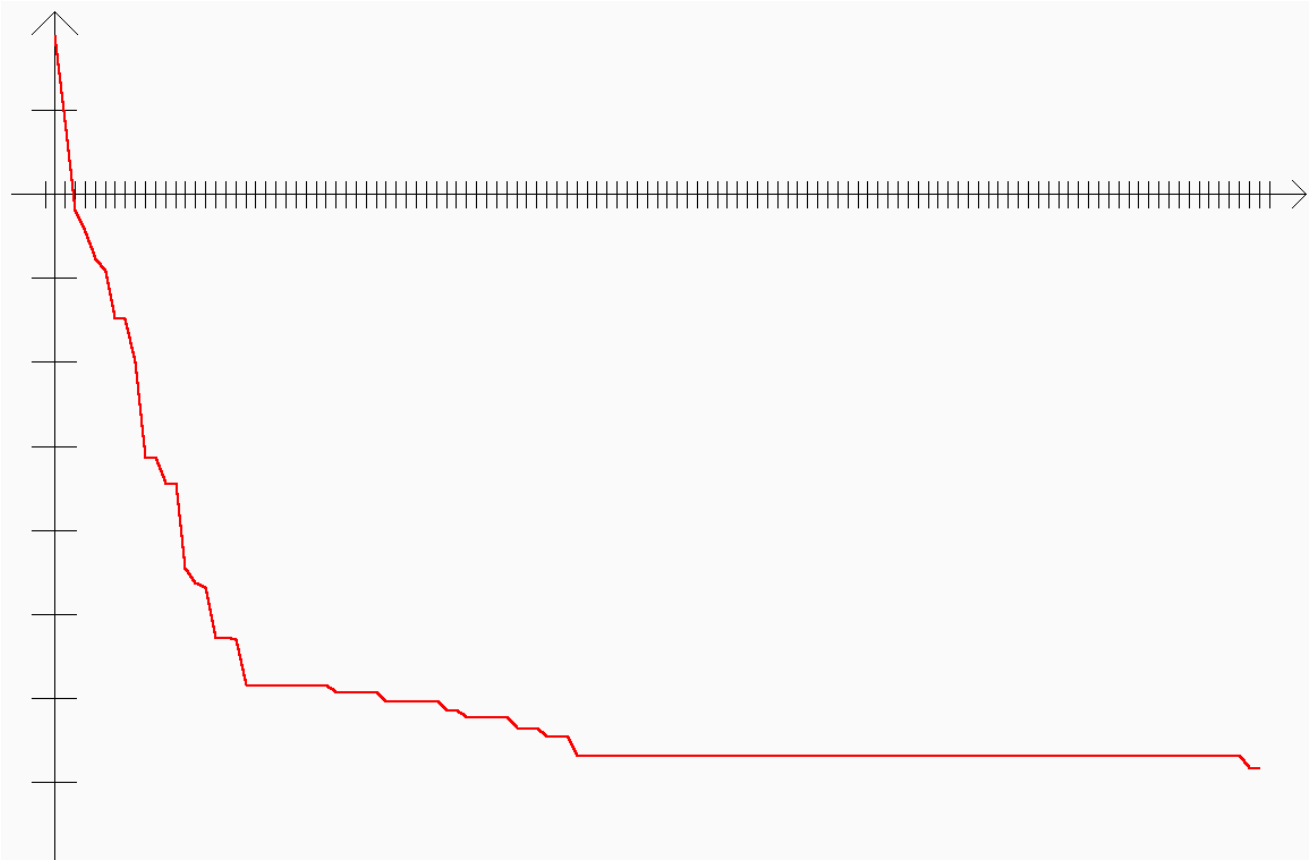

Рисунок 3.18 График зависимости качества аппроксимации граничной функции 3 от количества базисных функций при использовании ультра-гибридного метода и максимальном числе базисных функций 120

