МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт **математики, физики и информационных технологий**
Кафедра **«Прикладная математика и информатика»**

01.03.02 ПРИКЛАДНАЯ МАТЕМАТИКА И ИНФОРМАТИКА

СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ И КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ

БАКАЛАВРСКАЯ РАБОТА

на тему «Разработка адаптивного алгоритма решения интегральных уравнений методом квадратур по заданным оценочным критериям»

Студент А.Д. Целуев _____

Руководитель О.В. Лелонд _____

Консультант по аннотации Н.В. Ященко _____

Допустить к защите

Заведующий кафедрой к.т.н., доцент А.В. Очеповский _____

«_____» _____ 2017 г.

Тольятти 2017

АННОТАЦИЯ

Объектом исследования ВКР является метод квадратур численного решения интегральных уравнений. При этом под предметом проводимых в работе исследований понимается автоматизация процесса выбора алгоритма по заданным качественным оценочным критериям. Цель работы - проведение анализа целесообразности применения формулы метода прямоугольников решения интегральных уравнений с точки зрения точности получаемых результатов.

В работе используется прямой метод обработки информации (метод исследования) на основе данных, получаемых путём точных (классических) решений интегральных уравнений.

В результате выполнения работы разработан адаптивный алгоритм вычисления собственных значений интегрального уравнения Фредгольма второго рода и осуществлена его программная реализация.

Разработанный алгоритм рекомендуется применять при проведении научно-исследовательских работ, в которых необходимо получение собственных значений интегральных уравнений Фредгольма в качестве основных решений.

В дальнейшем данный алгоритм может быть применен при анализе возможности внедрения новых методов вычислений путем получения недостающей информации о действительных значениях функции методами LU-разложения, либо QR-вращений.

Объем ВКР – 68 страниц, работа содержит 76 формул, 12 рисунков, 12 таблиц, 6 приложений; состоит из четырех глав, число использованных источников – 36.

ABSTRACT

The topic of the given graduation work is “Development of an Adaptive Algorithm for Solving Integral Equations by the Quadrature Method Using Given Estimation Criteria”.

The object of investigation of the stimulated Raman scattering is the Method of the numerical solution quadratures of integral equations. At the same time, under the subject of research carried out in the work, automation of the process of selecting an algorithm based on specified qualitative evaluation criteria. Objective: the aim of this work is to conduct an analysis of the expediency of applying the formula of the method of rectangles to solving integral equations in terms of the accuracy of the results obtained.

In this work, a direct method of information processing is used (the method of investigation) on the basis of data obtained by exact (classical) solutions of integral equations.

As a result of the work, an adaptive algorithm for computing the eigenvalues of the Fredholm integral equation of the second kind is developed and its software implementation is implemented.

The developed algorithm is recommended to be used in carrying out research work which it is necessary to obtain the eigenvalues of Fredholm integral equations as the main solutions in.

In the future, this algorithm can be applied to the analysis of the possibility of introducing new methods of calculation by obtaining the missing information about the actual values of the function by LU decomposition methods, or QR-rotations.

The graduation work consists of an explanatory note on 68 pages, 76 formulas, 12 figures, 12 tables, 6 applications; and the list of 36 references.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Во всех областях современной фундаментальной науки и технических областях деятельности человека зачастую приходится сталкиваться с математическими задачами решения интегральных уравнений, для которых не существует точных решений в явном виде (классических решений), либо эти решения в конечном итоге представляют собой настолько сложные и объемные функциональные зависимости или системы уравнений, что применять их на практике неприемлемо. В таких случаях существует возможность применения численных методов решения интегральных уравнений. Но при этом исследователь того или иного процесса также может столкнуться с рядом проблем, обусловленных, в частности, некорректностью поставленных задач, недостаточной точностью таких вычислений, либо отсутствием возможности дальнейшего применения полученных результатов ввиду их громоздкости. Целью данной работы является проведение анализа целесообразности применения квадратурного метода прямоугольников решения интегральных уравнений с точки зрения точности получаемых результатов.

Существующие алгоритмы, реализующие численные методы, в основной своей массе не дают оценок погрешности получаемых результатов, либо вычисление погрешности производится после получения результата по заданным параметрам, используемым в алгоритмизации процесса вычислений. Другими словами, исследователь получает качественную оценку полученных результатов после полной реализации алгоритма.

Если метод итераций [7] основан на приближении искомых зависимостей по заданным параметрам погрешности вычислений, то аппарат метода квадратур как таковой может и не давать такой оценки. Соответственно, при такой постановке задачи необходимо прямо указать на необходимость задания погрешности вычислений как основного параметра, который напрямую будет управлять последующим вычислительным процессом.

Кроме разработки первичных процедур определения приемлемости будущего решения в работе уделено особое внимание качественной оценке получаемых результатов вычислений по объему информации, т.к. в случае

очень мелкого разбиения сетки, объемы хранимой и обрабатываемой информации увеличиваются квадратично, соответственно для данного метода существует ограничение по машинной производительности.

В существующих методах численного интегрирования оценки погрешности вычислений приводятся, но при этом дается общая оценка приемлемости результата как такового, причем даже при погрешности вычислений 0,5 такой результат зачастую оказывается приемлемым. В работе особое внимание уделено разработке алгоритма определения действительной погрешности вычислений с учетом погрешности задаваемых в качестве исходных данных параметров, которые при использовании методов численного интегрирования обычно получаются путем проведения эксперимента с фиксацией измеренных параметров протекания процесса.

В рамках выполненной квалификационной работы проводится анализ метода квадратур с целью повышения точности вычислений и создания аппарата автоматизированного выбора количества разбиений сетки путем задания качественных критериев оценки получаемых результатов вычислений.

Объектом исследования в данной работе является метод квадратур численного решения интегральных уравнений. При этом под предметом проводимых в работе исследований следует понимать автоматизацию процесса выбора алгоритма по заданным качественным оценочным критериям.

1 ТЕОРЕТИЧЕСКОЕ ОБОСНОВАНИЕ ЗАДАЧИ

1.1 Описание исследуемой задачи

С использованием основных законов сохранения многие исследуемые процессы могут быть описаны посредством математически вычисляемых функций. Поскольку сами по себе законы сохранения энергии, массы, импульса носят интегральный характер, то, в частности, методы численного анализа физических зависимостей находят все большее применение во всех сферах деятельности человека, начиная с энергетики, гидродинамики, акустики и заканчивая экономическими взаимоотношениями, и разработками описательной части бизнес-процессов предприятий.

Отсутствие законных обоснований того или иного процесса связанно чаще всего с некорректной постановкой исследуемых задач или с тем, что в процессе исследований были допущены значительные ошибки, связанные с измерением исходных параметров процесса. В связи с этим математическое решение таких задач оказывается также некорректным или содержащим ошибки (промахи).

Понятие корректности или некорректности постановки задачи решения интегральных уравнений было введено Жаком Адамаром [29]. В данном понимании задача решения интегральных уравнений относительно $y \in Y$ называется корректно поставленной на паре метрических пространств (Y, F), если 1) любому элементу $f \in F$ соответствует решение $y \in Y$ (условие существования решения), 2) из $f_1 = Ay_1 = Ay_2 = f_2$ следует равенство $y_1 = y_2$, т.е. решение определено однозначно (условие единственности решения), 3) для любой погрешности $\varepsilon > 0$ можно указать такое $\delta(\varepsilon) > 0$, что если $\rho_F(f_1, f_2) \le \delta(\varepsilon)$, то $\rho_Y(A^{-1} f_1, A^{-1} f_2) \le \varepsilon$, т.е. обратный оператор A^{-1} непрерывен на F, иначе говоря, любым условно малым ошибкам исходных данных соответствуют малые ошибки решения (условие устойчивости решения).

При невыполнении хотя бы одного из указанных условий задача становится некорректной (некорректно поставленной). Таким образом, само по себе решение интегральный уравнений, не имеющих однозначного решения в

явном виде, методами численного интегрирования является некорректным. В такой ситуации исследователь должен постоянно ориентироваться и применять дополнительные методы анализа при оценке полученных результатов вычислений. Основные типы уравнений в некорректно поставленных задачах и специфика их решений подробно описывается в трудах [7]. Такие уравнения в общем случае относятся к уравнениям первого рода с постоянными пределами интегрирования. Поскольку, как указывалось выше, рассматриваемые задачи зачастую не имеют обоснованного описания на уровне закона, то при постановке задачи исследователь напрямую сталкивается со следующими проблемами: 1) в какой области необходимо произвести аналитическое исследование, т.е. какие пределы интегрирования должны быть установлены при разработке математического аппарата решения поставленной задачи, 2) какая точность вычислений должна быть получена в результате (условие приемлемости результатов вычислений), 3) достаточность требований непрерывности или разрывности основной функции на исследуемом участке (точность начальных полученных данных). Третий вопрос зачастую остается неразрешимым, поскольку конечные решения $y \in Y$ не всегда показывают отступление от линейности исследуемого процесса и точки разрыва функции остаются неизвестными.

В проводимых ранее исследованиях часто упоминается о том, что решение интегральных уравнений первого рода с постоянными пределами интегрирования является некорректным, в частности из-за невыполнения условия устойчивости решения. Даже самые незначительные ошибки при получении исходных параметров могут приводить к настолько большим ошибкам, что численное решение не будет иметь никакого практического значения.

Численные методы решения уравнений первого рода с постоянными пределами интегрирования являются сами по себе некорректными [7]. В частности, к таким уравнениям относятся уравнения Фредгольма I рода, имеющие вид

$$\int_a^b K(x,s)y(s)ds = f(x), c \le x \le d, \quad (1.1)$$

где ядро уравнения $K(x,s)$ и правая часть $f(x)$ - известные функции, а $y(s)$ - искомая функция.

Если решать уравнение (1.1) методом квадратур, заменяя интеграл конечной суммой по формуле трапеций с заданным шагом $h = const$ и решая получаемую СЛАУ относительно значений $y(a)$, $y(a+h)$, …, $y(b)$, то вместо истинного решения на практике получается так называемая знакопеременная «пила» большой амплитуды. Но если подставить полученное решение в уравнение (1.1), то результат оказывается не отличающимся от правой части $f(x)$ в нескольких первых цифрах. Следует также отметить тот факт, что при увеличении дробления, т.е. при уменьшении шага h амплитуда колебаний относительно истинных значений становится еще более значительной. Как показывает практика, аналогичная неустойчивость возникает при решении уравнения (1.1) методами собственных функций, итераций и проекционными методами.

Ввиду таких выводов, сделанных еще в начале прошлого века на основе классического (по Адамару) понимания корректности поставленной задачи, долгое время исследования в направлении развития теории численных методов решения интегральных уравнений Фредгольма I рода не проводились. Позднее Тихоновым А.Н. [29] было сформулировано новое определение корректности.

Рассмотрим задачу решения уравнения (1.1), которое в общем виде может быть записано следующим образом:

$$Ay = f,\ y \in Y,\ f \in F, \quad (1.2)$$

где Y и F - некоторые метрические пространства, а A - непрерывный оператор, отображающий Y на F.

В такой трактовке задача решения уравнения (1.2) называется условно корректной, или корректной по Тихонову, если 1) заведомо известно, что решение y существует и принадлежит некоторому заданному множеству, или множеству корректности $M : y \in M$; 2) решение единственно в классе функций, принадлежащих M; 3) бесконечно малым вариациям f, не выводящим

решение y за пределы M, соответствуют бесконечно малые вариации решения y [7].

Благодаря выводам Тихонова было предложено преобразование, на основании которого вместо решения некорректного уравнения первого рода следует искать решение интегрального уравнения Фредгольма II рода (1.3)

Рассмотрим уравнение

$$\alpha y_\alpha(t) - q y''_\alpha(t) + \int_a^b R(s,t) y_\alpha(s)\,ds = F(t), a \le t \le b, \tag{1.3}$$

где

$$R(s,t) = R(t,s) = \int_c^d \tilde{K}(x,t)\tilde{K}(x,s)\,dx, \tag{1.4}$$

$$F(t) = \int_c^d \tilde{K}(x,t)\,\tilde{f}(x)\,dx, \tag{1.5}$$

$\tilde{K}(x,t)$, $\tilde{f}(x)$ – есть приближенные значения в области $a \le t \le b, c \le x \le d$ функций $K(x,t)$, $f(x)$, q – порядок регуляции (при $q=0$, задача сводится к решению интегрального уравнения Фредгольма II рода).

В данной постановке краевые условия имеют вид:

$$y'_\alpha(a) = y'_\alpha(b) = 0 \tag{1.6}$$

Такой подход дал сильнейший толчок развитию теории численных методов и применению данных методов на практике. Причем, кроме основных вычислительных процессов нахождения конечных функций и их значений, были воплощены в жизнь методы определения критериев приемлемости результатов решения таких задач, что также является немаловажным фактором для оценки применимости конкретных методов.

Для решения уравнений вида (1.3) применяются методы квадратур, вырожденных ядер, наименьших квадратов, Галеркина-Петрова, коллокации, простой итерации, моментов и другие.

Как говорилось ранее, данные методы в той или иной степени подвержены влияниям внешних ошибок, связанных прежде всего с недостаточной точностью получения исходной информации об исследуемом процессе, а также с неточностями, вносимыми при выполнении вычислений (машинные

округления). Дать качественную оценку точности вычислений на основании современных методов решения уравнений возможно только после выполнения таких вычислений, другими словами, поиск правильного решения сводится к постоянному анализу применяемых математических моделей, последующему уточнению, выбору другой модели, уточнению и так далее, до тех пор, пока конечное решении не окажется приемлемым [6], [11], [16]. При этом информация о вычислительной точности самих методов и точности получения исходных данных является заведомо известной. Таким образом, теоретически существует возможность разработки модели численного решения интегральных уравнений посредством предварительного анализа исходных данных и последующего автоматизированного выбора того или иного метода с целью минимизации погрешности конечного решения, а также с точки зрения количества выполняемых операций в процессе вычисления и требуемых для этого машинных мощностей.

Рассмотрение вопроса решения уравнений Фредгольма I рода в данной работе не случайно, поскольку в рассматриваемых методах по Тихонову намерено отбрасывается погрешность численного метода, составляющая определенную величину, которая может повлиять на конечный результат [7], [14]. Поэтому в исследовании данный параметр будет вводится константным значением. Это необходимо для дополнительной оценки приемлемости применения разрабатываемого математического аппарата решения методом квадратур либо итерационными методом уравнений Фредгольма II рода.

1.2 Анализ существующих решений

Пусть задано линейное интегральное неоднородное уравнение Фредгольма II рода:

$$y(x) - \lambda \int_a^b K(x,s)\,y(s)\,ds = f(x), a \le x \le b, \tag{1.7}$$

где независимые переменные изменяются в интервале $[a,b]$, а ядро уравнения $K(x,s)$ определено в квадрат $V = \{x,s : a \le x \le b, a \le s \le b\}$ на плоскости (x,s). При этом не будем рассматривать ситуации, когда ядро уравнения имеет

разрывы, т.е. в постановке задачи на данном этапе укажем, что ядро является непрерывным в пространстве V [29].

Наиболее применимыми и хорошо приспособленными для программного вычисления решений таких уравнений являются численные методы механических квадратур. Суть их в том, что замена интеграла в уравнении на конечную сумму производится на базе какой-либо квадратурной формулы. В результате такого преобразования задача сводится к алгебраической системе относительно дискретных значений искомого решения, соответствующих заданным или определяемых самим выбором формулы значений аргументов в сетке.

Для решения интегрального уравнения необходимо воспользоваться квадратурной формулой определенного вида. Запишем данную замену следующим образом [6]:

$$\int_a^b y(s)\,ds \approx \sum_{j=1}^{n} A_j y(s_j) \tag{1.8}$$

где $s_j \in [a, b]$ – непосредственно узлы сетки; A_j – весовые коэффициенты соответствующие данным узлам; n – количество узлов сетки.

Подставляя правую часть приближенного равенства (1.8) с $y(s) = K(x, s)\,y(s)$ в интегральное уравнение Фредгольма II рода (1.7), получим следующее соотношение:

$$y(x) = \lambda \sum_{j=1}^{n} A_j K(x, s_j)\,y(s_j) + f(x) \tag{1.9}$$

В таком случае приближенное решение интегрального уравнения сводится к решению уравнений вида

$$y(x_i) = \lambda \sum_{j=1}^{n} A_j K(x_i, s_j)\,y(s_j) + f(x_i) \tag{1.10}$$

Для краткости записи системы линейных алгебраических уравнений используют следующие сокращения:

$$K_{ij} = K(x_i, s_j), \quad f_i = f(x_i), \quad y_j = y(s_j) \tag{1.11}$$

Тогда уравнение (1.10) сводится к системе линейных алгебраических уравнений с n неизвестными, которая записывается следующим образом:

$$\begin{cases} (1-\lambda A_1 K_{11})y_1 - \lambda A_2 K_{12} y_2 - \ldots - \lambda A_n K_{1n} y_n = f_1 \\ -\lambda A_1 K_{21} y_1 + (1-\lambda A_2 K_{22}) y_2 - \ldots - \lambda A_n K_{2n} y_n = f_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ -\lambda A_1 K_{n1} y_1 - \lambda A_2 K_{n2} y_2 - \ldots + (1-\lambda A_n K_{nn}) y_n = f_n \end{cases} \quad (1.12)$$

В общем понимании данная СЛАУ, если она совместна, имеет решение, но при этом параметр λ напрямую оказывает влияние на однозначность такого решения [16], [17], [25]. В дальнейшем на основании данного параметра будет рассмотрен один из вариантов решения задачи настоящего исследования.

Итак, находя решение СЛАУ (1.12), получаем n чисел y_1, y_2,…, y_n - так называемый каркас приближенного решения интегрального уравнения на заданной сетке $x = x_i\ (i = 1, 2, \ldots, n)$. В случае, когда в поставленной задаче решения интегрального уравнения необходимым и достаточным является получение именно в таком виде таблицы приближенных значений, то задача считается решенной. Если же необходимо найти аналитическое представление решения $y(x)$ из всего диапазона значений $x \in [a, b]$, то найденные значения $y_j = y(x_j)\ (j = 1, 2, \ldots, n)$ нужно подставить в равенство (1.9), или аппроксимировать функцию по данным значениям. Исходя из практики применения квадратурных формул были сделаны выводы [6], что если ядро и свободный член уравнения не являются достаточно гладкими, то для дискретизации интегрального уравнения не следует применять высокоточных квадратур, а лучше ограничиться такими квадратурными формулами, как формулы трапеций или прямоугольников.

Точность такого метода численного решения интегрального уравнения определяется сразу несколькими взаимосвязанными факторами: 1) применяемой квадратурной формулой, 2) количеством узлов при построении начальной сетки, 3) свойствами определяющих уравнение функций (в частности, свойством периодичности).

