МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

В.В. ЛАНДОВСКИЙ

# ЧИСЛЕННЫЕ МЕТОДЫ

Утверждено
Редакционно-издательским советом университета
в качестве учебного пособия

НОВОСИБИРСК
2022

Рецензенты: д.т.н., проф. *В.Д. Фроловский*,
к.т.н., доц. *Клочков Г. А.*

Работа подготовлена на кафедре автоматизированных систем управления для студентов II курса АВТФ направлений 09.03.01 - «Информатика и вычислительная техника», 09.03.03 - «Прикладная информатика».

**Ландовский В.В.** Численные методы: учеб. пособие. / В.В. Ландовский – Новосибирск: Изд-во НГТУ, 2022. – 64 с.

В настоящем пособии рассмотрены классические численные методы решения алгебраических и дифференциальных уравнений, задач дифференцирования, интегрирования и аппроксимации функций. Каждая тема сопровождается методическими указаниями для лабораторных работ и вариантами заданий.

**ОГЛАВЛЕНИЕ**

ВВЕДЕНИЕ ..... 5
1. РЕШЕНИЕ НЕЛИНЕЙНЫХ УРАВНЕНИЙ ..... 7
1.1. Метод половинного деления ..... 9
1.2. Метод простых итераций ..... 11
1.3. Метод Ньютона ..... 12
1.4. Метод секущих ..... 14
1.5. Метод хорд ..... 16
1.6 Обусловленность задачи уточнения корня и устойчивость методов ..... 17
1.7 Задание к лабораторной работе ..... 18
2. РЕШЕНИЕ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ ..... 20
2.1 Метод Гаусса ..... 21
2.2 Метод прогонки ..... 23
2.3 Метод простых итераций ..... 23
2.4 Метод Зейделя ..... 26
2.5 Задание к лабораторной работе ..... 27
3. АППРОКСИМАЦИЯ ФУНКЦИЙ ..... 30
3.1 Интерполяция многочленом ..... 31
3.2 Интерполяционный многочлен в форме Лагранжа ..... 32
3.3 Интерполяционный многочлен в форме Ньютона ..... 32
3.4 Сглаживание методом наименьших квадратов ..... 34
3.5 Задание к лабораторной работе ..... 36
3.6 Интерполяция сплайнами ..... 38
3.7 Кубический сплайн ..... 39
4 ЧИСЛЕННОЕ ДИФФЕРЕНЦИРОВАНИЕ ..... 42
4.1 Погрешности вычисления производных ..... 44
4.2 Задание к лабораторной работе ..... 46
5 ЧИСЛЕННОЕ ИНТЕГРИРОВАНИЕ ..... 47
5.1 Метод прямоугольников ..... 47
5.2 Метод трапеций ..... 50

5.3 Метод Симпсона ..... 51
5.4 Методы более высоких порядков ..... 53
5.5 Выбор шага интегрирования ..... 54
5.6 Задание к лабораторной работе ..... 55
6 РЕШЕНИЕ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ ..... 56
6.1 Метод Эйлера ..... 57
6.2 Модификации метода Эйлера ..... 58
6.3 Метод Рунге-Кутты четвертого порядка ..... 60
6.4 Метод Рунге–Кутты–Мерсона (с контролем точности) ..... 60
6.4 Семейство явных многошаговых методов Адамса ..... 61
6.5 Задание к лабораторной работе ..... 62
ЛИТЕРАТУРА ..... 64

# ВВЕДЕНИЕ

Численные методы – это методы, которые сводят решение задачи к последовательности вычислений – элементарных арифметических и логических действий над числами. К типовым задачам, которые могут быть решены численно относятся: решение систем линейных и нелинейных алгебраических уравнений; дифференцирование и интегрирование функций; задачи аппроксимации; решение систем дифференциальных уравнений. В отличие от аналитических методов численные без труда могут быть реализованы в виде компьютерных программ, так как по своей природе уже представляют собой строго формализованные алгоритмы. Результат работы таких алгоритмов всегда числовой, даже если задача состоит в поиске функции, численным решением будет набор её значений. Входной информацией могут служить как числовые данные, так и выраженные с помощью элементарных функций зависимости. Математические библиотеки большинства языков программирования, позволяют без труда получить значения элементарных функций, но получают их в результате работы численных методов.

Решение, полученное численным методом на компьютере, не может считаться точным, хотя бы в силу того, что не все числа могут быть представлены точно в ограниченном объеме памяти регистров ЭВМ. Результаты выполнения арифметических операций округляются, возникающую погрешность принято называть ***вычислительной***. Вычислительная погрешность каждой отдельной операции на современных компьютерах ничтожно мала, но в некоторых ситуациях она может накапливаться в процессе поиска решения и существенно искажает результат. Многие методы построены так, чтобы найти приближенное решение с заранее ***заданной точностью***, её называют ***погрешностью метода***. Теоретически такую точность можно задать сколь угодно малой, но на практике она ограничена вычислительной погрешностью.

Вычислительную или машинную точность не следует путать с минимальным положительным числом, которое может быть представлено на ЭВМ. В силу особенностей форматов хранения чисел с плавающей точкой плотность чисел, которые могут быть представлены точно неравномерна на

числовой оси. В окрестности нуля она максимальна, по мере удаления от нуля падает. Для 32 разрядной двоичной ЭВМ нижняя граница диапазона возможных значений $10^{-38}$, но вычислительная погрешность примерно $10^{-7}$.

Говорят, что вычислительная задача поставлена ***корректно***, если выполнены следующие три условия. Решение существует при любых допустимых исходных данных. Это решение единственно. Решение устойчиво по входным данным. ***Устойчивость по входным*** данным означает что, решение непрерывно зависит от входных данных. В этом случае погрешность результата будет уменьшаться при уменьшении погрешности входных данных, и теоретически решение можно получить с любой точностью.

Понятие ***обусловленности*** тесно связано с устойчивостью. Говорят, что задача хорошо обусловлена, если малому изменению исходных данных соответствует малое изменение решения. Незначительные погрешности в исходных данных плохо обусловленной задачи приводят к большим погрешностям в решении. Классическим примером такой задачи служит поиск корней многочлена

$$P(x) = (x-1)(x-2)(x-3)\cdot\ldots\cdot(x-20) = x^{20} - 210x^{19} + \ldots$$

Очевидно, что корнями являются $x_1 = 1, x_2 = 2, \ldots, x_{20} = 20$. Если изменить коэффициент –210 на $10^{-7}$, то половина корней станет мнимыми. Количественную меру степени обусловленности задачи называют ***числом обусловленности***, его определяют как отношение погрешностей решения к вызвавшим их погрешностям входных данных. Если говорят об абсолютной погрешности. Различают абсолютное и относительное число обусловленности характеризующие соответственно изменение абсолютной или относительной погрешности.

Свойство численного метода (например, методов решения задачи Коши) заключающееся во влиянии погрешности, возникающей на некотором шаге, на последующие шаги также называют устойчивостью. Если погрешность накапливается, то говорят, что ***метод неустойчив***.

# 1. РЕШЕНИЕ НЕЛИНЕЙНЫХ УРАВНЕНИЙ

Пусть дана некоторая функция $f(x)$ и требуется найти все или некоторые значения $x^*$, для которых $f(x^*)=0$. Относительно $f(x)$ часто предполагается, что она непрерывна и непрерывно дифференцируема в окрестности корня.