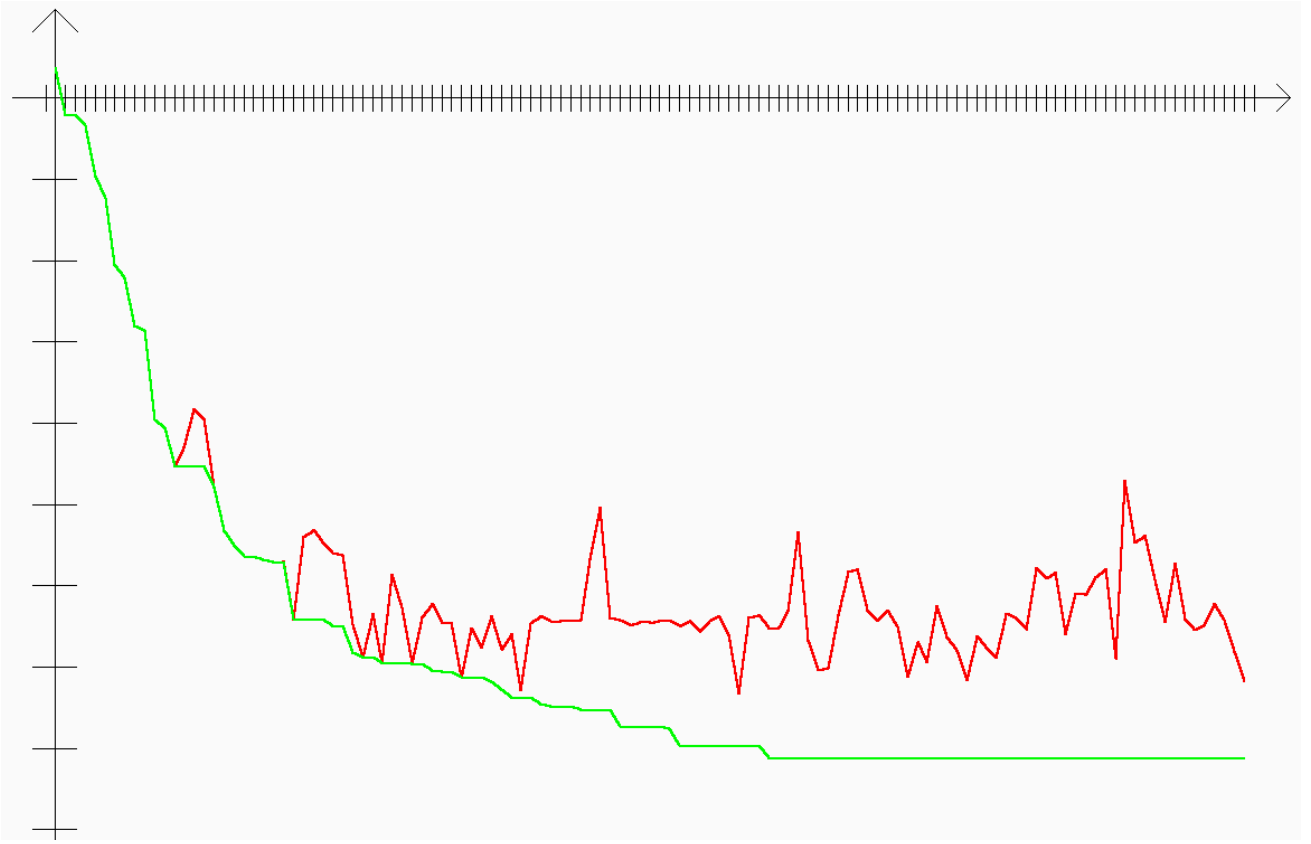

Рисунок 3.19 Сравнительный график зависимости качества аппроксимации функции 2 от количества базисных функций при использовании гибридного метода (красный) и ультра-гибридного метода (зелёный) и максимальном числе базисных функций 120