Выводы по главе

Решая интегральные уравнения Фредгольма II рода численным методом механических квадратур, необходимо понимать, что замена интеграла конечной суммой всегда дает в результате приближенное решение, т.е. вносит определенную погрешность. При этом с целью достижения заданной точности таких вычислений необходимо разработать алгоритм оценки погрешности путем нахождения решений заданного интегрального уравнения путем последовательного увеличении количества разбиений сетки и дальнейшего сравнения получаемых результатов. Поскольку такое увеличение количества узлов сетки уменьшает погрешность результата вычисления, теоретически существует возможность нахождения линейной зависимости погрешности вычислений от числа узлов сетки. Кроме того, как выше в данной работе уже упоминалось, сложность в подборе адекватной модели решения интегральных уравнений численными методами заключается в возможности проведения оценки полученных результатов существующими методами только после выполнения всего комплекса вычислений. При неудовлетворительном результате, а именно при получении слишком больших погрешностей вычислений, исследователю необходимо проанализировать ошибки и принять новую модель для решения поставленной задачи. Для получения конечного результата с заданной погрешностью необходимо задавать начальное количество итераций в целях получения начальной оценки сходимости метода, это позволит произвести вычисление необходимого количества разбиений сетки.

2 РАЗРАБОТКА ЧИСЛЕННОЙ СХЕМЫ

2.1 Анализ и выбор вычислительного метода

Существующие методы анализа приемлемости получаемых решений интегральных уравнений основаны на проведении по крайней мере одной пары итераций [16]. Рассмотрим данные методы более подробно. Как было определено в предыдущей главе, квадратурные методы не содержат определенных правил, позволяющих точно указать исследователю, какое значение n необходимо задать, или, говоря другими словами, какой шаг h разбиения отрезка $[a, b]$, на котором производится интегрирование, необходимо выбрать, чтобы получить результирующее значение интеграла I с заданной точностью ε.

На примере квадратурной формулы прямоугольников рассмотрим варианты решения задачи определения глобальной погрешности при двух итерациях. Причем в целях определения погрешности необходимо задать в качестве начального ядра функцию, для которой может быть найдено классическое решение. Глобальная погрешность квадратурной формулы прямоугольников вычисляется по формуле

$$r^{\Pi}(h) = \frac{b-a}{24} f''(\xi_{\Pi}) h^2, \quad \xi_{\Pi} \in [a, b], \tag{2.1}$$

где $\xi_{\text{п}}$ - является неопределенной точкой на отрезке $[a, b]$.

Таким образом, как видно из формулы (2.1), при увеличении количества n узлов сетки отрезка $[a, b]$ ошибка метода квадратур численного интегрирования по формуле прямоугольников уменьшается пропорционально квадрату шага разбиения сетки h.

Для анализа необходимо задать начальную функцию ядра, при которой решение интегрального уравнения может быть найдено в явном виде (классическое решение), причем сама функция должна быть непрерывно дифференцируемой на всем диапазоне интегрирования $[a, b]$. Введем следующие обозначения для упрощения представления СЛАУ (1.12)

$$A_j = A = h, \quad f_i = f\left(x_i - \frac{h}{2}\right), \qquad K_{ij} = K\left(x_i - \frac{h}{2}, x_j - \frac{h}{2}\right) \tag{2.2}$$

Для примера найдем решение интегрального уравнения

$$y(x) = \int_1^2 \frac{y(s)ds}{\sqrt{x+s^2}} + \sqrt{x+1} - \sqrt{x+4} + x \tag{2.3}$$

Необходимо заметить, что данное уравнение имеет решение в явном виде $y(x) = x$.

Если подставить соотношения (2.2) в СЛАУ (1.12) и принять количество узлов сетки $n = 5$, то СЛАУ примет следующий вид:

$$\begin{cases} 0{,}86841y_1 - 0{,}11974y_2 - 0{,}10927y_3 - 0{,}10013y_4 - 0{,}09216y_5 = 0{,}29082 \\ -0{,}12624y_1 + 0{,}884337y_2 - 0{,}10615y_3 - 0{,}09771y_4 - 0{,}09026y_5 = 0{,}5144 \\ -0{,}12149y_1 - 0{,}11198y_2 + 0{,}89672y_3 - 0{,}09545y_4 - 0{,}08847y_5 = 0{,}73593 \\ -0{,}11724y_1 - 0{,}10863y_2 - 0{,}10063y_3 + 0{,}906648y_4 - 0{,}08679y_5 = 0{,}9557 \\ -0{,}11341y_1 - 0{,}10556y_2 - 0{,}09818y_3 - 0{,}09138y_4 + 0{,}914797y_5 = 1{,}17395 \end{cases}$$

Решение данной СЛАУ прямым методом даст следующий результат:

$$\begin{cases} y_1 = 1{,}100949 \\ y_2 = 1{,}300934 \\ y_3 = 1{,}500915 \\ y_4 = 1{,}700894 \\ y_5 = 1{,}900872 \end{cases}$$

Действительные значения интегрального уравнения (2.3) в узлах сетки равны

$$\begin{cases} y_1 = 1{,}1 \\ y_2 = 1{,}3 \\ y_3 = 1{,}5 \\ y_4 = 1{,}7 \\ y_5 = 1{,}9 \end{cases}$$

При первичном анализе можно сделать вывод, что решение данного интегрального уравнения Фредгольма II рода уже при $n = 5$ дает хорошую сходимость результатов. Максимальное расхождение полученного решения и действительного результата соответствующий узлу y_1, и составляет $\Delta(y_1) = 0{,}000949$.

Как видно из (2.1) при уменьшении шага разбиения в два раза точность вычислений будет увеличена в четыре раза [25], [25]. Данное утверждение имеет следующий практический смысл: если требуется повысить точность измерений на порядок, то следует уменьшить шаг разбиения в четыре раза. При этом предполагается, что такое увеличение не повлечет за собой значительное увеличение массивов хранимых и обрабатываемых данных. Следует произвести начальную оценку полученных результатов, но выполнять это предложенным способом (2.1) нецелесообразно, поскольку такая постановка задачи повлечет за собой значительное увеличение вычислительных мощностей, причем изначально не будет известно о возможности решения вопроса нахождения второй производной функции численными методами, т.е. задача может оказаться неразрешимой.

Для данных целей приемлемее применить принцип Рунге практического оценивания погрешностей [6], [16], [17]. При этом предполагается начальную оценку погрешности производить на основании алгоритма прямоугольников-трапеций. С этой целью требуется осуществить решение уравнения (2.3) по формуле трапеций. Если применить формулу трапеций, то получаемое количество собственных значений интегрального уравнения всегда равно количеству узлов. Поэтому для получения значений решения в пяти точках необходимо участок интегрирования разбить на четыре отрезка $n = 4$.

Введем следующие обозначения для упрощения представления СЛАУ (1.12) по формуле трапеций

$$A_j = h,\ \lambda_j = \frac{1}{2},\ (j = 1, n),\ \lambda_j = 1,\ (j = 2, 3, \ldots, n-1),\ f_i = f(x_i),\ K_{ij} = K(x_i, x_j) \quad (2.4)$$

В такой постановке задачи решение заданного уравнения по квадратурной формуле трапеций дало результат А, определенно имеющий большую погрешность $\Delta(y_1) = -0{,}002987$.

$$A\begin{cases} y_1 = 0{,}997013 \\ y_2 = 1{,}247064 \\ y_3 = 1{,}497137 \\ y_4 = 1{,}74722 \\ y_5 = 1{,}997306 \end{cases}$$

По существу, оценивать такие результаты не представляется возможным согласно принципу Рунге в связи с тем, что количество точек сеток указано для формул прямоугольников и трапеций разное. Это делается намеренно с целью получения решений по формулам прямоугольников и трапеций с совпадающим одним узлом. Необходимо решить такую же задачу, применяя формулу прямоугольников с узлами сетки при $n = 3$. Полученное решение также будет совпадать в одной точке $x = 1{,}5$.

$$\begin{cases} y_1 = 1{,}169279 \\ y_2 = 1{,}502533 \\ y_3 = 1{,}83577 \end{cases}$$

Расположим данные результаты по указанной точке сетки в порядке возрастания количеств разбиений сетки и вычислим погрешность Δ решений, получаемую относительно классического решения заданного уравнения:

$$\begin{cases} y_1 = 1{,}502533; \quad (\textit{формула прямоугольников}) \text{ при } n = 3; \Delta = 0{,}002533 \\ y_2 = 1{,}497137; \quad (\textit{формула трапеций}) \text{ при } n = 4; \Delta = 0{,}002863 \\ y_3 = 1{,}500915; \quad (\textit{формула прямоугольников}) \text{ при } n = 5; \Delta = 0{,}000915 \end{cases}$$

Поскольку из полученных результатов становится ясным, что в такой логике применение формул прямоугольников и трапеций, без дальнейшего восстановления начальной функции, не дает ясного решения поставленных задач по принципу Рунге при достаточно малом количестве разбиений сетки, то такая модель не может быть применена в данной работе.

Для дальнейшего анализа необходимо остановиться на применении одних только формул прямоугольников, это даст возможность адекватного суждения о сходимости результатов в точках, получаемых в пересечениях сеток при разных заданных n [29].

Проведем анализ на примере функции, заданной уравнением (2.3) при задании четных значениях $n = 2, 4, 6, 8, 10$.

Таблица 2.1 – Вычисленные собственные значения функции в узлах сеток

Истинные значения	Вычисленные параметры сетки									
	n=2		n=4		n=6		n=8		n=10	
	y(x)	Δ(y(x))	y(x)	Δ(y(x))	y(x)	Δ(y(x))	y(x)	Δ(y(x))	y(x)	Δ(y(x))
1,05									1,050238	0,000238
1,0625							1,062872	0,000372		
1,083333					1,083993	0,00066				
1,125			1,126478	0,001478						
1,15									1,150237	0,000237
1,1875							1,187869	0,000369		
1,25	1,255794	0,005794			1,250652	0,000652			1,250235	0,000235
1,3125							1,312865	0,000365		
1,35									1,350233	0,000233
1,375			1,376447	0,001447						
1,416667					1,417308	0,000641				
1,4375							1,43786	0,00036		
1,45									1,45023	0,00023
1,55									1,550228	0,000228
1,5625							1,562855	0,000355		
1,583333					1,583963	0,00063				
1,625			1,626408	0,001408						
1,65									1,650225	0,000225
1,6875							1,68785	0,00035		
1,75	1,755507	0,005507			1,750617	0,000617			1,750222	0,000222
1,8125							1,812845	0,000345		
1,85									1,85022	0,00022
1,875			1,876366	0,001366						
1,916667					1,917271	0,000604				
1,9375							1,937839	0,000339		
1,95									1,950217	0,000217

При применении формулы (2.1) в случае если действительное решение известно, становится возможным вычислить вторую производную в решении для глобальной погрешности формулы прямоугольников в применении к полученным узловым значениям искомой функции [17]. Для вычисления значения глобальной погрешности в узловых значениях необходимо производить выборку максимального по модулю отклонения полученных значений от истинного в данной точке по следующей формуле:

$$r^{\Pi}(h^n) = \max$$ (2.5)

Тогда вторая производная искомой функции в некоторой точке из отрезка $[a, b]$ вычисляется по формуле

$$f''(\xi_{\text{П}}) = \frac{r^{\text{П}}(h)}{h^2} \cdot \frac{24}{b-a} \qquad (2.6)$$

На основании формул (2.5), (2.6) получаем следующие численные значения второй производной функции для заданных $n = 2, 4, 6, 8, 10$.

Таблица 2.2 – Численные значения второй производной функции

	$n=2$	$n=4$	$n=6$	$n=8$	$n=10$
$f''(\xi_{\text{П}})$	0,556222	0,567478	0,57019	0,571325	0,571931

Нетрудно заметить, что даже при увеличении исходного количества узловых точек сетки в 5 раз числовое значение второй производной функции изменяется на небольшую величину, а при последних значениях в рассмотренном примере – остается практически неизменным [6], [11].

Анализируя таблицу собственных значений функций в точках сетки (Таблица 2.1), можно убедиться, что для узловых точек $x_i = 1{,}25; 1{,}75$, в которых сетки функций пересекаются, значения второй производной функции для значений $n = 2, 6, 10$ изменяются с еще меньшей скоростью.

Таким образом, на основании формулы (2.6) уже после получения первого приближенного решения интегрального уравнения Фредгольма II рода на основе проведенного предварительного анализа можно получить приближенное значение n^+ разбиений отрезка $[a, b]$, которое будет удовлетворять заданным критериям точности вычислений, по формуле

$$n^+ \approx \sqrt{\frac{f''(\xi_{\text{П}}) \cdot (b-a)^3}{r^{\text{П}}(h) \cdot 24}}, \qquad (2.7)$$

где $r^{\text{П}}(h)$ – заданное значение глобальной погрешности вычислений по абсолютной величине; $f''(\xi_{\text{П}})$ – значение второй производной, вычисленное по формуле (2.6) при заданном начальном значении n.

Поскольку из таблицы числовых значений второй производной функции $f''(\xi_{\text{П}})$ (Таблица 2.2) видно, что выполняются неравенства

$f''(\xi_{\text{П}})^{n=2} < f''(\xi_{\text{П}})^{n=4} < f''(\xi_{\text{П}})^{n=6} < f''(\xi_{\text{П}})^{n=8} < f''(\xi_{\text{П}})^{n=10}$, то становится возможным сделать вывод о том, что при меньшем n значение n^+ будет более полно удовлетворять критериям точности. Другими словами, чем меньше величина изначально заданного n, тем большей получается величина n^+ и, соответственно, погрешность вычисления при разбиении сетки на n^+ узлов, становится меньше.

2.2 Построение и тестирование алгоритма

Проведенный анализ относительно квадратурного метода решения интегрального уравнения Фредгольма II рода (2.3) по формулам прямоугольников при задании четных значениях $n = 2, 4, 6, 8, 10$ позволяет сформулировать задачу итерационного процесса приближений к истинному решению. Таким образом, в случае, если существует возможность прямой оценки глобальной погрешности в точках сетки без привлечения к данным решениям аналитических методов (классических решений), то данный процесс можно назвать полностью вычислительным (численным методом решения) [24].

Процесс сжимания сетки по определенному алгоритму по сути является итерационным. Если произвести разложение погрешности измерений по следующему принципу:

$$r^{\text{П}}(h)^1 = (y_i^{\,1} - y_i^{\,2}) + (y_i^{\,2} - y_i^{\,3}) + \ldots + (y_i^{\,n-1} - y_i^{\,n}) = y_i^{\,1} - y_i^{\,n} \text{ при } n \to \infty \quad (2.8)$$

где i – любой из узлов сетки, то становится ясным, что сходимость итерационного процесса будет обеспечена при сколь угодно неограниченно возрастающем n. Основываясь на выражении (2.6) и на выводах, сделанных при расчете n^+ (2.7), строится следующая зависимость (для $n = 2$):

$$r^{\text{П}}(h)^2 = (y_i^{\,2} - y_i^{\,6}) + (y_i^{\,6} - y_i^{\,10}) + \ldots + (y_i^{\,n-4} - y_i^{\,n}) = y_i^{\,2} - y_i^{\,n} \text{ при } n \to \infty \quad (2.9)$$

Условно примем, что оценки выше 10-й итерации не влияют на определение полной погрешности, тогда выражение (2.9) можно переписать в следующем виде:

$$r^{П}(h)^2 \approx (y_i^{\,2} - y_i^{\,6}) + 2\cdot(y_i^{\,6} - y_i^{\,10}) = y_i^{\,2} + y_i^{\,6} - 2\cdot y_i^{\,10} \qquad (2.10)$$

Тогда при установке конкретных значений требуемой погрешности определения собственных значений задачу можно поставить как задачу для решения уравнения методом итераций при последовательном приближении результата вычислений к заданным параметрам.

Но в таком случае при получении сколь угодно достоверных значений n^+ сам процесс итераций можно остановить уже на каком-то определенном n. В соответствии с (2.10) процесс итераций можно остановить уже на третьем шаге, выполнив вычисления при $n = 2, 6, 10$. При этом в узлах пересечений данных сеток (для $x_i \in [1, 2]$ такими узлами являются $x_i = 1{,}25; x_i = 1{,}75$ при использовании формулы прямоугольников), следует запоминать получающиеся значения приближений значений искомой функции.

Рассчитывая таким методом n^+ для решения уравнения (2.3) при заданном значении $r^{П}(h) = 0{,}0001$, получим следующие результаты:

при $x_i = 1{,}25$

$$r^{П}(h)^2 \approx y_i^{\,2} + y_i^{\,6} - 2\cdot y_i^{\,10} = 1{,}255794 + 1{,}250652 - 2\cdot 1{,}250235 = 0{,}005976$$

при $x_i = 1{,}75$

$$r^{П}(h)^2 \approx y_i^{\,2} + y_i^{\,6} - 2\cdot y_i^{\,10} = 1{,}755507 + 1{,}750617 - 2\cdot 1{,}750222 = 0{,}005679$$

Данные оценки получаются несколько грубее действительных параметров:

при $x_i = 1{,}25$

$r^{П}(h)^2 = 0{,}005793981$

при $x_i = 1{,}75$

$r^{П}(h)^2 = 0{,}005506937$

При этом условно можно считать, что получаемые результаты вычислений будут отвечать заданным критериям погрешности, поскольку формула (2.10) вводит еще более жесткие условия вычислений.

Далее по формуле (2.6) вычисляется вторая производная функции (по (2.5) необходимо выбрать максимальное значение из модулей $r^{П}(h)^2$):

$$f''(\xi_{П}) = \frac{r^{П}(h)}{h^2}\cdot\frac{24}{b-a} = \frac{0{,}005976}{0{,}5^2}\cdot\frac{24}{2-1} = 0{,}573698$$

В результате получаем

$$n^{+} \approx \sqrt{\frac{f''(\xi_{\text{П}})}{r^{\text{П}}(h)} \cdot \frac{(b-a)}{24}} = \sqrt{\frac{0{,}573698}{0{,}0001} \cdot \frac{(2-1)}{24}} = 15{,}46095$$

Данное значение для применения в алгоритме следует округлить до ближайшего верхнего целого значения, кратного двум. Таким образом в поставленной задаче необходимо решить СЛАУ размерностью 16×16. Полученное решение представлено в следующей таблице:

Таблица 2.3 – Результаты вычисления значений функции при $n = 16$

n	Значение y_i		$r^{\text{П}}(h)^2$	n	Значение y_i		$r^{\text{П}}(h)^2$
	вычисл.	истинное			вычисл.	истинное	
1	1,031343	1,03125	0,000093	9	1,531339	1,53125	0,000089
2	1,093843	1,09375	0,000093	10	1,593839	1,59375	0,000089
3	1,156342	1,15625	0,000092	11	1,656338	1,65625	0,000088
4	1,218842	1,21875	0,000092	12	1,718837	1,71875	0,000087
5	1,281342	1,28125	0,000092	13	1,781337	1,78125	0,000087
6	1,343841	1,34375	0,000091	14	1,843836	1,84375	0,000086
7	1,40634	1,40625	0,000090	15	1,906335	1,90625	0,000085
8	1,46884	1,46875	0,000090	16	1,968835	1,96875	0,000085

На основании результатов вычислений (Таблица 2.3) в дальнейшем принимается, что функция (2.6) удовлетворяет заданным оценочным критериям при решении уравнения Фредгольма II рода. При этом следует учитывать, что при решении других интегральных уравнений данный алгоритм может давать результаты, не удовлетворяющие заданным критериям оценки [6], [16], [24].