Выделяют два этапа: ***локализация*** (или отделение) корня и ***уточнение*** корня. Локализация корня заключается в определении отрезка, содержащего один и только один корень. На наличие корня на отрезке указывает различие знаков функции на его концах, но это не гарантирует, что он единственный. Свойство будет выполняться для любого нечётного количества корней. Универсального, эффективного во всех случаях подхода не существует. В любом случае на этом этапе потребуются знания, выходящие за рамки формальной постановки задачи, только вычислительных процедур здесь недостаточно. Прежде всего нужно определиться с диапазоном возможных значений корня, вся числовая ось от плюс до минус бесконечности пусть останется для анализа. В численных методах бесконечность конечна, хотя бы потому, что конечны числовые типы данных. Работа с числами бесконечной разрядности может быть реализована, но это совсем другая тема, в которой возникает переменное время выполнения операций со всеми неприятными последствиями.

Простейший, относительно универсальный способ отделения корней – выбрать длину отрезка не слишком малой (чтобы их количество не оказалось слишком велико) и не слишком большой (чтобы избежать размещения нескольких корней на одном отрезке) и проверить все отрезки в пределах интересующего диапазона. На этапе уточнения корня его значение вычисляют с **заданной точностью**. Приближенное значение корня уточняют с помощью различных итерационных методов.

Суть ***итерационных методов*** состоит в построении последовательных приближений к решению задачи. Вначале выбирают одно или несколько ***начальных приближений***, а затем последовательно, используя найденные ранее приближения и однотипную процедуру расчета, строят новые приближения. В

результате такого итерационного процесса можно построить бесконечную последовательность приближений к решению.

Говорят, что последовательность приближений $x_1, x_2, \ldots x_n$ ***сходится*** к точному решению $x = x^*$, если при неограниченном возрастании числа итераций предел этой последовательности существует и равен $x^*$, т.е. $\lim\limits_{n\to\infty} x_n = x^*$

Важной характеристикой итерационных методов является ***скорость сходимости*** метода. Говорят, что метод сходится со скоростью геометрической прогрессии со знаменателем $q < 1$, если для любого n справедливо

$$|x_n - x^*| \le \varepsilon_0 q^n \qquad (1.1)$$

где $\varepsilon_0$ граница погрешности начального приближения $x_0$.

Скорость сходимости можно выразить через соотношение расстояний между последовательными приближениями и корнем: $|x_n - x^*| = C|x_{n-1} - x^*|^p$, где $x_{n-1}$ и $x_n$ – последовательные приближения, $C$ – константа из интервала (0;1), не зависящая от $n$, а $p$ называют порядком сходимости. Зная скорость сходимости, то есть зная $C$ и $p$ можно сформулировать условие окончания итерационного процесса. Выполнение такого условия на некотором шаге номер $n$ будет означать, что $x_n$ удален от корня не более чем на величину требуемой точности вычислений $\varepsilon$.

Пусть $C$ =1/2, $p$=1, тогда каждое следующее приближение в два раза ближе к корню чем предыдущее. Если на этапе локализации определено, что корень расположен на отрезке $(a, b)$, то $|x_0 - x^*| \le (b-a)$, и тогда можно дать априорную оценку необходимого числа итераций $N$: $N=\log_2((b-a)/\varepsilon)$. На практике скорость сходимости зависит не только от метода, он и от поведения $f(x)$ в окрестности корня, а точное соотношение заменяется неравенством

$$|x_n - x^*| \le C|x_{n-1} - x^*|^p. \qquad (1.2)$$

Утверждение относительно числа итераций остаётся справедливым.

Учитывая, что скорость сходимости может меняться по мере приближения к корню, априорная оценка может оказаться неэффективной. Если неравенство определяет наименьшую, то есть гарантированную, скорость приближения к

корню на всём интервале локализации, то реальное $N$, скорее всего, окажется меньше полученного априорно. В таком случае лучше пользоваться условием окончания на основе неравенства вида

$$|x_n - x_{n-1}| \le g(\varepsilon), \tag{1.3}$$

где $g$ в общем случае определяется методом уточнения корня и поведением $f(x)$ в окрестности корня.

В идеализированном примере, приведенном выше, после каждого шага расстояние до корня сокращалось ровно вдвое, тогда остановить вычисления можно, если разница между последовательными приближениями не превосходит требуемой точности:

$$|x_n - x_{n-1}| \le \varepsilon. \tag{1.4}$$

Количество итераций в этом случае совпадает с рассчитанным априорно. Если бы метод позволял двигаться к корню быстрее, но не медленнее (чем сокращать расстояние в два раза) применение такой апостериорной оценки было бы выгодно. Для всех методов первого порядка сходимости (линейная сходимость) и $C<1/2$ условие (1.4) будет более сильным. При $C>1/2$ выполнения этого условия недостаточно, оно может остановить итерации раньше достижения точности.

Для методов со сверхлинейной сходимостью ($p>1$) определение $g(\varepsilon)$ в (1.3) связано с анализом $f(x)$ и требует дополнительных вычислительных затрат или предварительных расчетов. Чтобы этого избежать используют упрощённое условие (1.4), оно, как правило, более сильное, и его выполнения достаточно.

### 1.1. Метод половинного деления

Пусть корень $x^* \in [a_0, b_0]$, $f(x)$ непрерывна на $[a_0, b_0]$ и $f(a_0)\, f(b_0) < 0$. Отрезок делят пополам: $x_0 = (a_0 + b_0)/2$. Если $f(x_0) = 0$, то $x^* = x_0$. В противном случае находят знаки $f(x)$ на концах отрезков $[a_0, x_0]$ и $[x_0, b_0]$. Тот из них, на концах которого $f(x)$ имеет значения разных знаков, принимают за новый отрезок $[a_1, b_1]$, и вычисляют следующее приближение $x_1 = (a_1 + b_1)/2$, и так далее.

В отличие от большинства других методов, метод половинного деления обладает безусловной сходимостью, то есть сходится всегда (при корректной

локализации корня). Как окончательный результат можно рассматривать середину отрезка $[a_n, b_n]$, либо сам отрезок (одну из границ или любую другую точку). Если за приближенное значение корня берется величина $x_n$, то вычисления заканчиваются, когда выполнено неравенство $b_n - a_n < 2\varepsilon$ или неравенство $n > \log_2((b_0 - a_0)/\varepsilon) - 1$.

Метод сходится со скоростью геометрической прогрессии со знаменателем 1/2, однако соотношение (1.2) справедливо не для всех $n$. Дело в том, что сходимость метода половинного деления не всегда монотонна (последовательность $x_0, x_1, \ldots x_n$ не всегда является монотонной). Соотношение (1.1) будет справедливо для всех $n$, так как оперирует оценкой погрешности, в противоположность (1.2), где сравниваются фактические расстояния до корня. Если использовать в (1.2) оценки погрешности $\varepsilon_n = (b_n - a_n)/2$, то оно будет выполняться: $\varepsilon_n \le \varepsilon_{n-1}/2$. Пример немонотонной сходимости показан на рисунке 1.1, $\Delta_n$ обозначают фактическое расстояния от $x_n$ до $x^*$.