4.3 Поиск расположения базисных точек, обеспечивающего наилучшую аппроксимацию

Нередко при тестировании ультра-гибридного метода (впрочем, в единичных случаях это наблюдалось и для других методов) для фиксированных базисных потенциалов удавалось достигнуть аппроксимации граничной функции до машинного нуля (в связи с чем, разумеется, невозможно было рисовать логарифмический график), что связано с «базисностью» последовательности исходных точек. Оказывается, существуют такие конечные наборы точек, что система базисных потенциалов от этих точек (при точном вычислении координат проекции функции φ на $L(\alpha_1, \dots, \alpha_m)$) обеспечивает максимально возможную аппроксимацию, а потому можно говорить уже не о приближённом решении, но о точном решении задачи Дирихле в пределах машинных возможностей. Однако, возникает вопрос о местоположении тех самых точек, при которых аппроксимация окажется наилучшей.

Ранее мы решали дифференциальное уравнение в круге с центром в начале координат и радиусом 0.5; при этом использовалась, как правило, одна и та же граничная функция φ и потенциалы брались на точках, расположенных случайно вблизи окружности радиуса 3.5. Тестирование показало, что при использовании ультра-гибридного метода и небольшом числе точек (15-50) аппроксимация довольно часто достигала машинного нуля. Теперь же целью исследования является анализ поведения качества аппроксимации при разном числе точек, расположенных вблизи разных по радиусу окружностей; из этого анализа предполагается выяснить закономерности расположения точек, на которых аппроксимация достигнет машинного нуля. Далее исследуются закономерности расположения таких точек; решение производится с помощью ультра-гибридного метода.

4.3.1 Описание контуров, вблизи которых берутся базисные точки

Для границы каждой тестовой области (на которой определены граничные функции) составляется последовательность подобных по форме контуров,

содержащих внутри себя как кривую, так и все предыдущие контуры (+). Назовём для простоты ***радиусом кривой*** некоторый параметр, от которого зависит «размер» кривой и её положение на плоскости с учётом обязательного выполнения свойства (+); так, для окружности с центром в начале координат параметром будет являться её радиус, для треугольника и квадрата – длина их сторон, для «острия» – радиус окружностей, участвующих в его формировании. Назовём ***минимальным радиусом*** радиус границы области, в которой решается дифференциальное уравнение, а ***максимальным радиусом*** – наибольший радиус в последовательности контуров, вблизи которых берутся базисные точки, то есть радиус последнего контура. Если далее не оговаривается обратного, минимальный радиус считается равным 0.5, максимальный – 3.5. Таким образом, для некоторых названных в (3.1) тестовых областей их границы вместе с последовательностями описанных сейчас контуров выглядят так, как показано на рисунках 4.1-4.2.