Разработка и тестирование пробных версий программ реализации адаптивного алгоритма выполнены в среде разработки Visual Studio 2017 на языке C++ [21], [32]. Следует провести некоторую адаптацию алгоритма под конкретное применение. Так же необходимо учесть цикличность выполнения определенных действий отдельными функциями. Из положений алгоритма следует, что основным циклическим (итерационным) процессом в данном алгоритме будет является непосредственное решение СЛАУ. Следует заметить, что применение приближенных численных методов решения СЛАУ всегда будет вносить дополнительную погрешность вычислений. Поэтому применение таких алгоритмов в решении поставленных задач недопустимо [7], [1].

При задании высоких требований к точности конечных вычислений потребуются значительные объемы памяти, т.к. даже при решении уравнения (2.3) по формуле прямоугольников при задании точности вычислений $\varepsilon = 1 \cdot 10^{-6}$ необходимо решать СЛАУ размерностью 4096×4096, что повлечет необходимость выделения оперативной памяти для хранения и обработки матрицы объемом 128 Мб. Таким образом, реализацию алгоритма для последующего проведения экспериментов необходимо выполнить в консольном приложении.

При первичных итерациях в соответствии с разработанным алгоритмом последовательно выполняются операции по нахождению собственных значений интегрального уравнения. Данную функцию необходимо определить основной и выполнить её отдельным вложением. При этом она должна содержать интерфейсную часть, позволяющую осуществлять вывод результатов в файл и на монитор для последующей обработки и анализа получаемого численного решения.

Основной модуль программы должен выдавать промежуточные результаты, получаемые при первичных итерациях, на экран, конечный результат работы вне зависимости от его объема также подлежит выводу на экран. На этапе проведения экспериментов не следует уделять внимание ошибкам оператора, должно быть принято, что реализация дополнительных функций по отслеживанию ввода ошибочных значений задаваемых параметров является избыточной [21], [32].

В целях применения конечной реализации программы в исследованиях, код необходимо реализовать без применения оптимизации алгоритма под конкретный язык программирования. Это позволит использовать каркас решения при распространении в приложениях пользователей. В целом для ускорения вычислительного процесса при программировании в среде разработки Visual Studio 2017 на языке C++ существует множество вариантов оптимизации кода, начиная от создания многопоточных целевых функций и заканчивая разработкой нейронно-сетевого приложения [21], [32] .

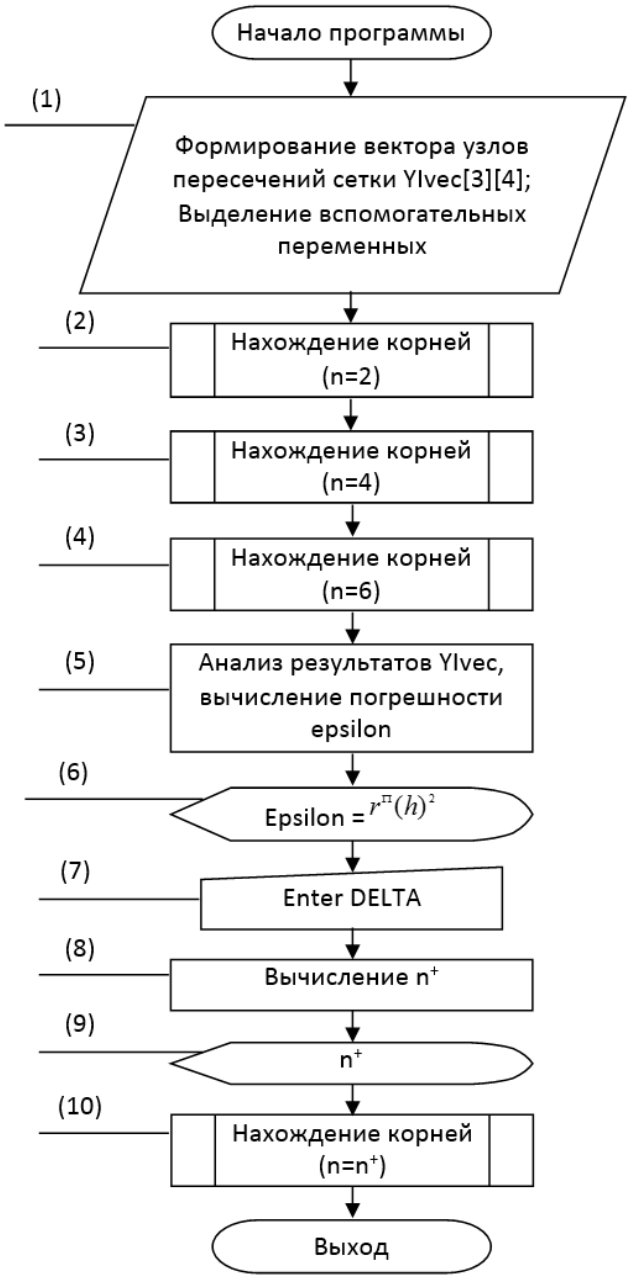

Рисунок 1 – Алгоритм основного модуля программы

Процедура решения интегральных уравнений должна быть универсальной и выдавать конечный результат вне зависимости от заданного количества узлов сетки. При выводе на монитор значения n^+ необходимо сделать прерывание, чтобы у оператора была возможность предварительно оценить количество выделяемой в дальнейшем оперативной памяти и время, которое потребуется на выполнение вычислений.

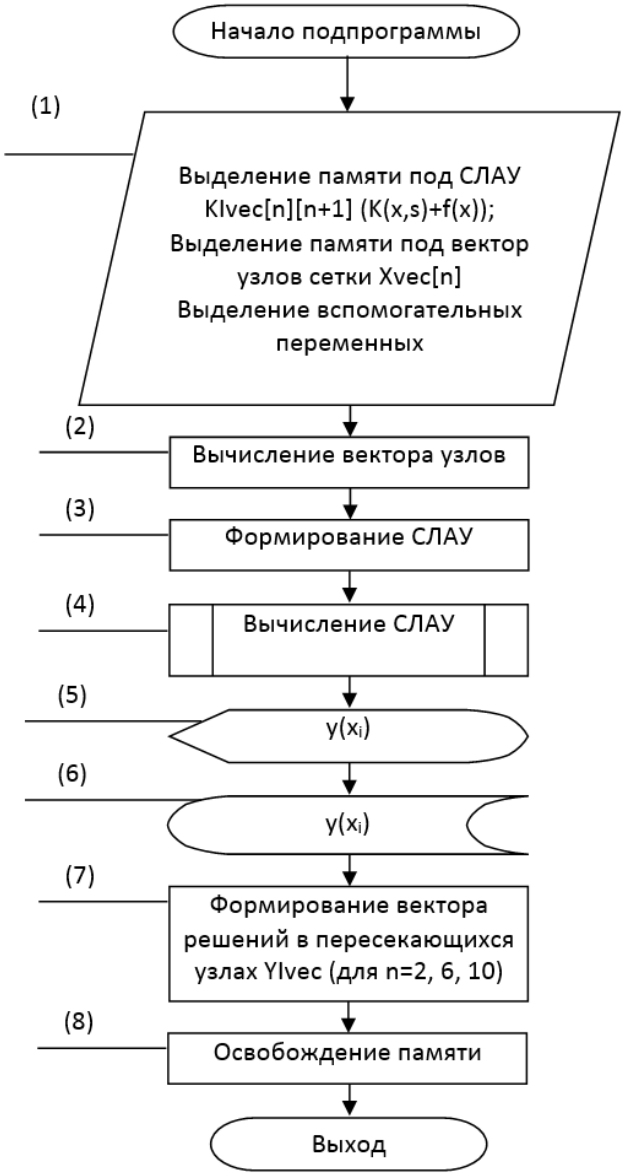

Рисунок 2 – Алгоритм подпрограммы нахождения решений интегрального уравнения

Для реализации применен алгоритм подпрограммы прямого решения СЛАУ методом Гаусса.

Выводы по главе

Разработанный адаптивный алгоритм решения интегральных уравнений Фредгольма II рода позволяет организовать вычислительные эксперименты для решения различных заданных уравнений.

Задача решения интегрального уравнения Фредгольма II рода представляется в исходном коде, без применения алгоритмов интерпретаций,

которые могут потянуть за собой глобальные ошибки в представлении данных.

Поскольку конечные результаты сохраняются в текстовом файле, то в случае решения интегральных уравнений, для которых известно классическое решение, либо известно решение, полученное с определенной точностью другими численными методами, результаты таких вычислений можно легко подвергать анализу.

Конечные аналитические выкладки целесообразно выполнять в MS Excel или среде MatCAD. Это позволит избежать написания «широкого» кода подпрограмм, реализующих стандартные математические преобразования, а также упростит графическое представление результатов.

3 ПРОВЕДЕНИЕ ВЫЧИСЛИТЕЛЬНОГО ЭКСПЕРИМЕНТА И КОРРЕКТИРОВКА РАЗРАБОТАННОГО АЛГОРИТМА

3.1 Планирование и осуществление вычислительного эксперимента

В целях получения адекватных оценок относительно корректности полученного алгоритма необходимо при проведении вычислительных экспериментов произвести обработку данных, получаемых при вычислении интегральных уравнений Фредгольма II рода, имеющих разные степени гладкости как ядра уравнения так и их свободных членов [17], [19], [20].

Необходимо также определить степени влияния внесения изменений в формулы (2.5), (2.6), (2.7), (2.10). При этом следует четко понимать, что сам по себе факт проведения большего или меньшего количества вычислений, т.е. временных затрат, не является основополагающем в настоящей работе и может быть отражен только в качестве дополнительной информации [10].

Применим разработанный алгоритм для решения некоторых интегральных уравнений [10].

Эксперимент 1.

$$y(x)=\frac{3}{10\pi}\int_{-\pi}^{\pi}\frac{y(s)ds}{0{,}64\cdot\cos^2\left(\frac{x+s}{2}\right)-1}+25-16\cdot\sin^2(x) \tag{3.1}$$

Точное решение уравнения (3.1):

$$y(x)=\frac{17}{2}+\frac{128}{17}\cos(x) \tag{3.2}$$

Эксперимент 2.

$$y(x)=-\int_0^1 xe^{xs}y(s)ds+e^x \tag{3.3}$$

Точное решение уравнения (3.3):

$$y(x)\equiv 1 \tag{3.4}$$

Эксперимент 3.

$$y(x)=\int_0^{\pi}(\sin(s)+\cos(x))y(s)ds+e^x-\frac{1+e^{\pi}}{2}+(-e^{\pi})\cos(x) \tag{3.5}$$

Точное решение уравнения (3.5):

$$y(x) = e^x \tag{3.6}$$

В целом по выполненным экспериментам можно сделать однозначный вывод о низкой адекватности разработанного алгоритма. Так, при решении задачи в эксперименте 1 в качестве задаваемых критериев точности вычислений был принят следующий ряд:

$$\begin{cases} \Delta_1 = 1 \cdot 10^{-3} \\ \Delta_2 = 1 \cdot 10^{-5} \\ \Delta_3 = 1 \cdot 10^{-7} \end{cases}$$

При этом в начальных делениях сетки при $n = 2, 6, 10$ прослеживалось очень быстрое приближение к точному решению, поскольку прямые оценки разностей в точках пересечений сеток уже при $n = 10$ стремились к $\Delta = 1 \cdot 10^{-7}$. Тем не менее эксперимент был проведен в полном объеме [24], [14].

Получены следующие действительные параметры выполненных построений:

Заданные значения: $\begin{cases} \Delta_1 = 1 \cdot 10^{-3} \\ \Delta_2 = 1 \cdot 10^{-5} \\ \Delta_3 = 1 \cdot 10^{-7} \end{cases}$ Результат: $\begin{cases} n+ = 116 \\ n+ = 1146 \\ n+ = 11444 \end{cases} \begin{cases} \Delta_1 = -1 \cdot 10^{-13} \\ \Delta_2 = -3 \cdot 10^{-13} \\ \Delta_3 = -1 \cdot 10^{-12} \end{cases}$

При задании $\Delta_3 = 1 \cdot 10^{-7}$ и последующем решении по заданному алгоритму, объем выделяемой памяти под обрабатываемую СЛАУ составил $0{,}976\,Gb$. Как видно из данных результатов, при значительной плотности сетки погрешность получаемых результатов начинает расти. Это говорит о том, что одним из влияющих факторов при таких решениях является вычислительная погрешность, связанная с ограничением ряда возможных значений количеством выделяемых байт. В разработанной для экспериментов программе использовался тип *double*. При этом замечено, что несмотря на периодический характер задаваемой функции (действительное решение уравнения) данные погрешности вычислений не имеют свойства взаимоуничтожаться, т.е. их распределение не является нормальным [7], [1].

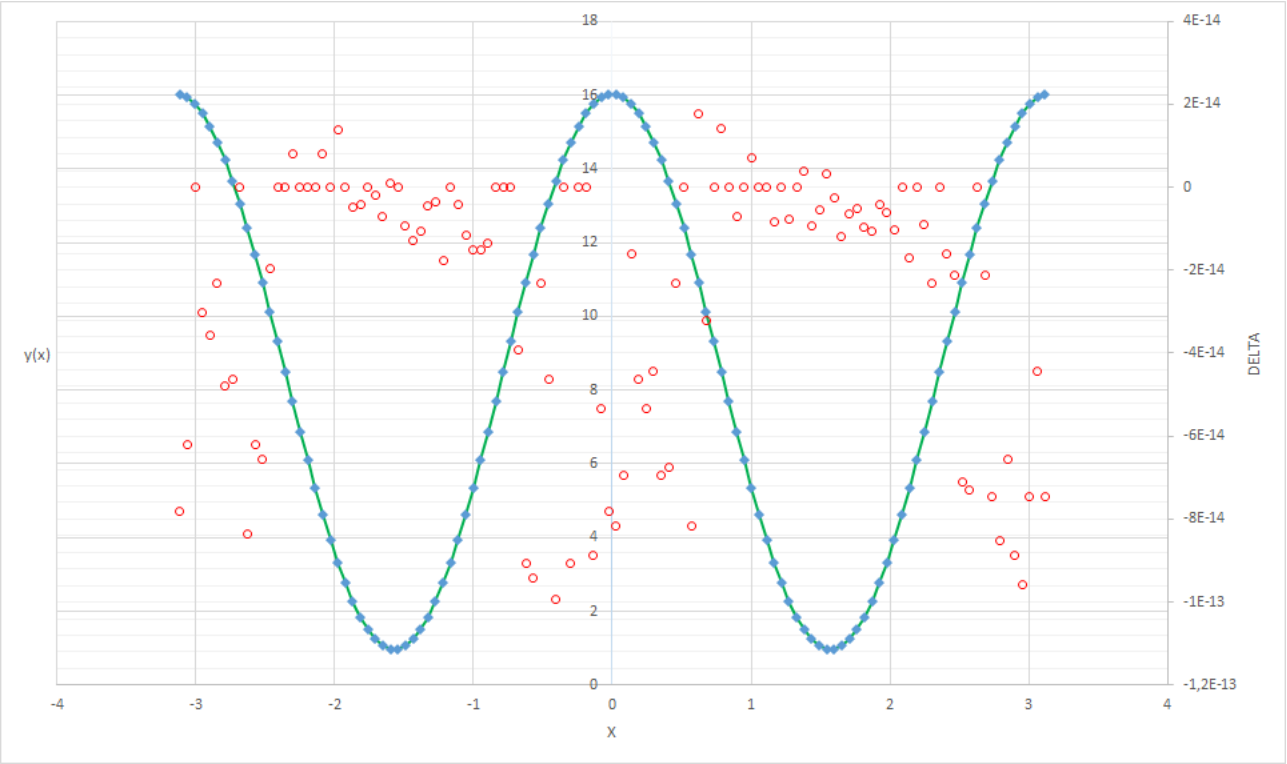

Рисунок 3 – Результаты эксперимента 1 при заданной точности $\Delta_1 = 1 \cdot 10^{-3}$

Уже при первом сжимании сетки видно, что абсолютные погрешности стремятся к нулю при приближении значения функции к минимуму. При этом, полученная точность измерений значительно превышает заданные оценочные критерии, что также говорит о низком уровне адекватности построенной модели.

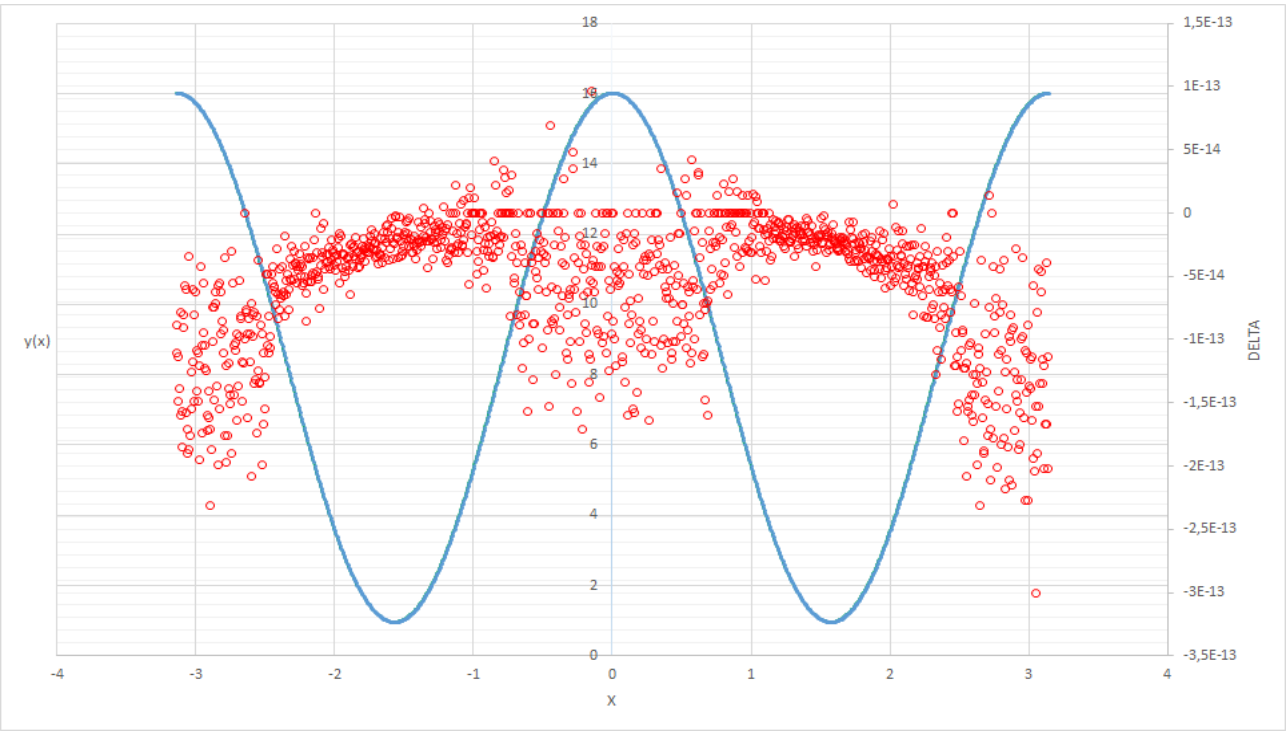

Рисунок 4 – Результаты эксперимента 1 при заданной точности $\Delta_2 = 1 \cdot 10^{-5}$

При дальнейшем сжимании сетки и последующем построении результатов таких вычислений получились еще более наглядные результаты в значительной мере подтверждающие выводы сделанные при первом разбиении. В около нулевой области графика функции дисперсия погрешности минимальна о чем свидетельствует плотность точек, соответствующих значениям функции на отрезке $x = [-1{,}6; 1{,}6]$.