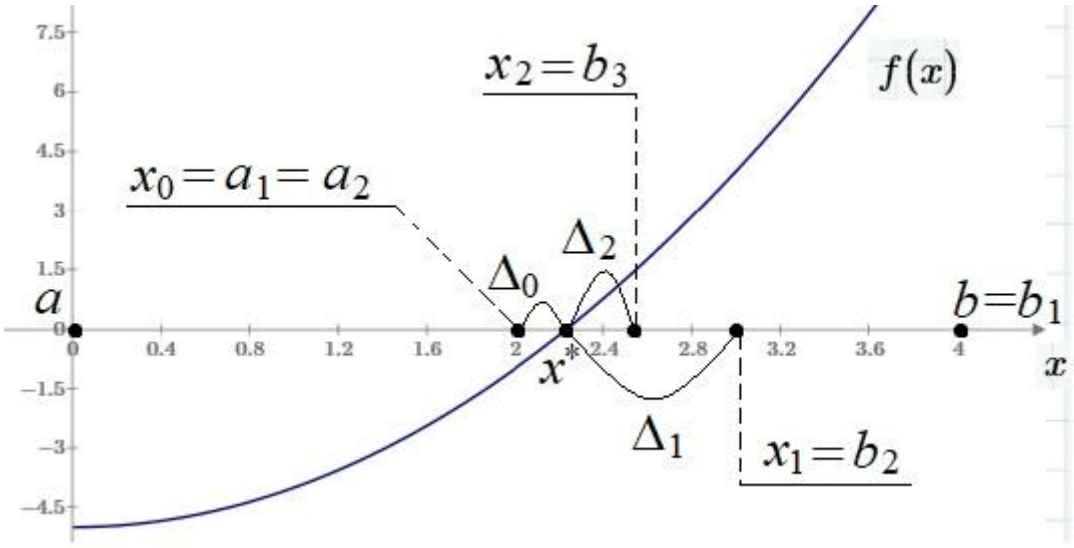


Рисунок 1.1

Шаг метода тем сильнее приближает $x_n$ к $x^*$, чем ближе $x^*$ к середине текущего отрезка. При этом уже на следующей итерации ситуация станет наихудшей, когда корень находится почти на границе отрезка. Если в некоторый момент корень окажется на расстоянии меньшем $\varepsilon$ от одной из границ, сходимость станет монотонной, соотношение (1.2) станет справедливым, причём фактически его можно будет считать равенством. Пример такой ситуации

изображен на рисунке 1.2. Использование (1.4) в качестве условия окончания допустимо, но не имеет преимуществ перед априорной оценкой количества шагов.

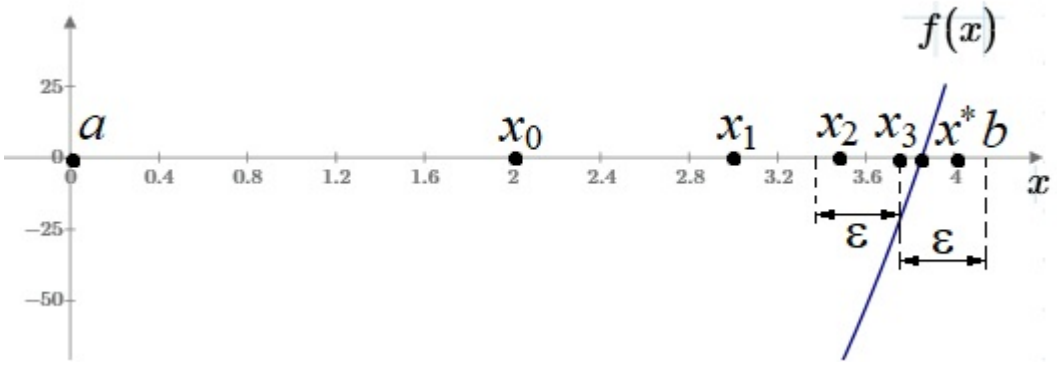


Рисунок 1.2

### 1.2. Метод простых итераций

Исходное уравнение $f(x) = 0$ заменяют эквивалентным: $x = \varphi(x)$. Если на $[a, b]$ $\varphi(x)$ непрерывна и $|\varphi'(x)|<1$, то итерационный процесс построенный по формуле $x_{n+1} = \varphi(x_n)$ сходится с любой точностью при любом $x_0$ из этого интервала.

Геометрическая интерпретация итерационного процесса представлена на рисунках 1.3, 1.4. Корнем исходного нелинейного уравнения является абсцисса точки пересечения кривой $y = \varphi(x)$ с прямой $y = x$. На рисунке 1.3 показаны примеры сходящегося итерационного процесса, на рисунке 1.4 – расходящегося.

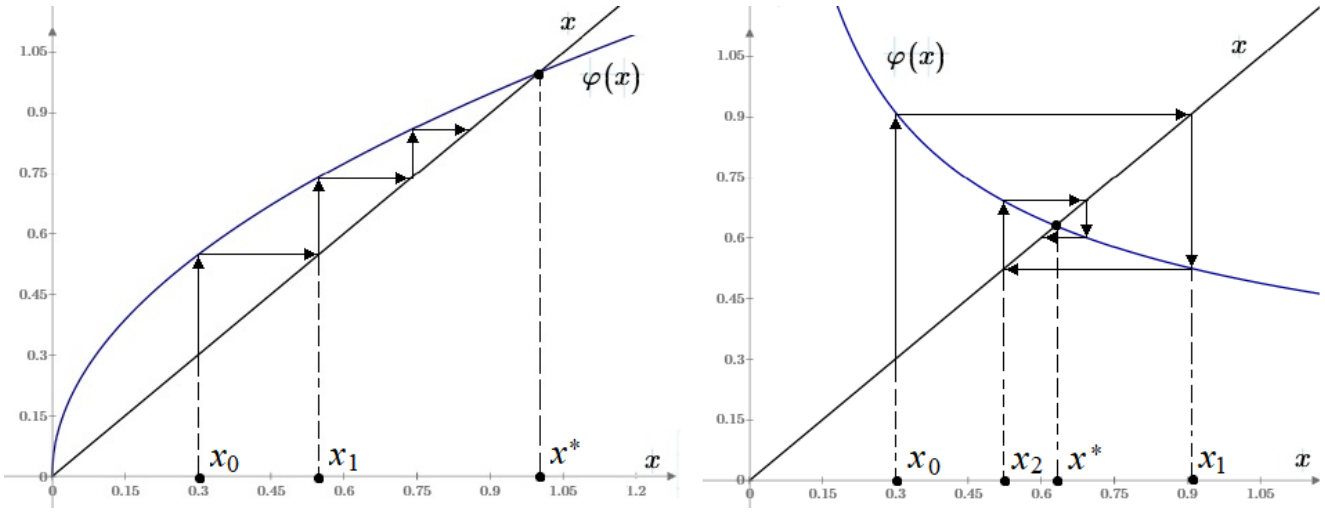


Рисунок 1.3

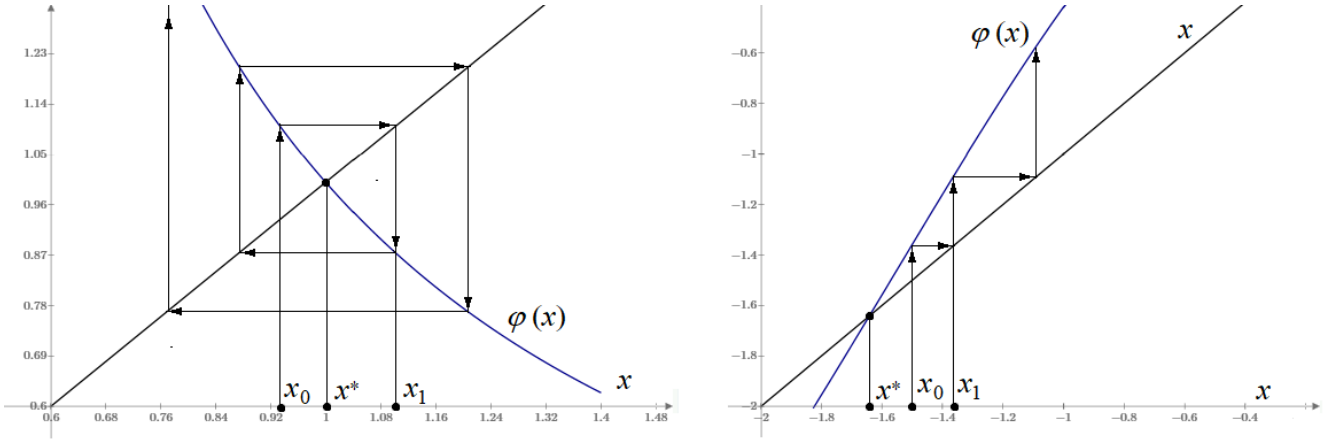


Рисунок 1.4

Скорость сходимости метода зависит от φ (*x*), пусть на [*a*, *b*] |φ '(*x*)|<*q*, тогда справедливо неравенство (1.2) с *C*=*q* и *p*=1. Если *q*<1/2 то в качестве условия окончания можно использовать (1.4), при *q*>1/2 необходимо пользоваться (1.3) приняв $g(\varepsilon) = \frac{q-1}{q}\varepsilon$.