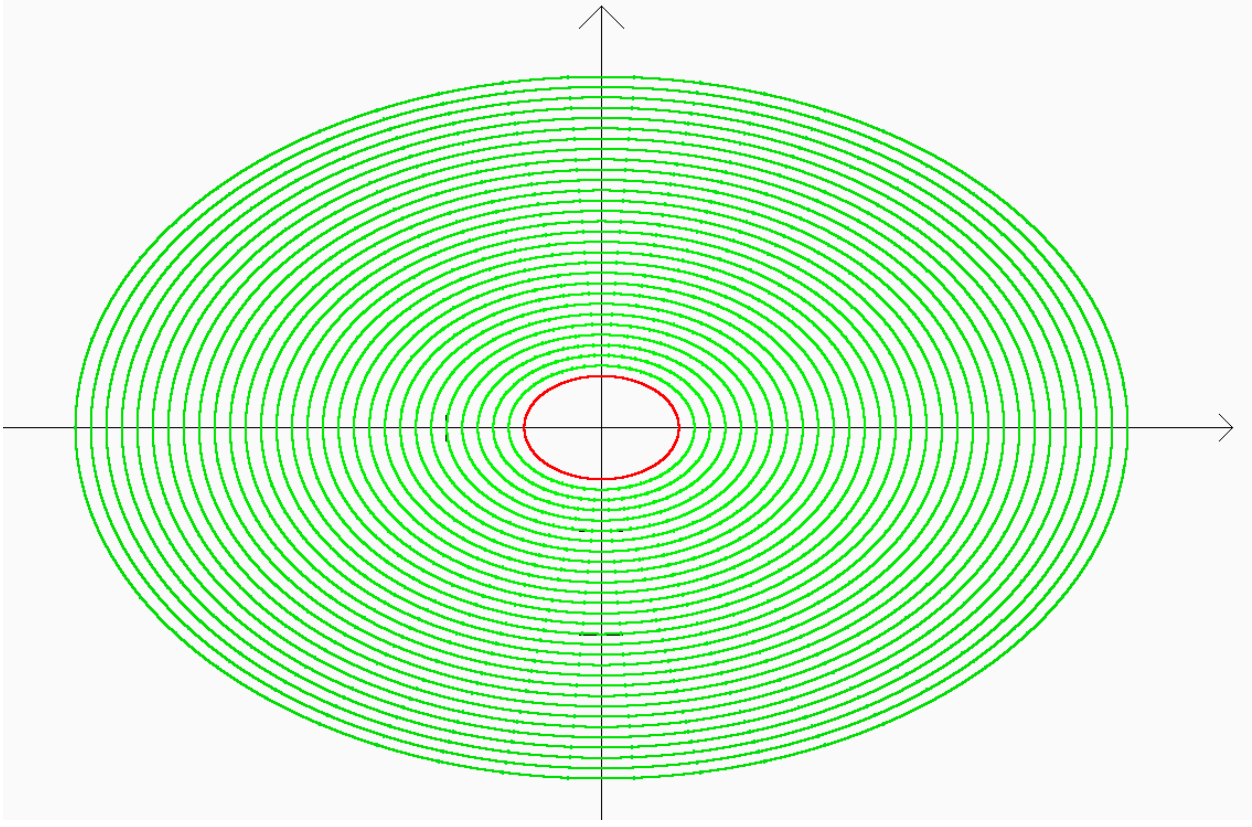

Рисунок 4.1 График границы области 1 и последовательности контуров, вблизи которых берутся базисные точки, при количестве контуров 30