В целях подтверждения выводов, сделанных выше, при построении графика результатов (при $\Delta_1 = 1 \cdot 10^{-7}$, Рисунок 5) использована относительная оценка погрешности в процентном выражении.

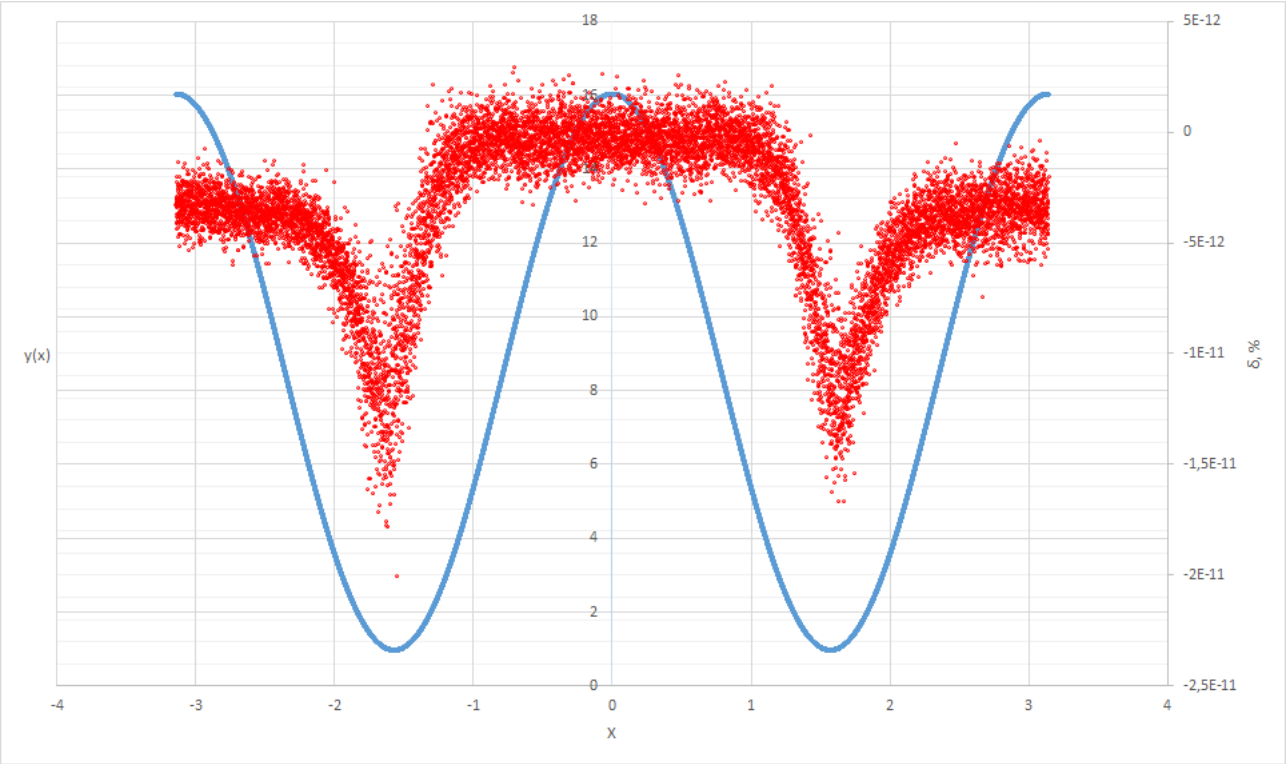

Рисунок 5 – Результаты эксперимента 1 при заданной точности $\Delta_3 = 1 \cdot 10^{-7}$

Результаты эксперимента 2 носят более качественный характер. Получены следующие характеристики:

Заданные значения: $\begin{cases} \Delta_1 = 1 \cdot 10^{-5} \\ \Delta_2 = 1 \cdot 10^{-7} \\ \Delta_3 = 1 \cdot 10^{-9} \end{cases}$ Результат: $\begin{cases} n+ = 42 \\ n+ = 408 \\ n+ = 4074 \end{cases} \begin{cases} \Delta_1 = 3 \cdot 10^{-5} \\ \Delta_2 = 3 \cdot 10^{-7} \\ \Delta_3 = 3 \cdot 10^{-9} \end{cases}$

Абсолютная погрешность вычислений в данном эксперименте в три раза превышает установленные критерии. Таким образом оценка адекватности построенной модели не может быть полностью положительной. Но при этом,

относительно положительным моментом в результатах решений является тот факт, что можно сделать вывод о возможности существования прямой зависимости между количеством делений сетки, искомым решением и погрешностью данного решения. На примере функции тождественно равной единице это очень хорошо видно при построении решений на графиках.

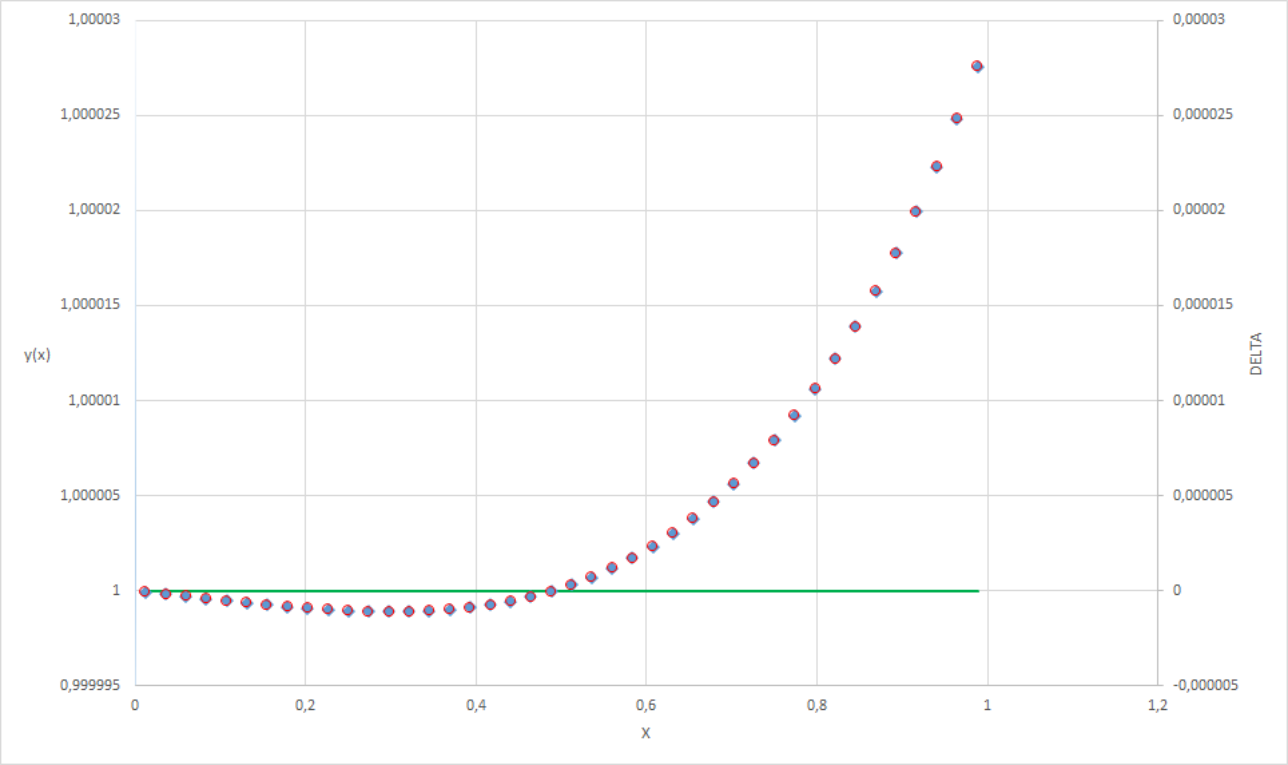

Рисунок 6 – Результаты эксперимента 2 при заданной точности $\Delta_1 = 1 \cdot 10^{-5}$

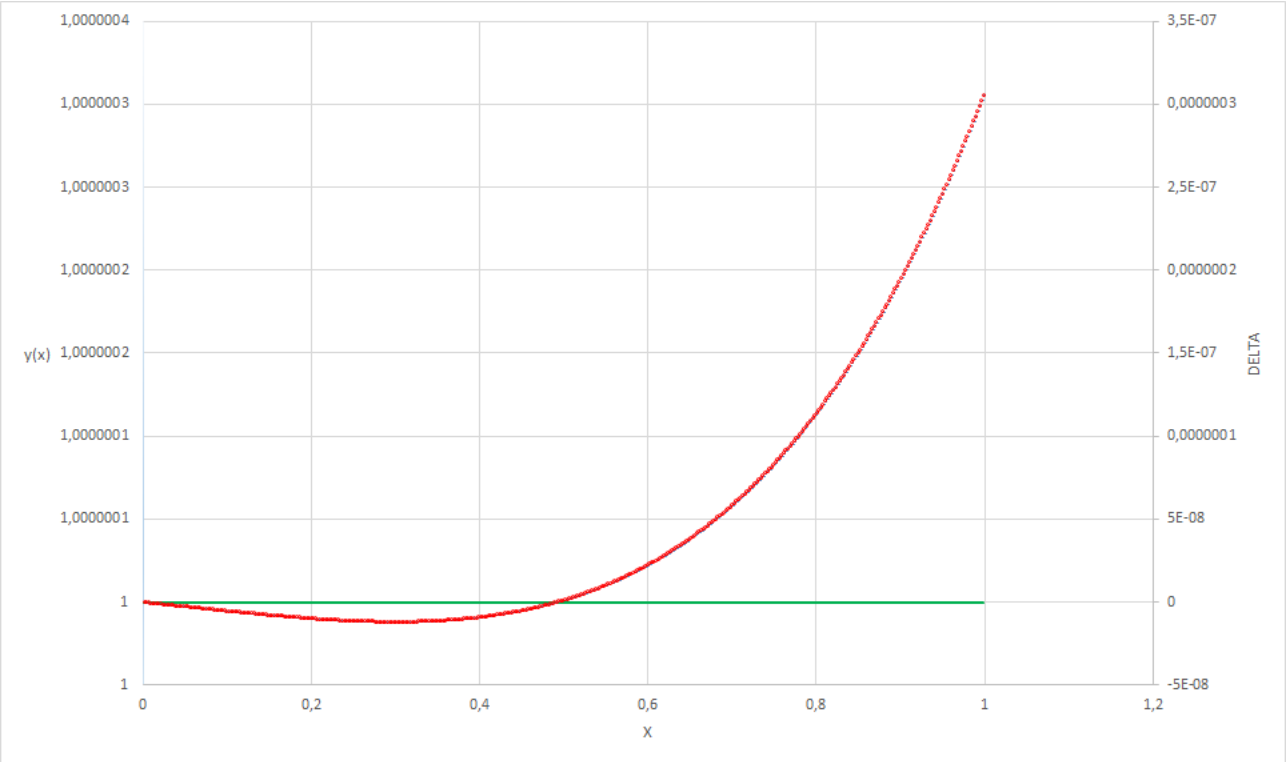

Рисунок 7 – Результаты эксперимента 2 при заданной точности $\Delta_2 = 1 \cdot 10^{-7}$

Если аппроксимировать погрешности решений интегрального уравнения численными методами квадратур по формуле прямоугольников в функцию $\Delta = \varepsilon(x), x \in [a, b]$, то на графиках можно видеть, что сама функция погрешности не изменяется вне зависимости от количества разбиений сетки.

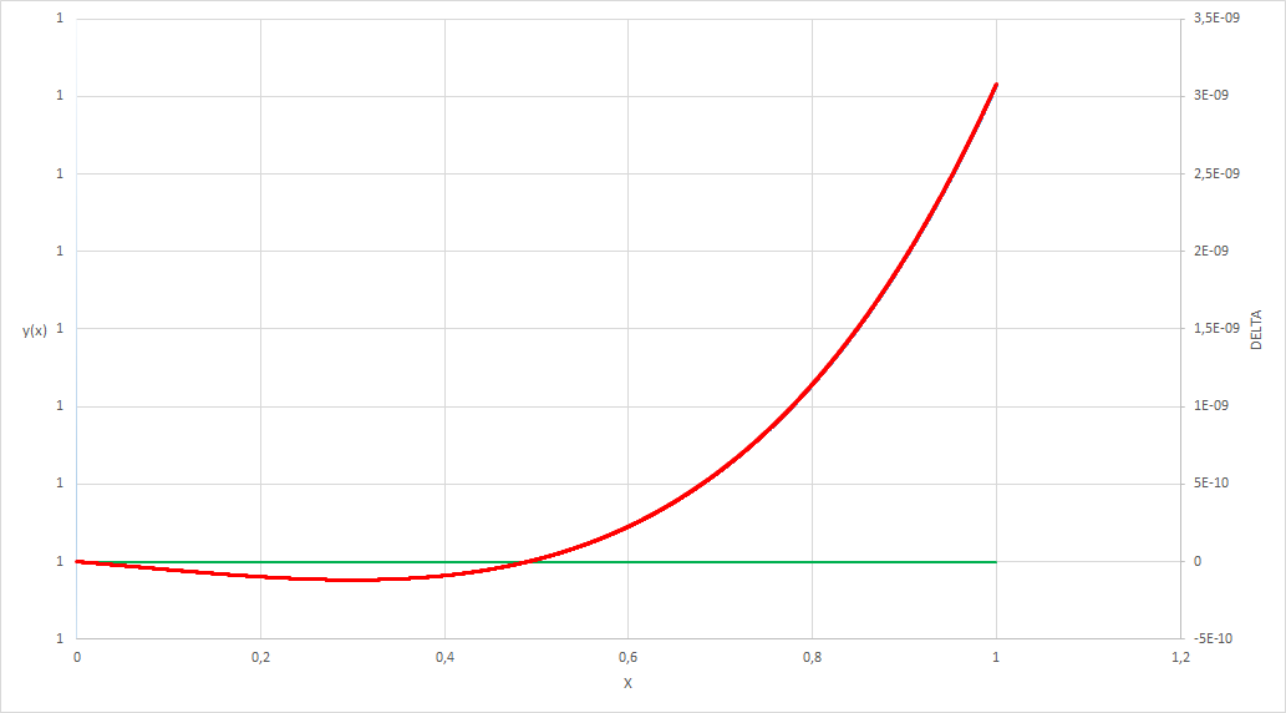

Рисунок 8 – Результаты эксперимента 2 при заданной точности $\Delta_3 = 1 \cdot 10^{-9}$

Получены следующие характеристики:

Заданные значения: $\begin{cases} \Delta_1 = 1 \cdot 10^{-3} \\ \Delta_2 = 1 \cdot 10^{-5} \\ \Delta_3 = 1 \cdot 10^{-6} \end{cases}$ Результат: $\begin{cases} n+ = 172 \\ n+ = 1706 \\ n+ = 5394 \end{cases} \begin{cases} \Delta_1 = -1{,}7 \cdot 10^{-3} \\ \Delta_2 = 2 \cdot 10^{-5} \\ \Delta_3 = 1{,}3 \cdot 10^{-5} \end{cases}$

Графики зависимостей, полученные при решении тестовых задач в эксперименте 3, представлены ниже (Рисунок 9, Рисунок 10, Рисунок 11).

Как видно из всех графических представлений решений и погрешностей решений интегральных уравнений Фредгольма II рода, характер распределения погрешности не зависит от степени сжатия сеток и всегда представляет собой определенную линейную функцию. При этом результаты эксперимента 1 и эксперимента 3 позволяют судить о возможном существовании такой комплексной функции, имеющей периодический характер, которая могла бы описывать погрешность вычислений в зависимости от степени сжатия сетки.

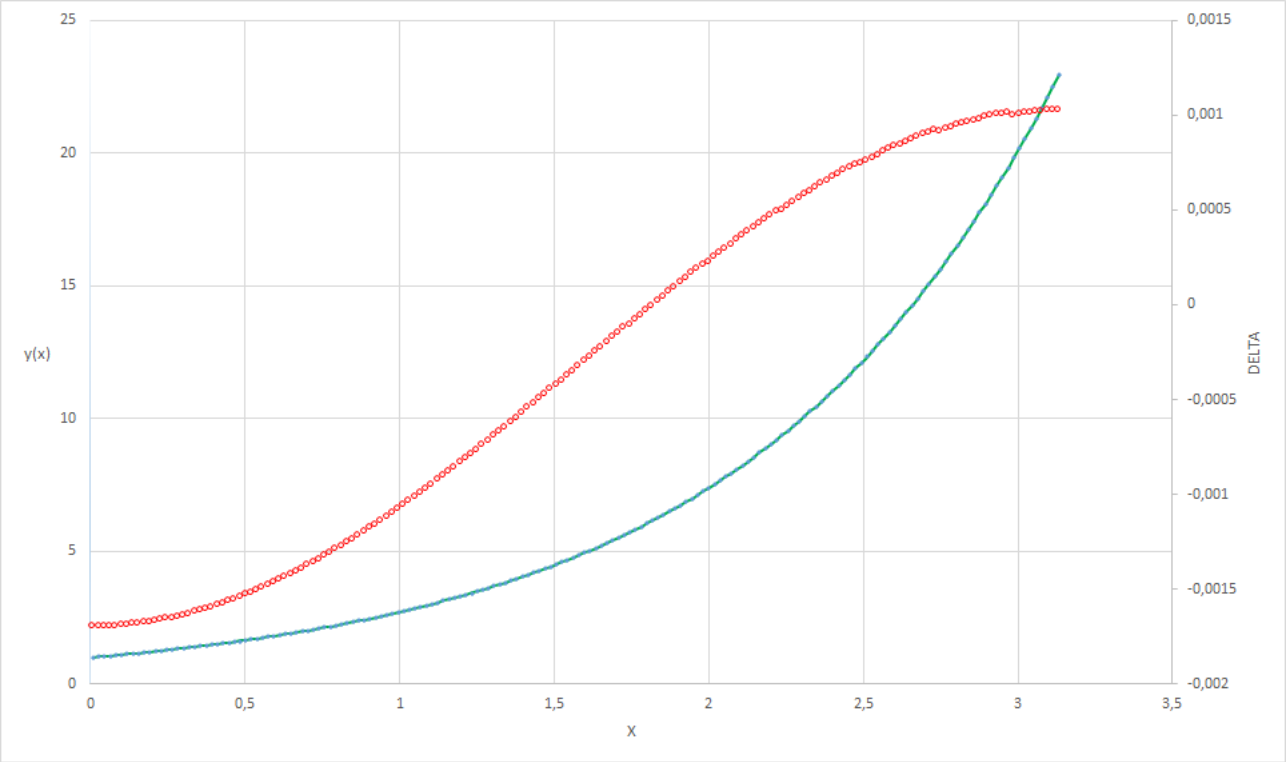

Рисунок 9 – Результаты эксперимента 3 при заданной точности $\Delta_1 = 1 \cdot 10^{-3}$

График погрешности вычислений носит четко выраженный периодический характер, т.е. дополняет свободный член интегрального уравнения [28].