Сказанное выше означает, что скорость сходимости и сходимость вообще зависит от преобразования исходного уравнения в эквивалентное. При наличии некоторой информации о поведении *f*(*x*) на [*a*, *b*] можно предложить универсальный способ преобразования, который гарантирует сходимость. Если на [*a*, *b*] *f*(*x*) и её первая производная непрерывны и 0<*m*<*f* '(*x*)<*M*, то за φ (*x*) можно принять *x*+λ*f*(*x*), где $\lambda = -\frac{1}{M}$. В этом случае φ '(*x*) на [*a*, *b*] будет меняться в интервале от 0 до $1 - \frac{m}{M} < 1$. Если производная *f* '(*x*) отрицательна и *m*<*f* '(*x*)<*M*<0, то для гарантии сходимости нужно взять $\lambda = \frac{1}{m}$. Обобщая, получим $\lambda = \pm \frac{1}{\max \left| f'(x) \right|_{[a,b]}}$, знак противоположен знаку *f* '(*x*) на [*a*, *b*].

### 1.3. Метод Ньютона

Итерационная формула метода Ньютона наиболее очевидным образом выводится из уравнения касательной к графику функции, в точке ($x_{n-1}$, $f(x_{n-1})$) оно имеет вид $y(x)= f(x_{n-1}) + f'(x_{n-1})(x - x_{n-1})$. Следующее приближение вычисляется как

абсцисса точки пересечения касательной с осью *Ox*, она определяется из уравнения полагая $y(x_n)=0$:

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}. \qquad (1.5)$$

На рисунке 1.5 показана иллюстрация итерационного процесса, построенного по такой формуле.

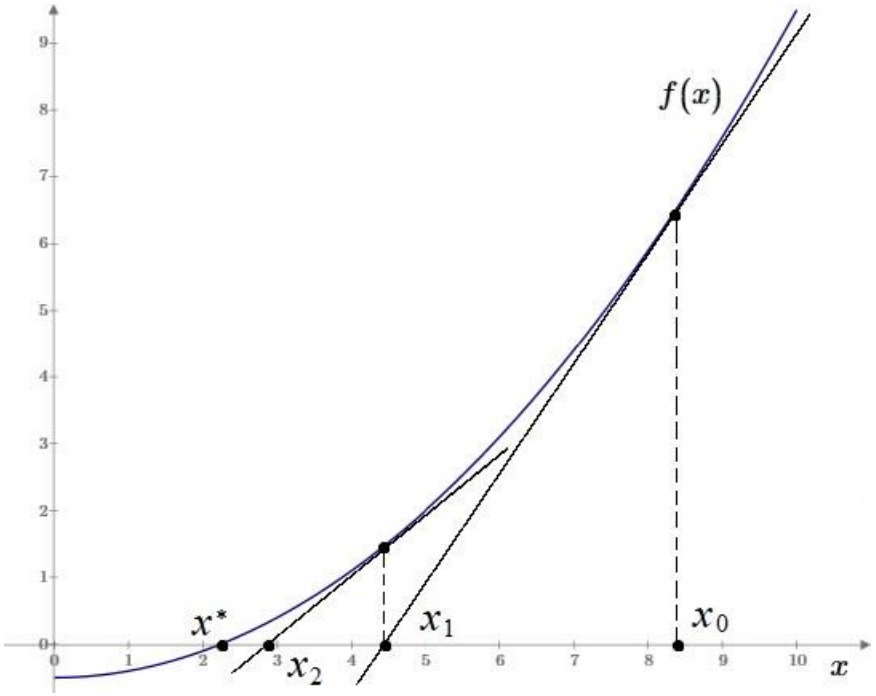


Рисунок 1.5

Аналогичную расчётную формулу можно было получить используя разложение $f(x)$ в окрестности точки $x_{n-1}$ в ряд Тейлора:

$$f(x) = f(x_{n-1}) + f'(x_{n-1})(x - x_{n-1}) + \frac{f''(x_{n-1})}{2}(x - x_{n-1})^2 + \ldots. \qquad (1.6)$$

Заменив $f(x)$ в уравнении первыми двумя членами разложения получим уравнение $f(x_{n-1}) + f'(x_{n-1})(x - x_{n-1}) = 0$. Новым приближением будем считать решение этого уравнения относительно *x*.

Очевидно, что итерационный процесс будет сходится только при соблюдении ряда условий. Во-первых, если корень локализован на отрезке [*a*, *b*], и соответственно $x_0 \in [a,b]$, $f(x)$ обязана быть непрерывна и непрерывно дифференцируема на этом отрезке. Во-вторых $f'(x^*)\neq 0$, следовательно $x^*$ - простой корень. Кроме того, сходимость будет зависеть от выбора начального

приближения. Пусть вторая производная также непрерывна, и обе производные не меняют знак на $[a, b]$, тогда за $x_0$ следует взять точку в которой знак функции совпадает со знаком второй производной: $f(x_0)\cdot f''(x_0) > 0$.

С помощью разложения $f(x)$ в ряд можно оценить скорость сходимости метода в малой окрестности корня, при соблюдении перечисленных выше условий. Можно показать, что для $n \geq 1$ неравенство (1.2) будет выполняться с параметрами $p=2$, $C = \frac{M_2}{2m_1}$, где $M_2 = \max\limits_{a \leq x \leq b} |f''(x)|$, $m_1 = \min\limits_{a \leq x \leq b} |f'(x)|$. Для этого вначале следует воспользоваться формулой конечных приращений Лагранжа $|f(x_n) - f(x^*)| = |f'(\xi)|\,|x_n - x^*|$, $\xi \in (x_n, x^*)$ и, поскольку $f(x^*)=0$, дать оценку $|x_n - x^*| \leq |f(x_n)|/m_1$. Затем подставить $x = x_n$ в (1.6) и, учитывая что $f(x_{n-1}) + f'(x_{n-1})(x_n - x_{n-1}) = 0$ согласно (1.5), получить $|f(x_n)| \leq M_2(x_n - x_{n-1})^2/2$. После чего объединить два неравенства в оценку погрешности на $n$-ном шаге:

$$|x_n - x^*| \leq \frac{M_2}{2m_1}|x_n - x_{n-1}|^2 \qquad (1.7)$$

Учитывая монотонный характер сходимости можно записать $|x_n - x_{n-1}| \leq |x_{n-1} - x^*|$, что позволяет заменить $|x_n - x_{n-1}|$ в (1.7) на $|x_{n-1} - x^*|$.