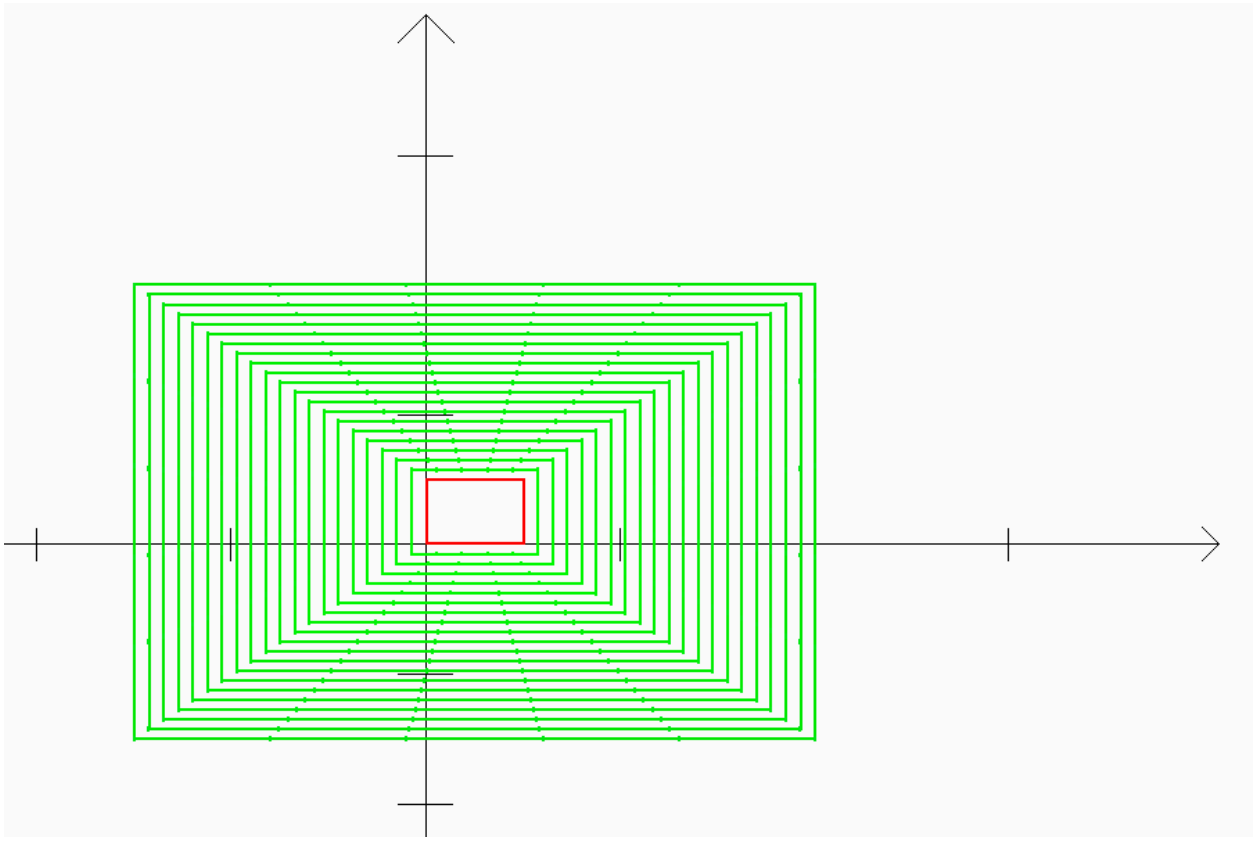

Рисунок 4.2 График границы области 3 и последовательности контуров, вблизи которых берутся базисные точки, при количестве контуров 20

4.3.2 Зависимость качества аппроксимации от кривых, вблизи которых берутся базисные точки

Тестирование программы на зависимость качества аппроксимации от выбранного контура при фиксированном числе базисных потенциалов дало интересные результаты. Оказалось, что аппроксимация существенно зависит от кривой, вблизи которой берутся базисные потенциалы, но об этой зависимости мало что можно сказать: далеко не всегда верно, что контуру с малым радиусом соответствует лучшая аппроксимация, а контуру с большим – худшая, и наоборот; существую контуры, имеющие последовательность точек, на которых аппроксимация достигает машинного нуля, причём эти последовательности точек не являются единственными и вовсе не обязаны включать в себя большое (большее 20-ти, к примеру) количество точек; вместе с этим встречаются последовательности, на которых аппроксимация не принимает машинный ноль даже в тех случаях, когда она обязана это делать (например, при граничной функции, совпадающей с первым базисным потенциалом и,

очевидно, имеющей вектор решения (1,0, … ,0), аппроксимация должна достигать полного нуля). Далее следует описание названных результатов и приводится их графическая иллюстрация.

4.3.2.1 Описание общей зависимости

4.3.2.2 Существование наборов точек, дающих машинный ноль при аппроксимации

При тестировании программы, в среднем, только три раза из пяти программа завершалась удачно и выдавала графики, подобные графикам предыдущего пункта. В остальных случаях программа «вылетала» по неизвестным причинам; к этим причинам можно отнести недочёты в самой программе, которые в связи с особенностями среды программирования очень тяжело отлаживаются, но, как оказалось, почти всегда это происходило потому, что на некоторых последовательностях точек (вблизи каких-то кривых) аппроксимация достигала машинного нуля, а из-за этого функция

$$\vartheta(radius) = \log_{10} \left\| \varphi - \sum_{m=1}^{N} c_m \alpha_m \right\|_{L_2(\partial Q)}^{radius}$$