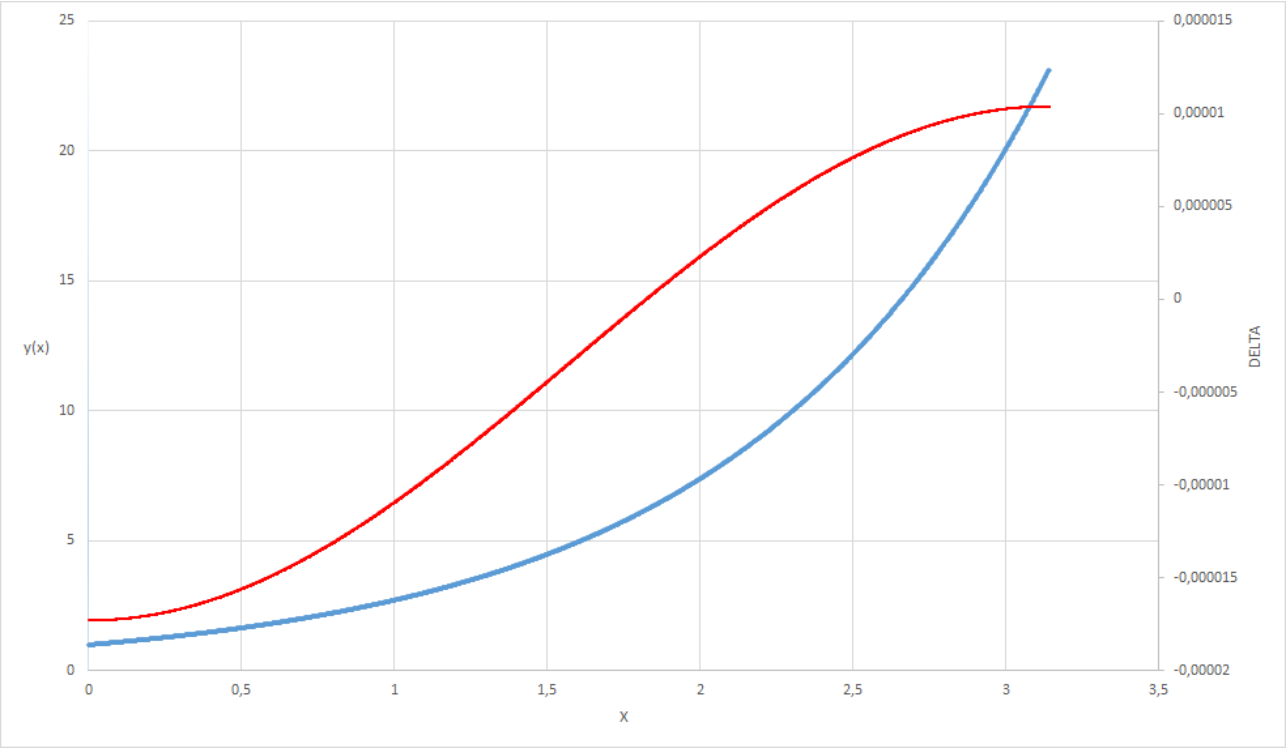

Рисунок 10 – Результаты эксперимента 3 при заданной точности $\Delta_2 = 1 \cdot 10^{-5}$

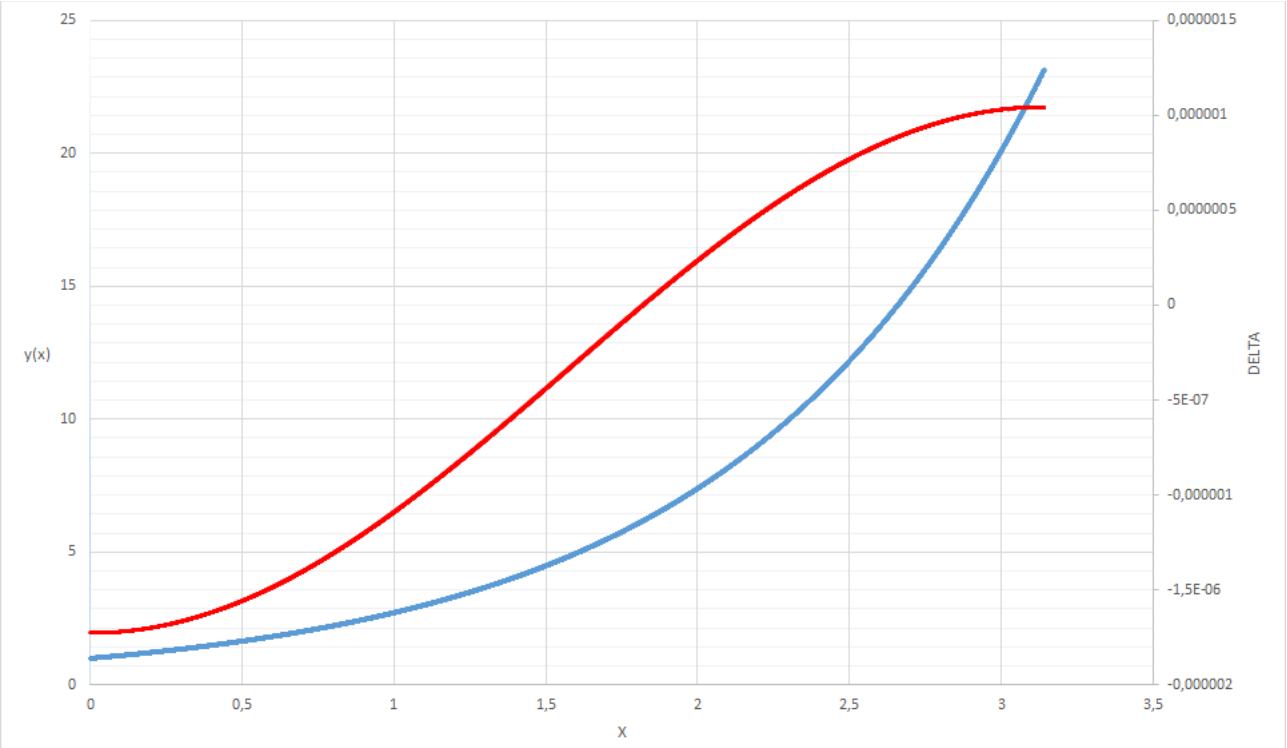

Рисунок 11 – Результаты эксперимента 3 при заданной точности $\Delta_3 = 1 \cdot 10^{-6}$

По формуле (2.7), являющейся следствием функции (2.1) не удается найти точное значение количества сжатий сетки для всех возможных экспериментов.

При этом, для некоторых задач, заданных определенными видами ядер и остаточных членов, разработанный алгоритм является вполне адекватным.

3.2 Модификация алгоритма

Из результатов анализа разработанного адаптивного алгоритма решения интегральных уравнений Фредгольма II рода методом механических квадратур с применением функции прямоугольников, приведенного в разделе 2.2 настоящей работы, следует, что данный алгоритм подлежит модификации.

Необходимо заметить, что при анализе численного метода, проведенного в разделе 2.1, были условно приняты некоторые допущения, которые и обусловили появление ошибок при реализации полученного алгоритма. Допущения заключались в подмене точного решения интегрального уравнения, решениями численным методом на каждом шаге увеличения количества разбиений сеток n. Поэтому основой ошибки явилось признание в качестве образцовой функции (2.1), выведенной ранее [6], [1].

Если вернуться на шаг назад и попытаться дать оценку факторам возникновения погрешности вычислений, то нетрудно заметить, что сама по себе искомая погрешность может быть получена путем разложения ряда получаемых значений (2.9) по следующей формуле:

$$\Delta \approx y_{n-j} - y_n, \tag{3.7}$$

где Δ – заданное значение погрешности искомых решений;

n – искомое количество разбиений сетки (степени сжатия сетки), при котором достигается решение с заданной погрешностью;

j – заданное количество пересечений сеток при разном n.

На примере проведенного эксперимента 1 для заданного интегрального уравнения Фридмана II рода (3.1) построим таблицу значений, получаемых в пересечениях сеток для $j = 2$ при $n \in [\,, 30]$.

Таблица 3.1 – Значения в точках пересечений сеток уравнения (3.1)

x	j=1	j=2	j=3	j=4
	n=2	n=6	n=10	n=14
	y(x)	y(x)	y(x)	y(x)
-1,570796327	4,21875	0,996207379	0,970688681	0,970588628
1,570796327	4,21875	0,996207379	0,970688681	0,970588628

x	j=5	j=6	j=7	j=8
	n=18	n=22	n=26	n=30
	y(x)	y(x)	y(x)	y(x)
-1,570796327	0,97059	0,970588235	0,970588235	0,970588235
1,570796327	0,97059	0,970588235	0,970588235	0,970588235

Если исходить из принятого приближения (3.7), то при $n = 26$ погрешность вычислений будет приблизительно равна $\Delta = y_{26} - y_{30} = 2{,}39808\text{E-}14$.

Заметим, что для двух точек пересечений каждое следующая точка пересечений j возникает при n_j, вычисляемом по следующей формуле:

$$n_j = n + (j-1) \cdot 2n = n(j-1), \tag{3.8}$$

где n – начально заданное количество пересечений сеток (начальное разбиение).

Введем новое обозначения для последующих вычислений:

$$i = n_j \tag{3.9}$$

Тогда для значений погрешности из формулы (3.7), можно записать следующую функцию:

$$\tilde{\Delta}_i = |y_i - y_{i+1}| \tag{3.10}$$

Покажем, как меняется величина погрешности решений уравнения (3.1) при $j = 1, 2, \ldots, 8$.

Таблица 3.2 – Величина погрешности при $j = 1, 2, \ldots, 8$

x	$\tilde{\Delta}_i$						
	n=2	n=6	n=10	n=14	n=18	n=22	n=26
-1,57079632	3,22254262	0,0255187	0,00010005	3,90841E-7	1,52672E-9	5,96401E-12	2,39808E-14
1,57079632	3,22254262	0,0255187	0,00010005	3,90841E-7	1,52672E-9	5,96301E-12	2,29816E-14

Анализируя полученные результаты, можно предположить, что изменение погрешности при последовательных итерациях в точках пересечений сеток имеет определенную функциональную зависимость. Для последующих

вычислений введем обозначение разности получаемых близлежащих значений погрешности:

$$\partial\tilde{\Delta}_i = \tilde{\Delta}_i - \tilde{\Delta}_{i+1} \tag{3.11}$$

Тогда, если предположить, что итерационный процесс нахождения погрешности вычислений имеет некоторую зависимость от второй производной искомой функции, т.е. сама функция является непрерывно дифференцируемой дважды, то в силу выражения (3.11) можно условно принять следующую функциональную зависимость (условимся называть данную величину ускорением схождения к точному решению):

$$f''_i(\xi_0) = \frac{\partial\tilde{\Delta}_i}{\partial i^2} \tag{3.12}$$

где ∂i – показатель увеличения степени сжатия сетки, равный $2n$ при n, равном начальному заданному количеству узлов сетки (в выше рассматриваемых примерах $n = 2$, поэтому величина $\partial i = const = 4$;

ξ_0 – некоторая точка, несовпадающая с точками сетки, но при большой степени сжатия сетки находящаяся в непосредственной близости к одной из точек $x \in [a, b]$, поэтому будем записывать данную точку без индекса.

Данное допущение имеет смысл лишь тогда, когда степень сжатия сетки довольно велика, либо при условии, что вычисление начальной погрешности ведется с учетом возмущений в отдельных точках, исходя из дисперсии распределения погрешности в узлах получаемых сеток [**Ошибка! Источник ссылки не найден.**]. На примере решения интегрального уравнения (3.1), исходя из значений получаемых погрешностей по формуле (3.7), записанных в виде таблицы (Таблица 3.2), найдем соответствующие значения по формуле (3.12).

Таблица 3.3 – Ускорение схождения к точному решению

	$f''_i(\xi)$					
x	i=1	i=2	i=3	i=4	i=5	i=6
-1,570796327	0,799255981	0,006354661	2,49157E-05	9,73285E-08	3,80189E-10	1,48501E-12
1,570796327	0,799255981	0,006354661	2,49157E-05	9,73285E-08	3,8019E-10	1,48501E-12

Нетрудно заметить, что данные величины ускорений схождения к точному решению обладают следующим свойством:

$$\lim_{i\to\infty}\frac{f''_i(\xi)}{f''_{i+1}(\xi)}=const \qquad (3.13)$$

Введем функциональную зависимость между величинами ускорений схождения:

$$\theta_i(f(x))=\frac{f''_i(\xi)}{f''_{i+1}(\xi)} \qquad (3.14)$$

Покажем, как изменяется отношение величин ускорений схождения собственных значений функции (3.1) при $i=1,2,\ldots,5$.

Таблица 3.4 – Отношение величин ускорений $\theta_i(f(x))$

	$\theta_i(f(x))$				
x	i=1	i=2	i=3	i=4	i=5
-1,570796327	125,7747667	255,046017	255,9962598	256,0002292	256,0185784
1,570796327	125,7747667	255,046017	255,9962618	255,9995544	256,0192513

Очевидно, что значение $\theta_i(f(x))$, полученное при $i=1$, значительно отличается от следующих значений, потому что в данной точке отсутствует информация о предыдущих разбиениях сетки, т.е. при $n=1$.

Для функциональной зависимости (3.14) можно ввести критерий относительной оценки сходимости, вычисляемый по формуле

$$\psi_i(\theta)=\left|\frac{\theta_{i+1}(f(x))-\theta_i(f(x))}{\theta_{i+1}(f(x))}\right| \qquad (3.15)$$

Таблица 3.5 – Значения относительных оценок сходимости

	$\psi_i(\theta)$			
x	i=1	i=2	i=3	i=4
-1,570796327	0,506854613	0,00371194	1,55051E-05	7,16715E-05
1,570796327	0,506854613	0,003711948	1,28616E-05	7,69351E-05

Поскольку из таблицы, представленной выше (Таблица 3.5), видно, что значения относительных оценок сходимости при $i=4$ выше, чем при $i=3$, то можно предположить, что решение интегрального уравнения численным методом квадратур при близком расположении к точному решению начинают «гулять» с разным ускорением, т.е. сила притяжения к действительному значению начинает изменяться. Тогда в качестве критерия окончания

итерационного процесса можно ввести величину, равную $\psi_i(\lambda) \le 0{,}001$. Тогда в окончательном виде критерий оценки можно записать следующим образом:

$$\psi_i(\lambda) = \left| \frac{\theta_{i+1}(f(x)) - \theta_i(f(x))}{\theta_{i+1}(f(x))} \right| < 0{,}001 \qquad (3.16)$$

Таким образом, при решении интегрального уравнения (3.1) итерационный процесс следует остановить на третьем шаге $i = 3$, при этом выполнится условие (3.16).

Поскольку до введения данного критерия оценки (3.16) все вычисления производились, исходя из абсолютных значений величин погрешностей, то сама по себе относительная оценка сходимости будет применима к любым заданным интегральным уравнениям.

Выразим функцию (3.7) в следующем виде:

$$\Delta_i \approx |y_{i-1} - y_i| \qquad (3.17)$$

Тогда можно предположить, что погрешность на i-ом шаге итераций будет иметь линейную зависимость от критерия оценки (3.16). Подставив в функцию (3.14) значения функций (3.11), (3.12), покажем, что это предположение верно:

$$\theta_i(f(x)) \approx \frac{f''_i(\xi)}{f''_{i+1}(\xi)} = \frac{\frac{\partial \tilde{\Delta}_i}{\partial i^2}}{\frac{\partial \tilde{\Delta}_{i+1}}{\partial i^2}} = \frac{\partial \tilde{\Delta}_i}{\partial \tilde{\Delta}_{i+1}}. \qquad (3.18)$$

Значение погрешности каждого следующего шага итераций можно вычислить, исходя из уравнения (3.18) по формуле

$$\tilde{\Delta}_{i+1} \approx \frac{\tilde{\Delta}_i}{\theta_I(f(x))}, \qquad (3.19)$$

где индексом I обозначен номер итерации при котором значение $\theta_i(f(x))$ удовлетворяет условию (3.16).

На основе данных выводов получается, что нет необходимости на каждом шаге полностью осуществлять процесс построения собственных значений интегрального уравнения Фредгольма II рода, достаточно провести некоторое количество итерационных вычислений, вычислить значение ускорений сходимости собственных значений функции и по полученным результатам

рассчитать шаг итерации, на котором будет обеспечено выполнение заданного критерия оценки погрешности, используя неравенство

$$|\Delta_n| > |\tilde{\Delta}_i|, \qquad (3.20)$$

где Δ_n – абсолютное значение заданной погрешности вычислений.

На основании результатов, полученных ранее (при проведении эксперимента 1), необходимо оценить адекватность применения неравенств (3.16), (3.20):

Заданные значения: $\begin{cases} \Delta_1 = 1 \cdot 10^{-3} \\ \Delta_2 = 1 \cdot 10^{-5} \\ \Delta_3 = 1 \cdot 10^{-7} \end{cases}$ Результат: $\begin{cases} n+ = 116 \\ n+ = 1146 \\ n+ = 11444 \end{cases} \begin{cases} \Delta_1 = -1 \cdot 10^{-13} \\ \Delta_2 = -3 \cdot 10^{-13} \\ \Delta_3 = -1 \cdot 10^{-12} \end{cases}$

Теперь в качестве заданных значений необходимо принять полученные в эксперименте 1 результаты. При оценке по неравенству (3.16) получено значение относительной величины ускорения сходимости $\theta_i(x) = 255{,}9962618$ при значении $\Delta_1 = -1 \cdot 10^{-13}$. По формуле (3.19) вычислим примерные значения погрешностей, которые были бы получены на следующих шагах итераций.