Согласно (1.7) в условии окончания (1.3) следует принять $g(\varepsilon) = \sqrt{\frac{2m_1\varepsilon}{M_2}}$. На практике обычно используют (1.4). Если $x_0 \in \left( x^* - \frac{m_1}{M_2}; x^* + \frac{m_1}{M_2} \right)$, то для всех n≥1 справедливо $|x_n - x^*| \leq |x_n - x_{n-1}|$, следовательно, использование (1.4) оправдано.

Вычисление производной на каждом шаге может затруднить практическое использование метода Ньютона. Ниже приводятся модификации, в которых используются приближенные значения производной.

### 1.4. Метод секущих

Значение производной в расчетной формуле метода Ньютона заменяют приближением: $f'(x_{n-1}) \approx \frac{f(x_{n-1}) - f(x_{n-2})}{x_{n-1} - x_{n-2}}$. Тогда она принимает следующий вид:

$x_n = x_{n-1} - \frac{(x_{n-1} - x_{n-2})\, f(x_{n-1})}{f(x_{n-1}) - f(x_{n-2})}$. В геометрическом смысле $x_n$ – это точка пересечения секущей – прямой, проходящей через $(x_{n-1}, f(x_{n-1}))$, $(x_{n-2}, f(x_{n-2}))$, с осью $Ox$, как показано на рисунке 1.6. Для обеспечения сходимости итерационного процесса к начальному приближению предъявляются такие же требования как в методе Ньютона. Скорость сходимости в окрестности корня будет ниже квадратичной, но выше линейной.

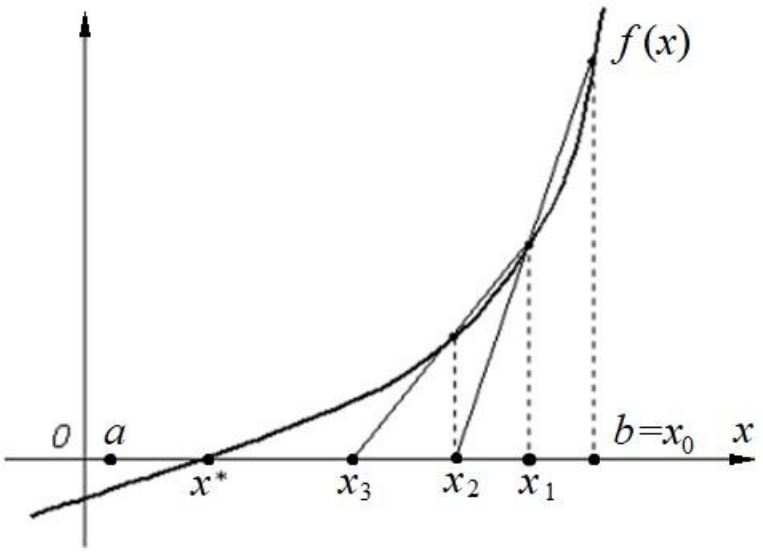


Рисунок 1.6

Для вычисления очередного приближения требуются два предыдущих. Если начальное приближение $x_0$ выбрано как одна из границ отрезка, содержащего корень, то для определения $x_1$ можно сделать шаг внутрь отрезка на величину требуемой точности вычислений, как показано на рисунке 1.7.

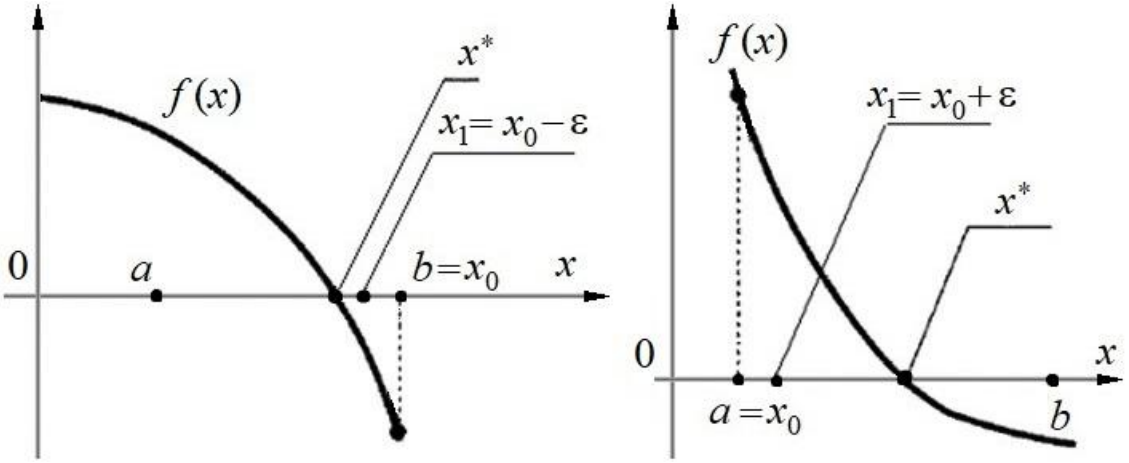


Рисунок 1.7

Уточнение корня можно прекратить, когда выполнится условие (1.4).

### 1.5. Метод хорд

Приближенное значение производной для расчётной формулы метода Ньютона можно определить следующим образом: $f'(x_n) \approx \frac{f(b) - f(x_n)}{b - x_n}$, при условии, что выполнены условия сходимости метода Ньютона, $f''(b) \cdot f(b) \geq 0$ и $x_0 = a$. Если $f''(a) \cdot f(a) \geq 0$, то $x_0 = b$, и $f'(x_n) \approx \frac{f(x_n) - f(a)}{x_n - a}$. Следующее приближение будет рассчитываться как

$$x_n = x_{n-1} - \frac{(b - x_{n-1}) f(x_{n-1})}{f(b) - f(x_{n-1})} \tag{1.8}$$

или $x_n = x_{n-1} - \frac{(x_{n-1} - a) f(x_{n-1})}{f(x_{n-1}) - f(a)}$ соответственно. Скорость сходимости зависит от поведения $f(x)$ на $[a, b]$ и от положения неподвижной точки – чем она ближе к корню, тем выше скорость. Последнее утверждение доказывается тем, что близость неподвижной точки к корню повышает точность аппроксимации производной.

На рисунке 1.8 видно, что расстояние между соседними приближениями не характеризует расстояние до корня, происходит это потому, что в данном конкретном случае скорость сходимости оказалась слишком низкой. Если $f'(x)$ сохраняет знак на $[a, b]$, то оценить скорость сходимости можно неравенством (1.2) с параметрами $p=1$, $C = \frac{M_1 - m_1}{m_1}$, где $M_1 = \max\limits_{a \leq x \leq b} |f'(x)|$, $m_1 = \min\limits_{a \leq x \leq b} |f'(x)|$. Для доказательства сперва следует получить оценку погрешности и по двум соседним приближениям, а затем свести её к (1.2) в силу монотонности последовательности приближений, также как при обсуждении метода Ньютона. Пусть $f'(x)>0$, $x_0 = a$, тогда по формуле конечных приращений Лагранжа $f(x_{n-1}) = -f'(\xi)(x^* - x_{n-1})$, $\xi \in (x_{n-1}, x^*)$; а $f(b) - f(x_{n-1}) = f'(\zeta)(b - x_{n-1})$, $\zeta \in (x_{n-1}, b)$. Если заменить $f(x_{n-1})$ в числителе (1.8) согласно первому соотношению и использовать $f'(\zeta)$ вместо $\frac{f(b) - f(x_{n-1})}{(b - x_{n-1})}$,

получится $x_n = x_{n-1} + \frac{f'(\xi)}{f'(\zeta)}(x^* - x_{n-1})$ или $x^* = \frac{f'(\zeta)}{f'(\xi)}(x_n - x_{n-1}) + x_{n-1}$ . Отнимем от обеих частей уравнения $x_n$ и получим: $x^* - x_n = \frac{f'(\zeta) - f'(\xi)}{f'(\xi)}(x_n - x_{n-1})$, поскольку $f'(\zeta) - f'(\xi) \le M_1 - m_1$, то можно записать $x^* - x_n \le \frac{M_1 - m_1}{m_1}(x_n - x_{n-1})$.