достигала значения $-\infty$ для некоторых последовательностей точек, лежащих вблизи кривой какого-то радиуса. Разумеется, невозможно на реальном рисунке изобразить ломанную, одна или несколько вершин которой лежат в бесконечности, но в единичных случаях находились последовательности точек, в которых аппроксимация достигала значений около 2.53434e-086 или даже 1.72193e-317, причём подобные результаты имели место для разных кривых и разных граничных функций. Для первой и второй ситуации графики изображены на рисунках 4.8, 4.9.

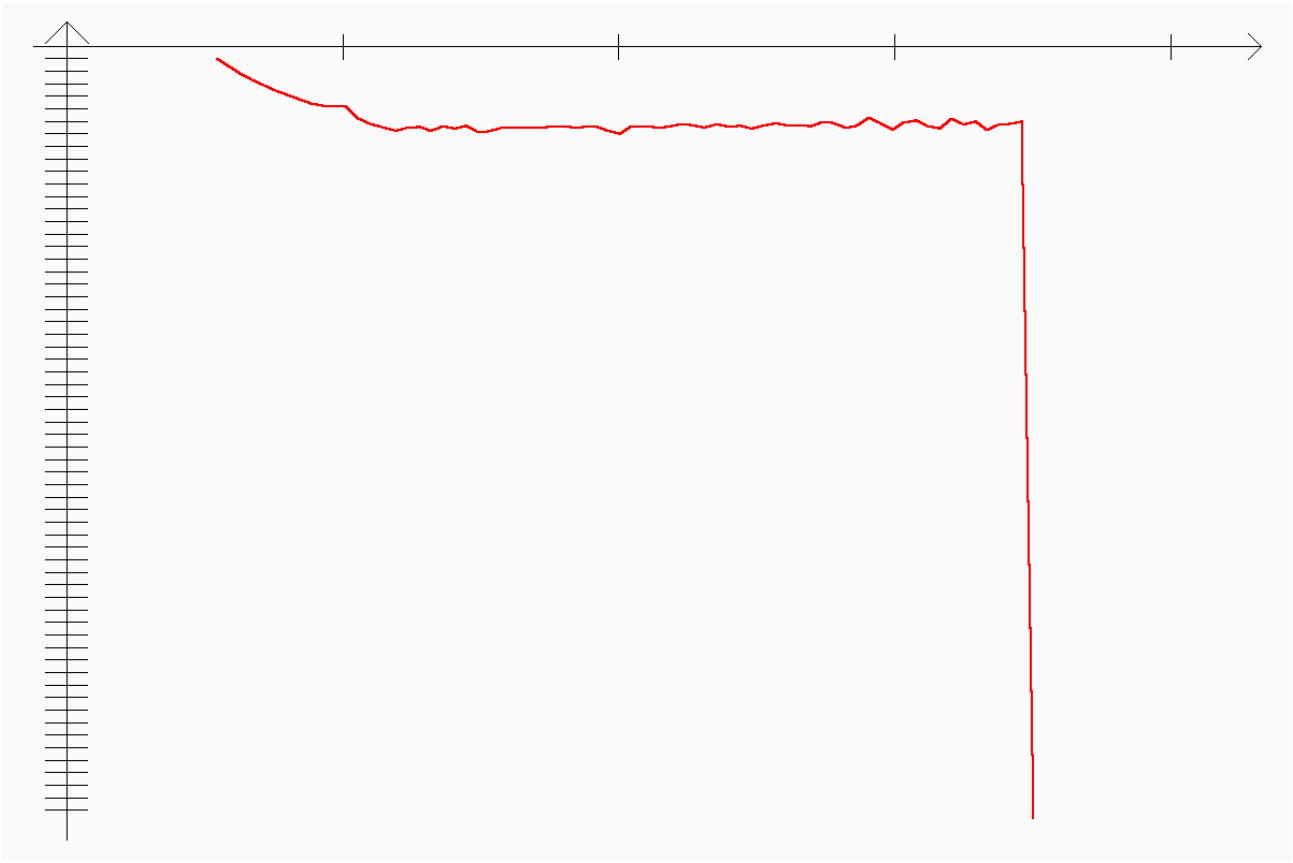

Рисунок 4.8 График качества аппроксимации граничной функции 3 на кривой 1 в зависимости от радиуса, 20 потенциальных функций, 70 кривых между окружностями радиусов 0.5 и 3.5