Таблица 3.6 – Вычисленные значения погрешностей

x	$\lvert\tilde{\Delta}_i\rvert$			
	4	5	6	7
-1,57079633	3,90841E-07	1,52674E-09	5,96393E-12	2,3297E-14
1,570796327	3,90841E-07	1,52674E-09	5,96393E-12	2,3297E-14

Из анализа результатов вычислений, представленных в таблице выше (Таблица 3.6), следует, что согласно формуле (3.19) полученный результат достигнет требуемых критериев оценки погрешности при $i = 7$, при этом на основании формулы (3.8) при $n = 2$ получим значение n^+:

$$n^+ = n \cdot (j - 1) = 2 \cdot (7 - 1) = 26$$

Таким образом, критерии оценки погрешности выполнятся при начальном разбиении сетки на двадцать шесть отрезков.

Анализируя полученные данные, можно установить, что оценку критерия окончания итерационного процесса (3.16) допустимо на порядок ухудшить и требовать выполнения неравенства

$$\psi_i(\theta) = \left| \frac{\theta_{i+1}(f(x)) - \theta_i(f(x))}{\theta_{i+1}(f(x))} \right| < 0{,}005 \qquad (3.21)$$

При этом, если обратиться к результатам, полученным при проведении эксперимента 1 по изначально разработанному алгоритму (Рисунок 3), то можно сделать вывод о том, что по полученным оценкам погрешности в точках пересечений начальной сетки при последующих её сжатиях допустимо получение адекватной оценки максимальной погрешности во всех точках сетки при конечных сжатиях.

Введем некоторые новые обозначения. Так, необходимо выделить диапазон изменений (амплитуду) собственных значений во всех точках сетки для значения I итерации, при котором однозначно выполняется условие (3.21):

$$M_1 = \max_{x \in [a,\, b]} f(x) - \min_{x \in [a,\, b]} f(x) \qquad (3.22)$$

Дополнительно потребуется диапазон изменений собственных значений в точках пересечений сеток:

$$M_2 = \max_{x \in [a,\, b]} f_i(x) - \min_{x \in [a,\, b]} f_i(x) \qquad (3.23)$$

Можно однозначно судить, что во всем диапазоне максимальная погрешность вычислений не превысит основную погрешность вычислений (при уровне доверительной вероятности $P = 0{,}99$). Тогда необходимо вычислить среднеквадратическое отклонение по точкам пересечений сеток, в которых производилось вычисление схождения процесса притягивания точек по формуле (3.21):

$$\Omega = 2\sqrt{\frac{\sum_{j=1}^{m} \Delta_{ij}^{\,2}}{m-1}}, \qquad (3.24)$$

где Δ_{i1}, Δ_{i2} – вычисленные по формуле (3.17) значения погрешностей в точках пересечений сеток на шаге итерации, при котором выполняется условие (3.21); m – количество точек, в которых сетки пересекаются (в рассматриваемом варианте алгоритма используется $m = 2$).

В дальнейшем в качестве оценки глобальной погрешности на шаге итераций, при котором выполняется условие (3.21) необходимо ввести следующую величину:

$$\Gamma = I \cdot \frac{\Omega \cdot M_1}{M_2}, \tag{3.25}$$

где I – номер шага итерации, при котором выполняется неравенство (3.21).

При вычислении максимально допустимой погрешности каждого следующего шага итераций в качестве начального параметра глобальной погрешности следует принимать значение, вычисляемое по формуле (3.25). Таким образом, формула (3.19) на начальном шаге приближений будет выглядеть следующим образом:

$$\tilde{\Delta}_{i+1} \approx \frac{\Gamma}{\theta_I \left(f \left(x \right) \right)} \tag{3.26}$$

Тогда результаты, полученные ранее (Таблица 3.6), несколько разойдутся с получаемыми в следствии решения по формуле (3.26):

Таблица 3.7 – Вычисленные значения максимально допустимых погрешностей

	i				
	3	4	5	6	7
$\left\|\tilde{\Delta}_i\right\|$	0,000565998	2,2192E-06	8,70117E-09	3,41161E-11	1,33764E-13

Проверим адекватность данного алгоритма относительно действительных собственных значений функции на седьмом шаге итераций:

$$\max\left|\Delta_i\right| = 1{,}3145E-13.$$

Значение вычисленной погрешности по данному алгоритму в третьем знаке превышает действительную оценку погрешности, но это полностью отвечает критерию адекватности относительно заданного условия в эксперименте 1.

При проведении вычислений по данному алгоритму относительно эксперимента 3, заданного интегральным уравнение Фредгольма II рода (3.5), условие (3.21) достигается только на шаге итераций $I = 27$, т.е. при $n = 106$. При этом оценка глобальной погрешности на шаге $I = 27$ дает следующий результат:

$$\Gamma = 0{,}00712788$$

Необходимо заметить, что это на порядок выше вычисленных приблизительных значений погрешности:

$$\tilde{\Delta}_{106} \approx \begin{cases} 0{,}00024410 \\ 0{,}00011795 \end{cases}$$

Последовательно вычисляя приблизительные значения погрешностей на следующих шагах итераций, получим следующую таблицу значений:

Таблица 3.8 – Приблизительные значения погрешностей

	i					
	28	29	30	31	32	33
$\lvert\tilde{\Delta}_i\rvert$	0,006178106	0,005354891	0,004641367	0,004022918	0,003486876	0,00302226

Проверяя достоверность полученного результата получаем следующее значение действительной погрешности на шаге итерации $I = 33$:

$$\max|\Delta_i| = 0{,}002971$$

Анализ общего результата реализации алгоритма и его модификации приведен в таблице ниже.

Таблица 3.9 – Анализ алгоритма и его модификаций

Задание	Алгоритм (2.7)		Модификация (3.21)		Модификация (3.16)	
			$\psi_i(\theta) = 0{,}005$		$\psi_i(\theta) = 0{,}001$	
	степень сжатия сетки	погрешность результата	степень сжатия сетки	погрешность результата	степень сжатия сетки	погрешность результата
$\lvert\Delta_i(x)\rvert$	n^+	$\max\lvert\Delta_i\rvert$	n^+	$\max\lvert\Delta_i\rvert$	n^+	$\max\lvert\Delta_i\rvert$
1	2	3	4	5	6	7
3,000E-03	–		130	0,002971243	248	0,000829841
1,000E-03	172	0,001697431	–			
1,000E-04	–		222	0,001018957	382	0,00034415
1,000E-05	1706	1,72553E-05	290	0,000597137	530	0,000178783
1,000E-06	5394	1,72658E-06	–			
1,000E-07	–				818	7,50539E-05

Выводы по главе

В ходе анализа, проведенного на основании результатов применения модификаций алгоритма (Таблица 3.9), а также общих выкладок, описанных в данной главе, были сделаны следующие выводы.

Модификация (3.21) применима для численного решения уравнений Фредгольма II рода при низких требованиях к критерию погрешности

результата. Такая модификация требует достаточно большого количества итераций, при этом для уравнений, обладающих высокой степенью гладкости, такая модификация алгоритма оказывается неприменимой ввиду того, что количество итераций, необходимое для выполнения условия (3.16) может значительно превышать требуемое из-за незначительных ограничений на погрешность. При высоких заданных требованиях к погрешности в результате решения уравнений, обладающих высокой степенью гладкости, наблюдается неадекватность данной модели и, как следствие, значительное расхождение получаемых результатов относительно действительных значений интегрального уравнения.

Модификация (3.16) отличается более точными результатами вычислений, при этом следует заметить, что для уравнений, обладающих низкой степенью гладкости, данная модификация алгоритма может давать положительный результат уже на малом количестве итераций. Таким образом, данную модификацию алгоритма следует использовать для проверки по степени сходимости (ускорения сходимости) на предмет степени гладкости, и в случае если таких итераций для достижения положительного результата оценки по критерию (3.16) необходимо выполнить менее $I = 10 \cdot n$, то дальнейшее решение следует выполнять именно по данному алгоритму.

При невыполнении положений, указанных в предыдущем абзаце, следует воспользоваться начальным алгоритмом, при этом результаты, полученные на итерации $I = 10 \cdot n$, будут обладать высокой степенью надежности. Однако следует заметить, что для оценки глобальной погрешности по формуле (2.7) необходимо воспользоваться равенствами (3.24), (3.25), (3.26).

4 СРАВНЕНИЕ ЧИСЛЕННЫХ МЕТОДОВ МЕХАНИЧЕСКИХ КВАДРАТУР И АПРОКСИМАЦИИ ЯДРА ВЫРОЖДЕННЫМ

4.1 Метод вырожденных ядер

Если задача состоит в решении интегрального уравнения с невырожденным ядром, но при этом оно имеет достаточную гладкость, то такое ядро можно аппроксимировать вырожденным, при этом одним из самых простых методов является разложение ядра в ряд Тейлора.

Введем понятие вырожденного ядра. Ядро называется вырожденным, если оно может быть представлено в следующем виде

$$K(x,s) = \sum_{i=1}^{m} \alpha_i(x)\beta_i(s), \tag{4.1}$$

где $\alpha_i(x)_{i=1}^{m}$, $\beta_i(s)_{i=1}^{m}$ – это линейно независимые системы непрерывных на отрезке $[a,b]$ функций.

Подставив выражение (4.1) в уравнение Фредгольма II рода (1.7), получим интегральное уравнение с вырожденным ядром:

$$y(x) - \lambda \int_a^b \left[\sum_{i=1}^{m} \alpha_i(x)\beta_i(s) \right] y(s)ds = f(x),\ x \in [a,b]. \tag{4.2}$$

Поменяем местами операции интегрирования и суммирования:

$$y(x) - \lambda \sum_{i=1}^{m} \alpha_i(x) \int_a^b \beta_i(s)y(s)ds = f(x),\ x \in [a,b]. \tag{4.3}$$

Введем обозначения:

$$c_i = \int_a^b \beta_i(s)y(s)ds,\ i = 1, 2, 3, \ldots, m. \tag{4.4}$$

Тогда (4.3) запишется в виде

$$y(x) = f(x) + \lambda \sum_{i=1}^{m} c_i \alpha_i(x),\ x \in [a,b]. \tag{4.5}$$

В уравнении (4.5) константы c_i являются неизвестными интегралами от искомой функции. Для их вычисления необходимо построить СЛАУ. Для этого подставим (4.5) в (4.2), в результате получим:

$$\sum_{i=1}^{m}\alpha_i(x)\left\{c_i-\int_a^b\beta_i(s)\left[f(s)+\lambda\sum_{j=1}^{m}c_j\alpha_j(s)\right]ds\right\}=0,\ x\in[a,b]. \qquad (4.6)$$

Поскольку, как отмечалось выше, система функций $\{\alpha_i(x)\}_{i=1}^{m}$ является линейно независимой, то коэффициенты в линейной комбинации (4.6) равны нулю, то есть

$$c_i-\int_a^b\beta_i(s)\left[f(s)+\lambda\sum_{j=1}^{m}c_j\alpha_j(s)\right]ds=0,\ i,\ j=1,2,\ldots,m \qquad (4.7)$$

или

$$c_i-\lambda\sum_{j=1}^{m}c_j\int_a^b\alpha_j(s)\beta_i(s)ds=\int_a^b\beta_i(s)f(s)ds,\ i,\ j=1,2,\ldots,m. \qquad (4.8)$$

Введем обозначения коэффициентов равенства (4.8):

$$a_{ij}=\int_a^b\alpha_j(s)\beta_i(s)ds,\ i,\ j=1,2,\ldots,m, \qquad (4.9)$$

$$f_i=\int_a^b\beta_i(s)f(s)ds,\ i=1,2,\ldots,m. \qquad (4.10)$$

Равенство (4.8) запишем в виде системы линейных алгебраических уравнений относительно неизвестных c_i:

$$c_i-\lambda\sum_{j=1}^{m}a_{ij}c_j=f_i,\ i,\ j=1,2,\ldots,m. \qquad (4.11)$$

Или в другом виде:

$$\begin{pmatrix}1-\lambda a_{11} & -\lambda a_{12} & \ldots & -\lambda a_{1m}\\ -\lambda a_{21} & 1-\lambda a_{22} & \ldots & -\lambda a_{2m}\\ \ldots & \ldots & \ldots & \ldots\\ -\lambda a_{m1} & -\lambda a_{m2} & \ldots & 1-\lambda a_{mm}\end{pmatrix}\cdot\begin{pmatrix}c_1\\ c_2\\ \ldots\\ c_m\end{pmatrix}=\begin{pmatrix}f_1\\ f_2\\ \ldots\\ f_m\end{pmatrix}. \qquad (4.12)$$

Решение интегрального уравнения Фредгольма II рода с вырожденным ядром сводится к вычислению интегралов (4.9), (4.10) и решению СЛАУ(4.12). Затем решение уравнения (4.2) можно записать в аналитическом виде, используя равенство (4.5).

4.2 Аппроксимация ядра вырожденным

Вернемся к задаче решения интегрального уравнения Фредгольма II рода с невырожденным ядром, но при этом имеющим достаточную гладкость. Для примера решим уравнение

$$y(x) = \int_0^{1/2} \exp(-x^2 s^2) y(s) ds + 1,\ x \in \left[0, \frac{1}{2}\right]. \tag{4.13}$$

Нахождение точного решения для (4.13) проблематично. Найдем приближенное решение методом аппроксимации ядра вырожденным ядром. Для этого разложим ядро уравнения в ряд. Ядро аппроксимируется известным разложением

$$e^x = 1 + \sum_{n=1}^{\infty} \frac{x^n}{n!}. \tag{4.14}$$

Из уравнения (4.13), раскладывая в ряд до $n = 2$, получаем вырожденное ядро следующего вида:

$$K^{(2)}(x, s) = 1 - x^2 s^2 + \frac{x^4 s^4}{4}. \tag{4.15}$$

Отсюда следует, что приближенное решение уравнения (4.13) является решением уравнения

$$\tilde{y}(x) = \int_0^{1/2} \left[1 - x^2 s^2 + \frac{x^4 s^4}{4}\right] y(s) ds + 1,\ x \in \left[0, \frac{1}{2}\right]. \tag{4.16}$$

Следовательно,

$$\tilde{y}(x) = 1 + c_1 + c_2 x^2 + c_3 x^4, \tag{4.17}$$

где коэффициенты c_i – определяются равенствами

$$c_1 = \int_0^{1/2} y(x) dx,\ c_2 = -\int_0^{1/2} x^2 y(x) dx,\ c_3 = \frac{1}{2} \int_0^{1/2} x^4 y(x) dx. \tag{4.18}$$

На основании равенств (4.17), (4.18) построим систему линейных уравнений:

$$\begin{cases} c_1 = \frac{1}{2} + \frac{1}{2}c_1 + \frac{1}{24}c_2 + \frac{1}{160}c_3 \\ c_2 = -\frac{1}{24} - \frac{1}{24}c_1 - \frac{1}{160}c_2 - \frac{1}{896}c_3 \\ c_3 = \frac{1}{320} + \frac{1}{320}c_1 + \frac{1}{1792}c_2 + \frac{1}{9216}c_3 \end{cases} \quad (4.19)$$

Решение системы (4.19) даст следующий результат:

$$c_1 = 0{,}993199\,,\ c_2 = -0{,}08254\,,\ c_3 = 0{,}006183\,. \quad (4.20)$$

Подставив полученные константные значения интегралов (4.20) в равенство (4.17), запишем решение в аналитическом виде:

$$\tilde{y}(x) = 1{,}993199 - 0{,}08254x^2 + 0{,}006183x^4. \quad (4.21)$$

Для решений интегральных уравнений, полученных методом аппроксимации ядра вырожденным, используются оценки погрешности, выражаемые следующими неравенствами:

$$\int_a^b \left| K(x,s) - K^{(n)}(x,s) \right| ds \le \varepsilon, \left| f(x) - f_n(x) \right| \le \delta\,. \quad (4.22)$$

Таким образом, само по себе нахождение погрешности решения интегрального уравнения Фредгольма II рода методов аппроксимации ядра вырожденным является тривиальной задачей, требующей нахождения решений при различных *n*.

Найдем решение уравнения (4.13) при $n = 3$.

$$\tilde{y}_3(x) = 1 + c_1 + c_2x^2 + c_3x^4 + c_4x^6, \quad (4.23)$$

где коэффициенты c_i – определяются равенствами

$$c_1 = \int_0^{1/2} y(x)\,dx,\ c_2 = -\int_0^{1/2} x^2 y(x)\,dx,\ c_3 = \frac{1}{2}\int_0^{1/2} x^4 y(x)\,dx,\ c_4 = -\frac{1}{6}\int_0^{1/2} x^6 y(x)\,dx \quad (4.24)$$

Решение СЛАУ, составленной на основании равенств (4.23), (4.24), даст следующий результат:

$$c_1 = 0{,}993198,\ c_2 = -0{,}08254,\ c_3 = 0{,}006183,\ c_4 = -0{,}00037\,. \quad (4.25)$$

Подставив полученные константные значения интегралов (4.25) в равенство (4.23), запишем решение в аналитическом виде:

$$\tilde{y}_3(x) = 1{,}993198 - 0{,}08254\,x^2 + 0{,}006183\,x^4 - 0{,}00037\,x^6. \quad (4.26)$$

На основе проведенных вычислений получаем:

$$|\tilde{y}_3(x) - \tilde{y}_2(x)|_{\max} = 0{,}00037x^6 \ . \qquad (4.27)$$

Максимума данная величина будет достигать при $x = 0{,}5$:

$$|\tilde{y}_3(x) - \tilde{y}_2(x)|_{\max} = 0{,}00037(0{,}5)^6 = 5{,}74716E-06.$$

Таким образом, можно сделать вывод, что при незначительном увеличении количества вычислений уточнение решения будет довольно малой величиной, что говорит о высокой точности такого решения уже при $n = 3$.

4.3 Сравнительный анализ рассмотренных методов

С целью сравнения получаемых результатов найдем решение интегрального уравнения Фредгольма (4.13) численным методом механических квадратур по формуле прямоугольников. Для данного решения будем использовать алгоритм, разработанный в третьей главе настоящей работы.

Найдем решение при задании критерия погрешности вычисления $\Delta_1 = 1{,}0E-10$ и $\Delta_2 = 1{,}0E-06$. При этом решение при значении погрешности, равном Δ_1, будем считать точным и относительно него вычислим приближенные значения погрешности решения, полученного методом аппроксимации ядра вырожденным, и решения, полученного методом механических квадратур, при значении погрешности Δ_2.