Грубая интерпретация полученной оценки такова: чем слабее меняется $f'(x)$ на $[a, b]$, тем выше скорость сходимости. Практически значимое следствие состоит в том, что если $M_1 \le 2\, m_1$, то для остановки итерационного процесса достаточно выполнения (1.4).

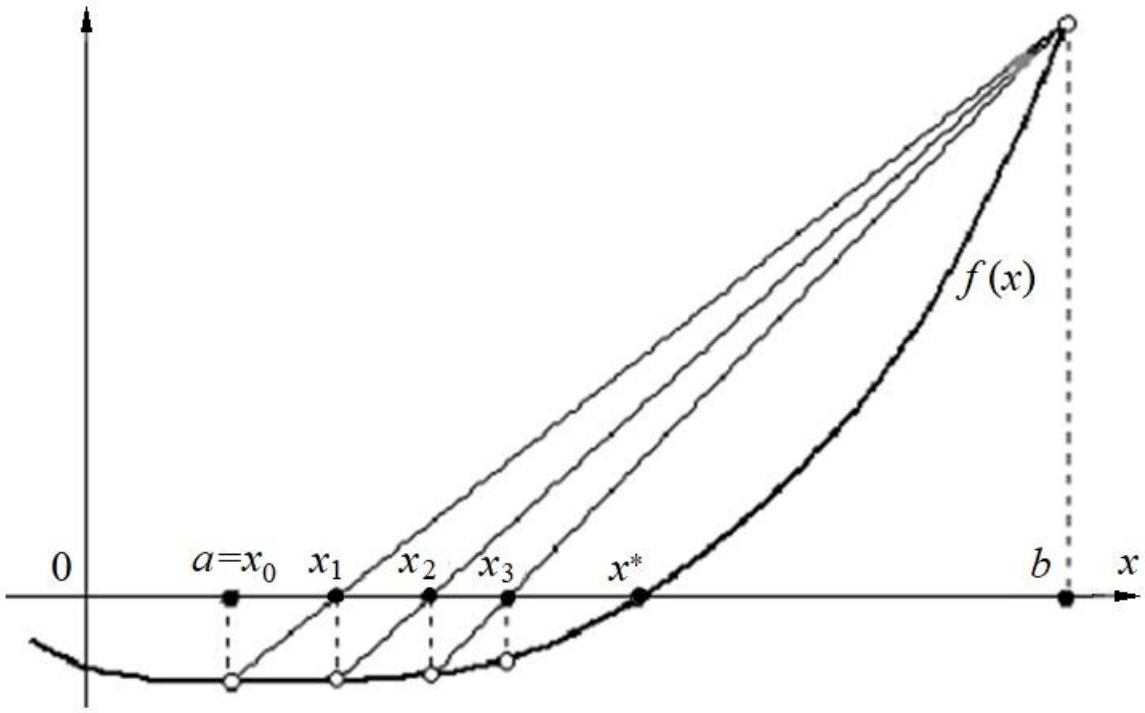


Рисунок 1.8

### 1.6 Обусловленность задачи уточнения корня и устойчивость методов

Итерационные методы устойчивы к ошибке, возникающей на некоторой итерации. Можно считать такую ошибку ухудшением начального приближения, если оно не выходит за пределы области сходимости, то внесенная ошибка будет затухать. Теоретически вычисления сходятся с любой точностью, но на практике это не так.

Значения $f(x)$ вычисляются на ЭВМ с погрешностью, связанной как с округлением, так и приближенными методами вычисления элементарных функций. Если $f$ непрерывна, то найдется некоторая малая окрестность корня, в которой значение $f(x)$ окажется меньше границы погрешности вычисления $f(x)$. Такую окрестность можно назвать ***интервалом неопределенности корня,*** поскольку невозможно точно определить, какое именно значение $x$ из интервала обращает $f(x)$ в ноль. Величина интервала неопределенности будет зависеть от скорости изменения $f(x)$ в окрестности корня. Абсолютным числом обусловленности можно считать $\frac{1}{\left|f'\left(x^*\right)\right|}$.

После того, как очередное приближение попало в интервал неопределенности корня, вычисления следует прекратить и считать, что получен максимально точный в данных условиях результат. Оценивать реальную величину интервала неопределенности на практике затруднительно, но попытаться определить попадание в него возможно. Для этого нужно сравнивать расстояния между соседними приближениями: выполнение неравенства $\left|x_n - x_{n-1}\right| \geq \left|x_{n-1} - x_{n-2}\right|$ может свидетельствовать о попадании $x_n$ в интервал неопределенности.

### 1.7 Задание к лабораторной работе

1. Для каждого корня заданного уравнения $f(x) = 0$ задать отрезок, содержащий корень, с помощью построения графика $f(x)$ в любом доступном математическом программном пакете. Границы отрезка выбрать как ближайшие к корню целые числа. При необходимости скорректировать значения границ так, чтобы выполнялись условия сходимости метода Ньютона, поведение производных проанализировать по графикам. В том же или подобном программном средстве вычислить значения корней максимально точно. Варианты $f(x)$ приведены в таблице 1.1.

2. Выполнить уточнение корней тремя методами: Ньютона, простых итераций и одного из методов заданного вариантом: половинного деления, хорд,

секущих. Для этого реализовать итерационный процесс соответствующих методов с помощью доступных систем программирования, для работы с дробными числами использовать 32 битное представление. При разработке алгоритмов стараться по возможности **минимизировать вычислительные затраты** (не вычислять $f(x)$ повторно, если можно использовать ранее вычисленное значение). Для метода Ньютона и его модификаций вычисления вести до выполнения условия (1.4), задать точность $10^{-3}$. В методах, использующих качестве начального приближения одно значение, попытаться (вопреки теоретическим представлениям) использовать как $x_0$ различные концы отрезков. Вывести промежуточные результаты (приближенные значения на каждой итерации), объяснить полученные результаты.

3. Для каждого метода и каждого корня экспериментально подобрать минимальное значение заданной погрешности, при которой вычисления сходятся к конечному значению (не возникает переполнения) и завершаются корректно (программа не входит в бесконечный цикл). Сравнить значения, вычисленные с этой точностью, с полученными в пунктах 1 и 2. Проделать тоже самое для положительного корня уравнения $f(x) = 100\,x^2\text{-}10000\,x\text{-}v$, где $v$ – номер варианта. Объяснить полученные результаты.

Отчёт по лабораторной работе должен содержать:

- графики $f(x)$ и её производных в масштабе достаточном для обоснования начальных приближений корней, результат вычисления корней в сторонних программных пакетах;
- тексты разработанных программ;
- результаты вычислений, полученные с помощью разработанных программ;
- объяснение полученных результатов.