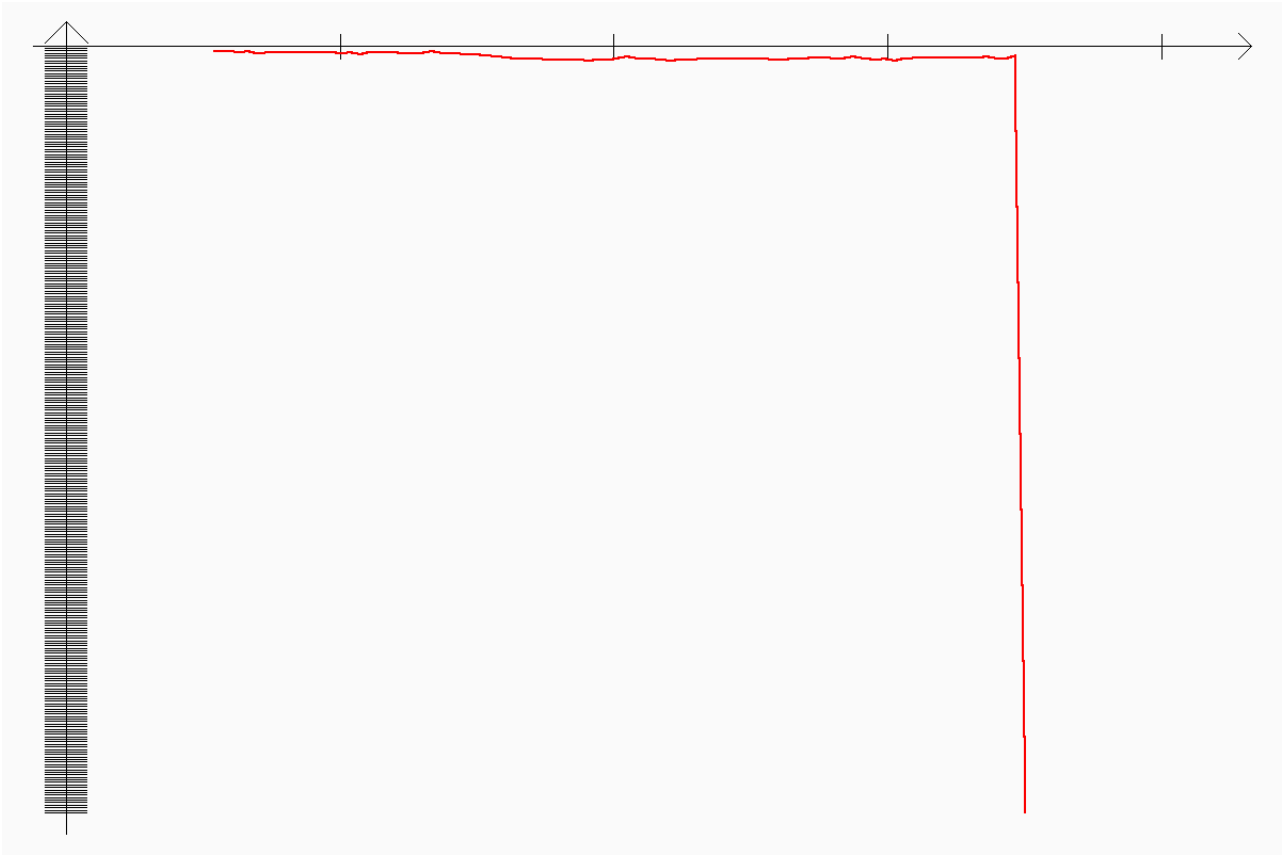

Рисунок 4.9 График качества аппроксимации граничной функции 3 на кривой 2 в зависимости от радиуса, 20 потенциальных функций, 80 кривых между окружностями радиусов 0.5 и 3.5

4.3.2.3 Расположение точек, дающих машинный ноль

Существование точек, дающих нулевую аппроксимацию, привело к вопросу о закономерностях расположения таких точек.

Заключение

Список использованных источников

1 Страуструп, Бьярне. Программирование: принципы и практика с использованием C++, 2-е изд.: Пер. с англ. - М.: ООО "И. Д. Вильяме", 2016. - 1328 с.: ил. - Парал. тит. англ.

2 Ахо, Альфред, В., Хопкрофт, Джон, Ульман, Джеффри, Д. А95 Структуры данных и алгоритмы.: Пер. с англ.: Уч. пос. — М.: Издательский дом "Вильяме", 2000. — 384 с.: ил. — Парал. тит. англ.

3 Мейерс, Скотт. М45 Эффективный и современный C++: 42 рекомендации по использованию C++ 11и C++14.: Пер. с англ. - М.: ООО "ИЛ. Вильяме", 2016. - 304 с.: ил. – Парал. тит. англ.

4 Васильев А. Н. Самоучитель C++ с примерами и задачами. – СПб.; Наука и Техника, 2010. – 480 с.: ил.

5 Алгоритмы / С. Дасгупта, Х. Пападимитриу, У. Вазирани; Пер. с англ. под ред. А. Шеня. — М.: МЦНМО, 2014. — 320 с.

6 Метод базисных потенциалов в задачах математической физики и гидродинамики: монография / А. В. Лежнев, В. Г. Лежнев. – Краснодар: Кубанский гос. ун-т, 2009. – 111 с.