При заданном значении погрешности $\Delta_1 = 1{,}0E-10$ вычислений методом механических квадратур по разработанному алгоритму была решена СЛАУ размерностью 5373×5373. Объем выделяемой памяти под хранение и обработку результатов составил 220 Мб. Время решения составило 1 минут 20 секунд.

При $\Delta_2 = 1{,}0E-06$ размерность СЛАУ - 130×130. Объем выделяемой памяти под хранение и обработку результатов составил 132 кб.

Представим результаты в графическом виде.

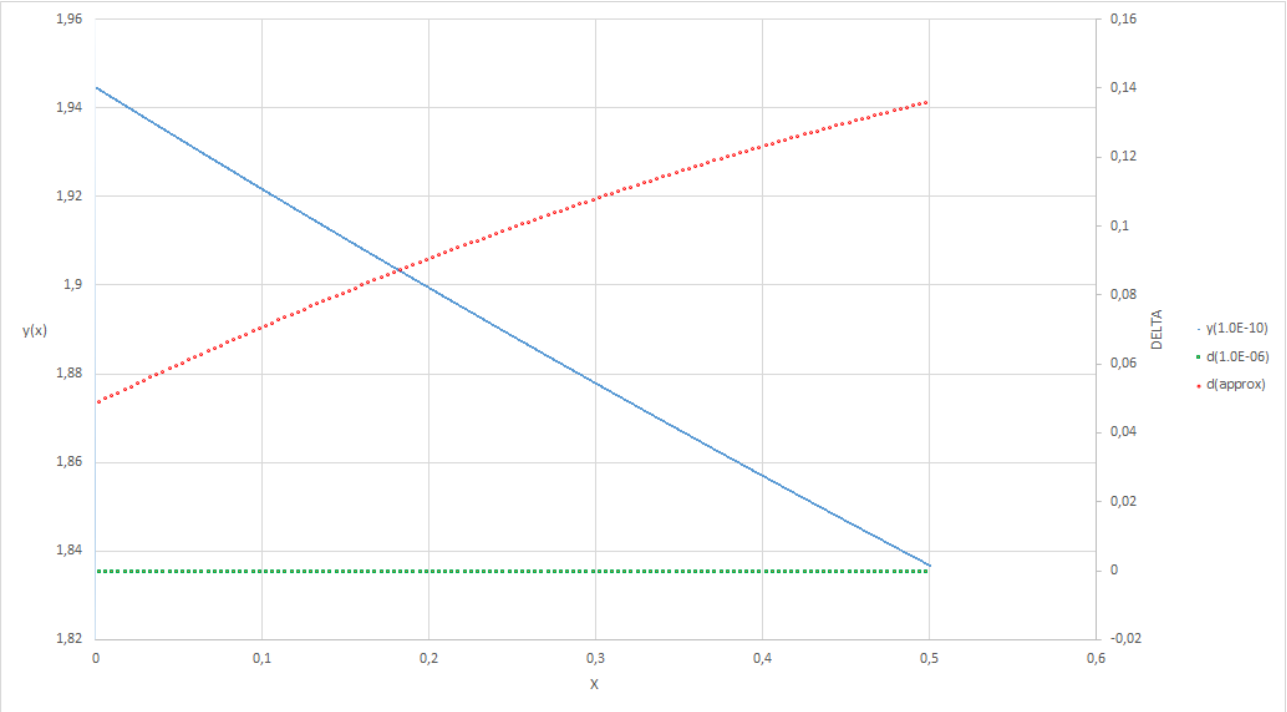

Рисунок 12 - Сравнение результатов вычислений методом механических квадратур и методом аппроксимации вырожденным ядром

На представленном графике погрешности вычислений решений интегрального уравнения методом механических квадратур при заданном значении $\Delta_2 = 1{,}0E-06$ обозначены зелеными маркерами. При этом из графического представления видно, что все значения погрешности полученных результатов лежат в околонулевой области. Максимальное отклонение от решения при заданном $\Delta_1 = 1{,}0E-10$ составило $\varepsilon_{MK} = -2{,}00999\text{E-}05$.

Значения погрешности решения интегрального уравнения методом аппроксимации вырожденным ядром (4.26) относительно решения методом механических квадратур при заданном $\Delta_1 = 1{,}0E-10$ равно $\varepsilon_{ABЯ} = 0{,}135917$.

Поскольку нами было получено, что при увеличении размерности СЛАУ на единицу при построении решения интегрального уравнения методом аппроксимации вырожденным ядром максимальное изменение результата составило $\left|\tilde{y}_3(x) - \tilde{y}_2(x)\right|_{\max} = 5{,}74716E-06$, то можно сделать вывод, что дальнейшее увеличение размерности СЛАУ не будет вносить значительных изменений в получаемые результаты.

Таким образом, минимальный размер СЛАУ при решении интегрального уравнения методом аппроксимации вырожденным ядром для достижения точности, полученной при решении методом механических квадратур, составит

$$\left|\frac{\varepsilon_{MK}}{\left|\tilde{y}_3(x)-\tilde{y}_2(x)\right|_{\max}}\right| = \left|\frac{-2{,}00999E-05}{5{,}74716E-06}\right| = 3{,}5\,.$$

Следовательно, необходимо решать интегральное уравнение методом аппроксимации вырожденным ядром при $n=6$.

Для более точного вычисления размерности СЛАУ, необходимой для получения результата с точностью не хуже $\Delta_2 = 1{,}0E-06$, можно каждое приращение по методу аппроксимации вырожденным ядром приблизительно вычислить путем разложения в ряд:

$$\delta_n \approx \left|\tilde{y}_3(x)-\tilde{y}_2(x)\right|_{\max} + \sum_{n=4}^{m} \frac{(-1)^n \cdot (b-a)^{4n}}{n!\cdot(2n+1)}\,. \qquad (4.28)$$

Рассчитывая таким образом приблизительное значение погрешности при $n=6,7,8,\ldots 130,$ получаем, что решение интегрального уравнения методом аппроксимации вырожденным ядром при $n=6$ по значению абсолютной погрешности практически не отличается от решения при $n=130$. Погрешность при этом приблизительно составляет $\delta_{130} \approx 0{,}135911.$ Такое положение указывает на существование неисключаемой систематической погрешности, избавиться от которой увеличением размерности выстраиваемой СЛАУ не удается.

ЗАКЛЮЧЕНИЕ

1. В общепринятых положениях по применению численных методов интегрирования при решении уравнений Фредгольма (I и II рода) довольно редко прибегают к действительному оцениванию получаемых результатов с точки зрения глобальной погрешности таких результатов. Однако такая оценка имеет непосредственную практическую ценность при проведении исследований в различных областях науки и техники.

2. Решение интегральных уравнений при невозможности проведения экспериментов и получения таких решений классическими методами практически исключает возможность такой оценки. Соответственно, требуется разработка численных методов, обладающих свойствами самодиагностики и исключения неопределенностей получаемых результатов.

3. Высокая степень достоверности получаемых результатов могла бы положить начало разработке новых численных методов интегрирования, которые, в свою очередь, могли бы обеспечить сведение к минимуму вероятности возникновения ошибки вычислений.

4. Для реализации этих положений требуется проведение дополнительных экспериментов (исследований) с целью оценки достоверности полученных в работе результатов и разработки возможных вариантов адаптации алгоритма решения интегральных уравнений методом механических квадратур по формулам прямоугольников к конкретным поставленным задачам.

5. В ходе работы были проведены вычислительные эксперименты и сделаны выводы на основании анализа полученных результатов в отношении решений интегральных уравнений, полученных классическими методами, а также в отношении решений, полученных методом аппроксимации ядра вырожденным. При этом необходимо заметить, что разработанный алгоритм на каждом этапе вычислений производит оценку глобальной погрешности вычислений, благодаря чему он может быть использован при проведении анализа точности получаемых решений другими методами численного

интегрирования (метод вырожденных ядер, метод наименьших квадратов, метод Галеркина-Петрова, метод коллокации, метод простой итерации, метод моментов и другие).

6. Тот факт, что современные вычислительные устройства, методы поточных вычислений, создаваемые нейронные сети и центры уже сейчас обладают практически неограниченными возможностями в части объемов обрабатываемой информации, делает возможным решение интегральных уравнений с очень высокой степенью сжатия сеток. В работе рассмотрен и применен один из самых простых методов – решение посредством формулы прямоугольников, поскольку в большинстве ситуаций он обладает более качественным уровнем надежности получаемых результатов в отличие от других формул, используемых при решении методами механических квадратур, хотя и требует большого объема выделяемых машинных ресурсов.

7. Поскольку погрешность вычислений методом квадратур при использовании формулы прямоугольников заведомо обладает свойством линейности, то при уже небольшом количестве итераций становится возможным построение функции зависимости погрешности от степени сжатия сетки интерполяционными методами в целях оценки погрешности получаемых решений интегрального уравнения на более высоких уровнях степени сжатия сеток.

8. Также следует заметить, что получение информации о погрешности вычислений с высокой степенью надежности дает возможность получения недостающих данных другими численными методами матричного интерпретирования, такими как метод LU-разложения либо все чаще применяющийся в настоящее время метод QR-вращений. Таким образом, становится возможным получение относительно точного конечного решения уже при малом разбиении сетки. Все вышесказанное свидетельствует о том, что разработанный алгоритм имеет высокую практическую значимость и, как следствие, может являться основой для дальнейших модификаций.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

Научная и методическая литература

1. Авхадиев Ф.Г. – Численные методы анализа: учебное пособие / Ф.Г. Авхадиев – Казань: Казанский (Приволжский) федеральный университет, 2013. – 126 с.

2. Арушанян И.О. – Материалы семинарских занятий по курсу “Методы вычислений” / О.Б. Арушаняна, Е.В. Чижонков – М.: Изд-во ЦПИ при механико-математическом факультете МГУ, 1999.

3. Бахвалов Н.С. – Численные методы: учебник / Н.С. Бахвалов – М.: Наука, 1975.

4. Бахвалов Н.С. – Численные методы: учебник / Н.С. Бахвалов, Н.П. Жидков, Г.М. Кобельков – М.: Наука, 1987.

5. Буханько А.А. – Чостковская О.П., Приближенные методы решения краевых задач для обыкновенных дифференциальных уравнений, уравнений с частными производными и интегральных уравнений: учеб.-метод. пособие / А.А. Буханько, О.П. Чостковская – Самара: Изд-во Самар. гос. аэрокосм. ун-та, 2011. – 68 с.: ил.

6. Вержбицкий В.М. – Численные методы. Математический анализ и обыкновенные дифференциальные уравнения: Учебное пособие для вузов / В.М. Вержбицкий – М.: Высш. шк., 2001 – 382 с.:ил.

7. Верлань А.Ф. – Интегральные уравнения, методы, алгоритмы, программы: справочник / А.Ф. Верлань, В.С. Сизиков – Киев, 1986.

8. Власова Е.А. – Приближенные методы математической физики: учебное пособие – Е.А. Власова, В.С. Зарубин, Г.Н. Кувыркин – М. Изд-во МГТУ; 2006. – 700с.

9. Каденова З.А. – Регуляризация и единственность решений линейных интегральных уравнений Фредгольма первого рода: диссертация / З.А. Каденова – Ош, Ошский технологический университет; 2006.

10. Калинина, В.Н. – Теория вероятностей и математическая статистика: учебник для бакалавров / В.Н. Калинина. – М.: Юрайт, 2013. – 472 с.

11. Калиткин Н.Н. – Численные методы: учебник / Н.Н. Калиткин – М.: Наука, 1978.

12. Канторович Л.В. – Функциональный анализ: учеб.пособие / Л.В. Канторович, Г.П. Акилов – СПб.: Невский Диалект, 2004. – 816 с.

13. Карчевский Е.М. – Численные методы решения интегральных уравнений и комплекс программ на языке Matlab: учебное пособие – Е.М. Карчевский – Казань: Казанский университет, 2017. – 61с.

14. Каханер Д. – Численные методы и программное обеспечение: учебное пособие / Д. Каханер, К. Моулер, С. Нэш – М.: Мир, 1998.

15. Колмогоров А.Н. – Элементы теории функций и функционального анализа: учебное пособие / А. Н. Колмогоров, С. В. Фомин – М.: Наука; 2008. – 547 с.

16. Краснов М.Л. – Интегральные уравнения: учебное пособие / А.И. Киселев, Г.И. Макаренко – М.: Наука, 1968.

17. Краснов М.Л. Интегральные уравнения (задачи и примеры с подробными решениями): Учебное пособие / М.Л. Краснов, А.И. Киселев, Г.И. Макаренко. – М.: Едиториал УРСС, 2005. – 192 с.

18. Крылов В.И. – Вычислительные методы: учебное пособие / В.И. Крылов, В.В. Бобков, П.И. Монастырный – М.: Наука, 1977.

19. Крылов В.И. – Начала теории вычислительных методов: в 5 т. Интегральные уравнения, некорректные задачи и улучшеение сходимости: учебное пособие / В.И. Крылов, В.В. Бобков, П.И. Монастырный – Минск: Наука и техника, 1984. – 263 с.

20. Кутыркин В.А. – Методы решения интегрального уравнения Фредгольма 2-го рода с аналитически заданным непрерывным и симметричным ядром: электронное учебное издание / В.А. Кутыркин, Ю.В. Юрин – М.: МГТУ имени Н.Э. Баумана, 2013. – 32 с.

21. Липачёв Е.К. – Технология программирования. Базовые конструкции C/C++: учебное пособие / Е.К. Липачёв – Казань: Казанский (Приволжский) федеральный университет, 2012. – 142 с.

22. Люстерник Л.А. – Математический анализ, Вычисление элементарных функций: учебник / Л.А. Люстерник, О.А. Червоненкис, А.Р. Янпольский – М.: Наука, 1961.

23. Маклецов С.В. – Учебно-методическое пособие по курсу «Компьютерный практикум» часть 2 / С.В. Маклецов – Казань. Казанский (Приволжский) федеральный университет, 2013. – 104 с.

24. Мастяева И.Н. – Семенихина О.Н. Численные методы: учебное пособие / Московский международный институт эконометрики, информатики, финансов и права / И.Н. Мастяева, О.Н. Семенихина – М., 2004. – 103 с.

25. Михлин С.Г. – Приближенные методы решения дифференциальных и интегральных уравнений: учебник / С.Г. Михлин, Х.Л. Смолицкий – М.: Наука, 1965.

26. Осиновская, И.В. – Численные методы решения алгебраических уравнений и их систем [Электронный ресурс]: электрон, учеб. пособие / И. В. Осиновская, А. Г. Шляпугин, Я. А. Ерисов; Минобрнауки России, Самар, гос. аэрокосм, ун-т им. С. П. Королева (нац. исслед. ун-т). – Электрон, текстовые и граф. дан. (1,41 Мбайт). – Самара, 2012. – 69 с.

27. Палий, И.А. – Теория вероятностей: Учебное пособие / И.А. Палий. – М.: ИНФРА-М, 2012. – 236 с

28. Петросян, Л.А. – Теория игр: Учебник / Л.А. Петросян, Н.А. Зенкевич, Е.В. Шевкопляс. – СПб.: БХВ-Петербург, 2012. – 432 с.

29. Тихонов А.Н. – Об устойчивости обратных задач: учебное пособие / А.Н. Тихонов – ДАН СССР, 1943, 39 №5. – с. 198.

30. Треногин В. А. – Функциональный анализ: учебное пособие / В. А. Треногин – М.: Наука; 2009. – 495 с.

31. Чудесенко, В.Ф. – Сборник заданий по специальным курсам высшей математики. Типовые расчеты: учеб. пособие / В.Ф. Чудесенко – СПб.: Издательство «Лань», 2005. – 128 с.

32. Шмидский Я.К. – Программирование на языке С++. Диалектика: учебное пособие / Я.К. Шмидский – 2004 – 363с.

Литература на иностранном языке

33. F. Santoyo, E. Chávez, E.S. Téllez, «A Compressed Index for Hamming Distances», Similarity Search and Applications, Oct. 2014, pp. 113-126.

34. Hadamard J. Sur les problemes aux derivees parielies et leur signification physique. Bull. Univ. Prenceton, 1902, 13, p. 49-52.

35. Koprinkova-Hristova, P. Artificial Neural Networks / Petia Koprinkova-Hristova, Valeri Mladenov, Nikola K. Kasabov. – Springer International Publishing, 2015. – 488 p. 15. Li, S.Z. Hamming Distance / S.Z. Li, A. Jain. – Encyclopedia of Biometrics, 2009. – 668 p.

36. Rigatos, G. G. Advanced Models of Neural Networks / Gerasimos G. Rigatos. – Springer Berlin Heidelberg, 2015. – 396 p.