Таблица 1.1

| № варианта | $f(x)$ |
|---|---|
| 1 | $x^2 - 20\sin(x) - 5$ |
| 2 | $2^x - 5x^2 + 10$ |
| 3 | $-x^4 + 15x^2 + 12x - 10$ |
| 4 | $2x^4 - 24x^2 - x + 8$ |
| 5 | $e^x - 4x^2 - 3x$ |
| 6 | $-x^2 + 7x - 4 \cdot \ln(x) - 7$ |
| 7 | $x^3 - x^2 - 7x + 5$ |
| 8 | $x^2 - 5 \cdot x \cdot \sin(x) + x + 1$ |
| 9 | $-x^4 + 4x^3 - 5 \cdot \ln(x) - 10$ |
| 10 | $x^2 - 5 \cdot x \cdot \cos(x) + x - 1$ |

# 2. РЕШЕНИЕ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ

Рассмотрим систему линейных алгебраических уравнений относительно $n$ неизвестных $x_1, x_2, \ldots, x_n$:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n. \end{cases} \qquad (2.1)$$

или в матричной форме

$$A\,x = b. \qquad (2.2)$$

Все способы решения систем линейных уравнений делятся на две группы: ***точные*** (или ***прямые***) методы и ***итерационные*** методы. Первые представляют собой алгоритмы с конечным числом шагов, в результате которых решение будет найдено, вторые позволяют получить приближенное решение системы с заданной точностью. Вследствие неизбежных округлений результаты точных методов, полученные с помощью ЭВМ, тоже являются приближенными.

Можно выделить группу специальных методов разработанных для ограниченного класса задач, они используют некоторые заранее известные

особенности системы (2.2). Ярким примером является метод прогонки для решения систем с трехдиагональной матрицей. Некоторые методы разработаны для более эффективного решения систем с симметричной или разряженной матрицей.

### 2.1 Метод Гаусса

Метод Гаусса, его еще называют методом Гауссовых исключений, состоит в том, что систему (2.1) вначале приводят последовательным исключением неизвестных к эквивалентной форме с треугольной матрицей (этот этап называют ***прямым ходом***):

$$\begin{cases} \alpha_{11}x_1 + \alpha_{12}x_2 + \ldots + \alpha_{1n}x_n = \beta_1 \\ \alpha_{22}x_2 + \ldots + \alpha_{2n}x_n = \beta_2 \\ \ldots\ldots\ldots\ldots\ldots \\ \alpha_{nn}x_n = \beta_n \end{cases},$$

после чего вычисляют $x_n = \dfrac{\beta_n}{\alpha_{nn}}$ и находят решение по рекуррентным формулам (этап называют ***обратным ходом***): $x_i = \dfrac{1}{\alpha_{ii}}\left(\beta_i - \sum_{j=i+1}^{n} \alpha_{ij}x_j\right), (i = n-1, n-2, \ldots, 1)$.

Алгоритм обратного хода очевиден, реализация прямого хода может допускает вариации. Простейший способ состоит в следующем. В первом столбце произвольно выбирается ненулевой элемент, строка, содержащая выбранный элемент, переставляется на место первой. Чтобы в остальных строках коэффициенты при $x_1$ равнялись нулю, к этим строкам прибавляют первую, умноженную на $-\dfrac{a_{m1}}{a_{11}}$, $m = 2, \ldots n$. В результате в первом столбце все коэффициенты под главной диагональю становятся нулевыми. Аналогичные действия последовательно выполняются для столбцов с номерами от 2 до $n$-1. При условии, что все $a_{ij}$ ненулевые алгоритм можно записать следующим образом:

$\alpha_{ij}^{(0)} = a_{ij}, \beta_{ij}^{(0)} = b_{ij}$, $i = 1, \ldots, n$; $j = 1, \ldots, n$;

$$\alpha_{ml}^{(k)} = \alpha_{ml}^{(k-1)} - \frac{\alpha_{mk}^{(k-1)}}{\alpha_{kk}^{(k-1)}}\alpha_{kl}^{(k-1)},\ \beta_{m}^{(k)} = \beta_{m}^{(k-1)} - \frac{\alpha_{mk}^{(k-1)}}{\alpha_{kk}^{(k-1)}}\beta_{k}^{(k-1)},$$

$k = 1,\ldots,n-1;\ m = k+1,\ldots,n;\ l = k,\ldots,n.$

Такую версию алгоритма называют ***схемой единственного деления***. Элементы $\alpha_{kk}^{(k-1)}$ называют ***главными*** (***ведущими***) элементами *k*-го шага.

Если какой-то ведущий элемент не равен нулю, но близок к нему, то в процессе вычислений могут возникать большие погрешности, которые будут накапливаться и могут существенно повлиять на результат. Для предотвращения этой ситуации разработаны модификации стандартного метода Гаусса схемы ***с выбором главного элемента***, которые обеспечивают, численную устойчивость процедуры решения.

При ***выборе главного элемента по столбцу*** в качестве ведущего элемента выбирают максимальный по модулю коэффициент $\alpha_{ik}^{(k-1)}$ при неизвестной $x_k$ в уравнениях с номерами $i = k,\ldots,n$. Затем уравнение, соответствующее выбранному коэффициенту с номером *i* меняют местами с *k*-м уравнением системы для того, чтобы главный элемент занял место коэффициента $\alpha_{kk}^{(k-1)}$.

Можно предложить похожий способ с выбором главного элемента по строкам. В этом случае изменится порядок исключения неизвестных – будут меняться положения столбцов. Описанные модификации называют ***схемами с частичным выбором***. Наилучшим в смысле обеспечения устойчивости будет ***схема полного выбора*** – поиск ведущего элемента по всей матрице. Для замены $\alpha_{kk}^{(k-1)}$ будет выбран максимальный элемент из $\alpha_{ij}^{(k-1)}$, $i = k,\ldots,n$, $j = k,\ldots,n$.

Вычислительная сложность метода Гаусса определяется прямым ходом и составляет $O(n^3)$ операций. Процедура выбора главного элемента по всей матрице имеет такую же асимптотическую оценку. Обратный ход и частичный выбор главного элемента требуют порядка $O(n^2)$ действий.

### 2.2 Метод прогонки

Адаптацию метода Гаусса для решения систем с трехдиагональной матрицей называют методом прогонки. Пусть матрица *A* системы уравнений (2.2) имеет вид:

$$A = \begin{bmatrix} d_1 & e_1 & 0 & 0 & \dots & 0 \\ c_2 & d_2 & e_2 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & c_{n-1} & d_{n-1} & e_{n-1} \\ 0 & \dots & 0 & 0 & c_n & d_n \end{bmatrix}$$

Прямой ход метода приведёт систему к следующему виду:

$$\begin{bmatrix} d_1 & e_1 & 0 & 0 & \dots & 0 \\ 0 & d_2 - c_2\alpha_2 & e_2 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & 0 & d_{n-1} - c_{n-1}\alpha_{n-1} & e_{n-1} \\ 0 & \dots & 0 & 0 & 0 & d_n - c_n\alpha_n \end{bmatrix} \begin{matrix} b_1 \\ b_2 - c_2\beta_2 \\ \dots \\ b_{n-1} - c_{n-1}\beta_{n-1} \\ b_n - c_n\beta_n \end{matrix}, \text{ где}$$

$$\alpha_2 = \frac{e_1}{d_1},\ \beta_2 = \frac{b_1}{d_1},\ \alpha_{i+1} = \frac{e_i}{d_i - c_i\alpha_i}, i = 2,\dots n-1,\ \beta_{i+1} = \frac{b_i - c_i\beta_i}{d_i - c_i\alpha_i}, i = 2,\dots n-1.$$

Обратный ход позволит найти решение по следующим соотношениям:

$$x_n = \frac{b_n - c_n\beta_n}{d_n - c_n\alpha_n},\ x_i = \beta_{i+1} - \alpha_{i+1}x_{i=1}, i = n-1,\dots,1.$$

Вычислительная сложность прямого и обратного хода, как и метода в целом, составляет O(*n*) операций.