7 Задачи плоской гидродинамики: Учебное пособие. Краснодар: Кубанский государственный университет, 2000. 91 с.

8 Колмогоров А. Н., Фомин С. В. Элементы теории функции и функционального анализа. – М.: Наука. Гл. ред. физ.-мат. лит. 1989.

9 Экстремальные задачи теории приближения. Н. П. Корнейчук. Главная редакция физико-математической литературы изд-ва «Наука». М., 1976 г., 320 с.

10 Лекции по теории аппроксимации. Н. И. Ахиезер. Главная редакция физико-математической литературы изд-ва «Наука». М., 1964 г.

11 Множество единственности потенциала простого слоя. Свидлов А. А., Дроботенко М. И., Бирюк А. Э.

12 Численные методы линейной алгебры: Учеб. пособие/ Г. С. Шевцов, О. Г. Крюкова, Б. И. Мызникова. – М.: Финансы и статистика: ИНФРА-М, 2008. – 480 с.

13 Купрадзе В. Д. О приближенном решении задач математической физики // УМН. 1967. т. XXII. № 2(134). С. 59–107.

14 Купрадзе В. Д., Алексидзе М. А. Метод функциональных уравнений для приближенного решения некоторых граничных задач. // ЖВМиМФ. 1964. № 4. С. 683–715.

15 Лекции по вычислительной математике: Учебное пособие / И. Б. Петров, А. И. Лобанов. – М.: Интернет-Университет Информационных Технологий; БИНОМ. Лабиринт знаний, 2006. – 523 с.

Ссылки

http://phys.bspu.unibel.by/static/um/inf/vmm/pdf/vm2-02.pdf