ПРИЛОЖЕНИЕ А

Основной блок программы

```
#define _USE_MATH_DEFINES

#include "stdafx.h"
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#include <time.h>
#include "math.h"
#include "FuncSqr.h"
#include "SLAUg.h"


int main(int argc, char **argv)
{
	//Выделение памяти под массивы результатов решений СЛАУ в точках пересечений
	double **YIvec = new double*[12];		//12 строк в массиве
	for (int count = 0; count < 12; count++)
		YIvec[count] = new double[6];	//6 столбцов
	int n, N, NN;
	double a, b, h, x, epsilon, Yxmax = 0, Yxmin = 0, M1, M2;
	float DELTA;
	char ch;// , buffer[100];
	//Формирование вектора узлов пересечений сетки
	n = 2;
	a = 0;				// -3.14159265358979323846;
	b = 3.14159265358979323846;
	h = (b - a) / n;
	x = a - h / 2;
	for (int i = 0; i < n; i++)
	{
		x += h;
		YIvec[i][0] = x;
	}
	//создаём новую страницу обработки данных
	printf("\f");
	//формиуются векторы результатов при n = 2, 6, 10, 14, 18
	for (int i = 0; i < 4; i++)
	{
		NN = n*(i * 2 + 1);
		CreateXi(YIvec, NN, a, b, Yxmax, Yxmin);
	}
	N = (NN / n - 1) / 2;
	//анализ результатов, вычисление погрешности
	do
	{
		N++;
		NN = n*(N * 2 + 1);
		CreateXi(YIvec, NN, a, b, Yxmax, Yxmin);
		epsilon = ProveYi(YIvec);
```

```
      } while ((epsilon > 0.001) && (N < 10 * n));
      printf("\n epsilon (n=%d) = %E", NN, epsilon);
      ////////////////////////////////////////////////////////
      //Вводим значение погрешности, с которой необходимо выполнить
вычисления
      ////////////////////////////////////////////////////////
      printf("\n Enter DELTA = x.xxxxxxxxx");
      printf("\n DELTA = ");
      scanf("%f",&DELTA);
      printf("\n DELTA = %E", DELTA);
      //вычисляем M1 и M2
      M1 = fabs(YIvec[5][0] - YIvec[4][0]);
      M2 = fabs(YIvec[1][1] - YIvec[0][1]);
      //вычисляем глобальную погрешность
      Yxmin = YIvec[8][1];
      if (YIvec[9][1] < Yxmin)   Yxmin = YIvec[9][1];
      epsilon = 2 * (sqrt(pow(YIvec[2][1], 2) + pow(YIvec[3][1], 2)));
      epsilon *= ((M1 / M2) * (((NN - 16) / n - 1) / 2));
      printf("\n EPSILON = %f", epsilon);
      //вычисляем n+
      NN = N;
      if (N < 10 * n) //вычисляем до ближайшего целого вверх значение n+
      {
            do
            {
                  NN++;
                  epsilon /= Yxmin;
            } while (epsilon > DELTA);
            N = n*(NN * 2 - 1);
      }
      else
      {
            epsilon *= (NN * NN);
            N = int(sqrt(epsilon / (DELTA)));
            N += n * (NN * 2 - 1);
      }
      printf("\n n+ = %d", N);
      do
      {
            ch = char(getch());
      } while (ch != 13);
      ////////////////////////////////////////////////////////
      //Нахождение окончательного решения уравнения
      ////////////////////////////////////////////////////////
      n = N;
      CreateXi(YIvec, n, a, b, Yxmax, Yxmin);
      do
      {
            ch = char(getch());
      } while (ch != 13);
      ////////////////////////////////////////////////////////
      for (int count = 0; count < 12; count++)
            delete[] YIvec[count];     //освобождение памяти
      ////////////////////////////////////////////////////////
    return 0;
}
```

ПРИЛОЖЕНИЕ Б

Подпрограмма решения СЛАУ

```
#include "stdafx.h"
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>

//Вычисление корней
void SlauGaus(double **SLAU, int count)
{
	//вспомогательные переменные
	double x;
	int jj;
	//создание треугольной матрицы
	for (int i = 0; i < count; i++)
	{
		for (int ii = i + 1; ii < count; ii++)
		{
			if (!(SLAU[ii][i] == 0)) //если равен нулю, то действие
пропускается
			{
				x = SLAU[ii][i] / SLAU[i][i]; // вычисляется общий
множитель
				for (int l = i; l < count + 1; l++)
				{
					SLAU[ii][l] -= SLAU[i][l] * x;
				}
			}
		}
	}
	//Вычисление результатов
	for (int i = 0; i < count; i++)
	{
		jj = count - i - 1;
		for (int j = count; j - 1 > jj; j--)
		{
			SLAU[jj][count] -= SLAU[jj][j - 1] * SLAU[j - 1][count];
		}
		SLAU[jj][count] /= SLAU[jj][jj];
	}
}
```

ПРИЛОЖЕНИЕ В

Подпрограмма решений интегральных уравнений

```
#define _USE_MATH_DEFINES

#include "stdafx.h"
#include <stdio.h>
#include "stdlib.h"
#include <conio.h>
#include <string.h>
#include "math.h"
#include "SLAUg.h"


//Вычисление значения ядра в узле сетки
double FuncKx(double x, double s)
{
	double tmp1;
	tmp1 = sin(s) + cos(x);
	return (tmp1);
}

//Вычисление значения остатка в узле сетки
double FuncFx(double x)
{
	return (exp(x) - (1 + exp(3.14159265358979323846)) / 2 + (1 -
exp(3.14159265358979323846))*cos(x));
}

//Вычисление собственных значений уравнения Фредгольма
void CreateXi(double **YIvec, int n, double a, double b, double Ymax,
double Ymin)
{
	//Open FILE
	FILE *f = fopen("Result.txt", "w");
	//Выделение памяти под массивы СЛАУ
	double **KIvec = new double*[n + 1];		//n строк в массиве (+1
под вектор результатов) (i)
	for (int count = 0; count < n; count++)
		KIvec[count] = new double[n + 1];	//n+1 столбцов (+1 под
вектор остатков) (j)
	double *Xvec = new double[n];			//вектор узлов сетки
	char bufprt[20];				//строка вывода
резльтатов
	//вспомогательные переменные
	double h = (b - a) / n;
	double x;
	int count;
	//создаём новую страницу обработки данных
	fprintf(f,"\f");
	//char ch;
	//Формирование вектора узлов сетки
	x = a - h / 2;
	for (int i = 0; i < n; i++)
	{
		x += h;
```

```
		Xvec[i] = x;
	}
	//Фомирование СЛАУ
	for (int i = 0; i < n; i++)
	{
		x = Xvec[i];
		for (int j = 0; j < n; j++)
		{//Вписываются значения ядра
			if (i == j) KIvec[i][j] = 1 - h * FuncKx(x, Xvec[j]);
			else KIvec[i][j] = (-1) * h * FuncKx(x, Xvec[j]);
		}
		KIvec[i][n] = FuncFx(x);
	}
	//Решение СЛАУ
	SlauGaus(KIvec, n);
	//выводим решение на дисплей
	printf("\n n = %d", n);
	fprintf(f, "\n n = %d", n);
	Ymax = KIvec[0][n];
	Ymin = Ymax;
	for (int i = 0; i < n; i++)
	{
		_gcvt(KIvec[i][n], 16, bufprt);
		printf("\n x[%d] = %lf; y[x] = %s", i+1, Xvec[i], bufprt);
		fprintf(f, "\n x[%d] = %lf; y[x] = %s", i + 1, Xvec[i],
bufprt);
		if (KIvec[i][n] > Ymax) Ymax = KIvec[i][n];
		if (KIvec[i][n] < Ymin) Ymin = KIvec[i][n];
	}
	//выводим значения в точках пересечений сеток
	if (n > 19) // если более 5той итерации - перенос значений
	{
		for (int i = 1; i < 5; i++)
		{
			YIvec[0][i] = YIvec[0][i + 1];
			YIvec[1][i] = YIvec[1][i + 1];
		}
		count = 0;
		for (int i = 0; i < n; i++)
		{
			if (((3 + 2 * (count + 1))*n) == 2 * (2 * n + 2 * (i +
1) - 1)) {
				YIvec[count][5] = KIvec[i][n];
				count++;
			}
		}
	}
	else
	{
		count = 0;
		for (int i = 0; i < n; i++)
		{
			if (((3 + 2 * (count + 1))*n) == 2 * (2 * n + 2 * (i +
1) - 1)) {
				YIvec[count][(n - 2) / 4 + 1] = KIvec[i][n];
				count++;
			}
		}
```

```
	}
	YIvec[4][0] = Ymax;
	YIvec[5][0] = Ymin;
	for (int cnt = 0; cnt < n; cnt++)
		delete [] KIvec[cnt]; //освобождение памяти
	fclose(f);
}

//анализ результатов, вычисление погрешности определения
//собственных значений уравнения Фредгольма
double ProveYi(double **YIvec)
{
	double epsilon;
	//находим значения ошибок
	for (int i = 0; i < 2; i++)
	{
		for (int j = 1; j < 5; j++)
		{
			YIvec[i + 2][j] = fabs(YIvec[i][j] - YIvec[i][j + 1]);
	//DELTA
		}
	}
	//находим значения второго приращения
	for (int i = 2; i < 4; i++)
	{
		for (int j = 1; j < 4; j++)
		{
			YIvec[i + 2][j] = fabs(YIvec[i][j] - YIvec[i][j + 1]);
	//DELTA_2
			YIvec[i + 4][j] = (YIvec[i + 2][j]) / 4;
	//f"(x)
		}
	}
	//находим отношение величин ускорений схождения
	for (int i = 6; i < 8; i++)
	{
		for (int j = 1; j < 3; j++)
		{
			YIvec[i + 2][j] = (YIvec[i][j]) / (YIvec[i][j + 1]);
	//TETTA
		}
	}
	//вычисляем проверочное значение КСИ
	for (int i = 8; i < 10; i++)
	{
		int j = 1;
		{
			YIvec[i + 2][j] = fabs((YIvec[i][j]) - (YIvec[i][j +
1])); //КСИ
			YIvec[i + 2][j] /= (YIvec[i][j + 1]);
		}
	}
	//находим максимум ошибки
	if (YIvec[10][1]>YIvec[11][1])  epsilon = YIvec[10][1];
	else                                    epsilon = YIvec[11][1];
	return epsilon;
}
```

ПРИЛОЖЕНИЕ Г

Таблица вычисленных собственных значений интегрального уравнения (алгоритм (2.6))

Эксперимент 1

$\Delta = 1 \cdot 10^{-3}$, алгоритм: (2.6)

$$f''(\xi_{\Pi}) \approx \frac{r^{\Pi}(h)}{h^2} \cdot \frac{24}{b-a}$$

i	x	$y_i(x)$	$y_{i0}(x)$	ε
1	-3,114509958	16,01836923	16,01836923	-7,81597E-14
2	-3,060344568	15,93022323	15,93022323	-6,21725E-14
3	-3,006179177	15,75496465	15,75496465	0
4	-2,952013787	15,49464824	15,49464824	-3,01981E-14
5	-2,897848396	15,15232598	15,15232598	-3,55271E-14
6	-2,843683005	14,73201128	14,73201128	-2,30926E-14
7	-2,789517615	14,23863194	14,23863194	-4,79616E-14
8	-2,735352224	13,6779724	13,6779724	-4,61853E-14
9	-2,681186834	13,05660586	13,05660586	0
10	-2,627021443	12,38181728	12,38181728	-8,34888E-14
11	-2,572856053	11,66151794	11,66151794	-6,21725E-14
12	-2,518690662	10,9041527	10,9041527	-6,57252E-14
13	-2,464525271	10,11860096	10,11860096	-1,95399E-14
14	-2,410359881	9,314072609	9,314072609	0
15	-2,35619449	8,5	8,5	0
16	-2,3020291	7,685927391	7,685927391	7,99361E-15
17	-2,247863709	6,881399038	6,881399038	0
18	-2,193698318	6,095847303	6,095847303	0
19	-2,139532928	5,338482059	5,338482059	0
20	-2,085367537	4,618182722	4,618182722	7,99361E-15
21	-2,031202147	3,943394144	3,943394144	0
22	-1,977036756	3,322027604	3,322027604	1,37668E-14
23	-1,922871366	2,761368056	2,761368056	0
24	-1,868705975	2,26798872	2,26798872	-4,88498E-15
25	-1,814540584	1,847674018	1,847674018	-3,9968E-15

i	x	$y_i(x)$	$y_{i0}(x)$	ε
26	-1,760375194	1,505351757	1,505351757	0
27	-1,706209803	1,24503535	1,24503535	-1,9984E-15
28	-1,652044413	1,069776771	1,069776771	-7,10543E-15
29	-1,597879022	0,981630766	0,981630766	1,11022E-15
30	-1,543713632	0,981630766	0,981630766	0
31	-1,489548241	1,069776771	1,069776771	-9,32587E-15
32	-1,43538285	1,24503535	1,24503535	-1,28786E-14
33	-1,38121746	1,505351757	1,505351757	-1,06581E-14
34	-1,327052069	1,847674018	1,847674018	-4,44089E-15
35	-1,272886679	2,26798872	2,26798872	-3,55271E-15
36	-1,218721288	2,761368056	2,761368056	-1,77636E-14
37	-1,164555897	3,322027604	3,322027604	0
38	-1,110390507	3,943394144	3,943394144	-3,9968E-15
39	-1,056225116	4,618182722	4,618182722	-1,15463E-14
40	-1,002059726	5,338482059	5,338482059	-1,5099E-14
41	-0,947894335	6,095847303	6,095847303	-1,5099E-14
42	-0,893728945	6,881399038	6,881399038	-1,33227E-14
43	-0,839563554	7,685927391	7,685927391	0
44	-0,785398163	8,5	8,5	0
45	-0,731232773	9,314072609	9,314072609	0
46	-0,677067382	10,11860096	10,11860096	-3,90799E-14
47	-0,622901992	10,9041527	10,9041527	-9,05942E-14
48	-0,568736601	11,66151794	11,66151794	-9,41469E-14
49	-0,514571211	12,38181728	12,38181728	-2,30926E-14
50	-0,46040582	13,05660586	13,05660586	-4,61853E-14
51	-0,406240429	13,6779724	13,6779724	-9,9476E-14
52	-0,352075039	14,23863194	14,23863194	0
53	-0,297909648	14,73201128	14,73201128	-9,05942E-14
54	-0,243744258	15,15232598	15,15232598	0
55	-0,189578867	15,49464824	15,49464824	0
56	-0,135413476	15,75496465	15,75496465	-8,88178E-14
57	-0,081248086	15,93022323	15,93022323	-5,32907E-14
58	-0,027082695	16,01836923	16,01836923	-7,81597E-14
59	0,027082695	16,01836923	16,01836923	-8,17124E-14

i	x	$y_i(x)$	$y_{i0}(x)$	ε
60	0,081248086	15,93022323	15,93022323	-6,92779E-14
61	0,135413476	15,75496465	15,75496465	-1,59872E-14
62	0,189578867	15,49464824	15,49464824	-4,61853E-14
63	0,243744258	15,15232598	15,15232598	-5,32907E-14
64	0,297909648	14,73201128	14,73201128	-4,44089E-14
65	0,352075039	14,23863194	14,23863194	-6,92779E-14
66	0,406240429	13,6779724	13,6779724	-6,75016E-14
67	0,46040582	13,05660586	13,05660586	-2,30926E-14
68	0,514571211	12,38181728	12,38181728	0
69	0,568736601	11,66151794	11,66151794	-8,17124E-14
70	0,622901992	10,9041527	10,9041527	1,77636E-14
71	0,677067382	10,11860096	10,11860096	-3,19744E-14
72	0,731232773	9,314072609	9,314072609	0
73	0,785398163	8,5	8,5	1,42109E-14
74	0,839563554	7,685927391	7,685927391	0
75	0,893728945	6,881399038	6,881399038	-7,10543E-15
76	0,947894335	6,095847303	6,095847303	0
77	1,002059726	5,338482059	5,338482059	7,10543E-15
78	1,056225116	4,618182722	4,618182722	0
79	1,110390507	3,943394144	3,943394144	0
80	1,164555897	3,322027604	3,322027604	-8,43769E-15
81	1,218721288	2,761368056	2,761368056	0
82	1,272886679	2,26798872	2,26798872	-7,54952E-15
83	1,327052069	1,847674018	1,847674018	0
84	1,38121746	1,505351757	1,505351757	3,77476E-15
85	1,43538285	1,24503535	1,24503535	-9,32587E-15
86	1,489548241	1,069776771	1,069776771	-5,32907E-15
87	1,543713632	0,981630766	0,981630766	3,10862E-15
88	1,597879022	0,981630766	0,981630766	-2,44249E-15
89	1,652044413	1,069776771	1,069776771	-1,19904E-14
90	1,706209803	1,24503535	1,24503535	-6,43929E-15
91	1,760375194	1,505351757	1,505351757	-5,10703E-15
92	1,814540584	1,847674018	1,847674018	-9,76996E-15
93	1,868705975	2,26798872	2,26798872	-1,06581E-14

i	x	$y_i(x)$	$y_{i0}(x)$	ε
94	1,922871366	2,761368056	2,761368056	-3,9968E-15
95	1,977036756	3,322027604	3,322027604	-6,21725E-15
96	2,031202147	3,943394144	3,943394144	-1,02141E-14
97	2,085367537	4,618182722	4,618182722	0
98	2,139532928	5,338482059	5,338482059	-1,68754E-14
99	2,193698318	6,095847303	6,095847303	0
100	2,247863709	6,881399038	6,881399038	-8,88178E-15
101	2,3020291	7,685927391	7,685927391	-2,30926E-14
102	2,35619449	8,5	8,5	0
103	2,410359881	9,314072609	9,314072609	-1,59872E-14
104	2,464525271	10,11860096	10,11860096	-2,13163E-14
105	2,518690662	10,9041527	10,9041527	-7,10543E-14
106	2,572856053	11,66151794	11,66151794	-7,28306E-14
107	2,627021443	12,38181728	12,38181728	0
108	2,681186834	13,05660586	13,05660586	-2,13163E-14
109	2,735352224	13,6779724	13,6779724	-7,4607E-14
110	2,789517615	14,23863194	14,23863194	-8,52651E-14
111	2,843683005	14,73201128	14,73201128	-6,57252E-14
112	2,897848396	15,15232598	15,15232598	-8,88178E-14
113	2,952013787	15,49464824	15,49464824	-9,59233E-14
114	3,006179177	15,75496465	15,75496465	-7,4607E-14
115	3,060344568	15,93022323	15,93022323	-4,44089E-14
116	3,114509958	16,01836923	16,01836923	-7,4607E-14

ПРИЛОЖЕНИЕ Д

Таблица вычисленных собственных значений интегрального уравнения (алгоритм (3.16))

Эксперимент 1

$\Delta = 1 \cdot 10^{-13}$, алгоритм: (3.16)

$$\psi_i(\theta) = \left| \frac{\theta_{i+1}(\theta(x)) - \theta_i(\theta(x))}{\theta_{i+1}(\theta(x))} \right| < 0{,}001$$

i	x	$y_i(x)$	$y_{i0}(x)$	ε
1	-3,020762167	15,81062074	15,81062074	-1,27898E-13
2	-2,779101194	14,13584563	14,13584563	-9,23706E-14
3	-2,53744022	11,16996621	11,16996621	-7,28306E-14
4	-2,295779247	7,592429702	7,592429702	1,77636E-15
5	-2,054118274	4,222806613	4,222806613	1,06581E-14
6	-1,8124573	1,833036983	1,833036983	1,64313E-14
7	-1,570796327	0,970588235	0,970588235	2,32037E-14
8	-1,329135353	1,833036983	1,833036983	1,79856E-14
9	-1,08747438	4,222806613	4,222806613	2,66454E-15
10	-0,845813407	7,592429702	7,592429702	-7,10543E-15
11	-0,604152433	11,16996621	11,16996621	-5,68434E-14
12	-0,36249146	14,13584563	14,13584563	-8,34888E-14
13	-0,120830487	15,81062074	15,81062074	-2,4869E-14
14	0,120830487	15,81062074	15,81062074	-1,3145E-13
15	0,36249146	14,13584563	14,13584563	-9,9476E-14
16	0,604152433	11,16996621	11,16996621	-7,99361E-14
17	0,845813407	7,592429702	7,592429702	-1,77636E-15
18	1,08747438	4,222806613	4,222806613	9,76996E-15
19	1,329135353	1,833036983	1,833036983	1,55431E-14
20	1,570796327	0,970588235	0,970588235	2,32037E-14
21	1,8124573	1,833036983	1,833036983	1,88738E-14
22	2,054118274	4,222806613	4,222806613	4,44089E-15
23	2,295779247	7,592429702	7,592429702	-1,77636E-15
24	2,53744022	11,16996621	11,16996621	-4,79616E-14
25	2,779101194	14,13584563	14,13584563	-7,4607E-14
26	3,020762167	15,81062074	15,81062074	-1,20792E-13

ПРИЛОЖЕНИЕ Е

Графическое сравнение алгоритмов (2.6) и (3.16)