### 2.3 Метод простых итераций

Для применения метода простых итераций систему (2.1) нужно преобразовать к виду

$$x = \alpha x + \beta, \tag{2.3}$$

где $\alpha = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \dots & \alpha_{1n} \\ \alpha_{21} & \alpha_{22} & \dots & \alpha_{2n} \\ \dots & \dots & \dots & \dots \\ \alpha_{n1} & \alpha_{n2} & \dots & \alpha_{nn} \end{bmatrix}$, $\beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \dots \\ \beta_n \end{bmatrix}$.

Предполагая, что диагональные коэффициенты $a_{ii}$, $i = 1,\ldots,n$ не равны нулю, из первого уравнения (2.1) выражают $x_1$, из второго – $x_2$ и так далее. Таким образом е

$\beta_i = \frac{b_i}{a_{ii}}$, $\alpha_{ij} = -\frac{a_{ij}}{a_{ii}}$, при $i \neq j$, $i = 1,\ldots,n$, $j = 1,\ldots,n$; $\alpha_{ij} = 0$ при $i=j$.

Очередное приближение $x^{(k+1)}$ вычисляется следующим образом:

$$x^{(k+1)} = \beta + \alpha x^{(k)}. \qquad (2.4)$$

В качестве начального приближения $x^{(0)}$ можно взять вектор $\beta$. Итерационные формулы в развернутом виде будут выглядеть так:

$$\begin{cases} x_i^{(0)} = \beta_i, i = 1,\ldots,n \\ x_i^{(k+1)} = \beta_i + \sum_{j=1}^{n} \alpha_{ij} x_j^{(k)}, i = 1,\ldots,n; k = 0,1,2,\ldots \end{cases}.$$

Эффективность итерационных методов зависит от удачного выбора начального приближения и быстроты сходимости процесса. Решением системы уравнений (2.1) является вектор, для анализа работы итерационных методов необходима некая количественная характеристика векторов. Для этой цели используются нормы, а в качестве меры различия двух векторов $x$ и $x^*$ вычисляют $\|x^* - x\|$.

Нормой квадратной матрицы $A$ называется функционал, обозначаемый $\|A\|$, удовлетворяющий условиям:

1. $\|A\| \geq 0$, причем $\|A\| = 0$ тогда и только тогда, когда $A = 0$;
2. $\|A\gamma\| = |\gamma| \cdot \|A\|$, для любого числа $\gamma$;
3. $\|A + B\| \leq \|A\| + \|B\|$;
4. $\|AB\| \leq \|A\|\|B\|$.

Как правило используются следующие виды норм и соответствующие им нормы векторов:

1. $\|A\|_1 = \max_j \sum_i |A_{ij}|$, $\|x\|_1 = \sum_i |x_i|$;
2. $\|A\|_\infty = \max_i \sum_j |A_{ij}|$, $\|x\|_\infty = \max_i |x_i|$;

3. $\|A\|_k = \sqrt{\sum_{ij} A_{ij}^2}$, $\|x\|_2 = \sqrt{\sum_i x_i^2}$.

Для сходимости итерационного процесса (2.4) достаточно чтобы норма матрицы $\alpha$ была меньше единицы:

$$\|\alpha\| < 1 . \tag{2.5}$$

При этом справедлива оценка погрешности:

$$\left\|x^{(k)} - x^*\right\| \le \|\alpha\|^k \left\|x^{(0)} - x^*\right\|, \tag{2.6}$$

где $x^*$ точное решение.

Как следствие процесс итерации сходится, если для исходной непреобразованной системы (2.1) выполнены неравенства:

$|a_{ii}| > \sum_{j=1}^{i-1} |a_{ij}| + \sum_{l=i+1}^{n} |a_{il}|$, $i = 1,\ldots,n$, то есть модули диагональных элементов преобладают над суммой остальных в строке, или

$|a_{jj}| > \sum_{i=1}^{j-1} |a_{ij}| + \sum_{l=j+1}^{n} |a_{lj}|$, $j = 1,\ldots,n$ - модули диагональных элементов преобладают над суммой остальных в столбце.

Для доказательства оценки погрешности (2.6) из итерационной формулы (2.4) нужно вычесть равенство $x^* = \alpha x^* + \beta$:

$$x^{(k+1)} - x^* = \alpha\left(x^{(k)} - x^*\right). \tag{2.7}$$

Переходя к нормам, с учётом свойства нормы произведения, получится неравенство $\left\|x^{(k+1)} - x^*\right\| \le \|\alpha\|\left\|x^{(k)} - x^*\right\|$. Поскольку это неравенство справедливо для всех $k$=0,1,…, то справедливо (2.6). Использовать эту оценку на практике невозможно, поскольку $\left\|x^{(0)} - x^*\right\|$ неизвестно.

Для получения апостериорной оценки погрешности равенство (2.7) можно записать в виде: $x^{(k)} - x^* = \alpha\left(x^{(k-1)} - x^{(k)} + x^{(k)} - x^*\right) = \alpha\left(x^{(k-1)} - x^{(k)}\right) + \alpha\left(x^{(k)} - x^*\right)$.

Переходя к нормам:

$\left\|x^{(k)} - x^*\right\|(1 - \|\alpha\|) \le \|\alpha\|\left\|x^{(k-1)} - x^{(k)}\right\|$ или

$$\left\| x^{(k)} - x^{*} \right\| \le \frac{\left\| \alpha \right\|}{1 - \left\| \alpha \right\|} \left\| x^{(k-1)} - x^{(k)} \right\|.$$

Если требуется найти решение с точностью ε, то вычисления следует вести до тех по, пока не выполнится условие:

$$\left\| x^{(k-1)} - x^{(k)} \right\| \le \frac{1 - \left\| \alpha \right\|}{\left\| \alpha \right\|} \varepsilon . \qquad (2.8)$$

Если $\left\| \alpha \right\| \le \frac{1}{2}$, то можно использовать упрощенное неравенство

$$\left\| x^{(k-1)} - x^{(k)} \right\| \le \varepsilon . \qquad (2.9)$$

Выбор той или иной нормы должен определяться требованиями, предъявляемыми к точности решения. Норма $\left\| x \right\|_1$ отвечает минимизации суммарного отклонения, $\left\| x \right\|_2$ - среднеквадратической ошибки, выбор $\left\| x \right\|_\infty$ обеспечит наименьшее максимальное отклонение в компонентах решения.

Условие (2.5) является достаточным условием сходимости метода простых итераций, необходимое и достаточное условие формулируется иначе. Метод сходится при любом начальном приближении тогда и только тогда, когда все собственные числа матрицы α по модулю меньше единицы.

### 2.4 Метод Зейделя

Метод Зейделя представляет собой модификацию метода итераций. Основная его идея заключается в том, что при вычислении $(k+1)$-го приближения неизвестной $x_i$ учитываются уже вычисленные ранее $(k+1)$-е приближения неизвестных $x_{1,} x_2, \ldots, x_{i-1}$.

Пусть получена эквивалентная система (2.3), и задано начальное приближение $x^{(0)} = \left( x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)} \right)^T$, тогда расчётные формулы метода Зейделя будут иметь вид:

$$
\begin{aligned}
x_1^{(k+1)} &= \beta_1 + \alpha_{12}x_2^{(k)} + \alpha_{13}x_3^{(k)} + \ldots + \alpha_{1n}x_n^{(k)} \\
x_2^{(k+1)} &= \beta_2 + \alpha_{21}x_1^{(k+1)} + \alpha_{23}x_3^{(k)} + \ldots + \alpha_{2n}x_n^{(k)} \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
x_n^{(k+1)} &= \beta_n + \alpha_{n1}x_1^{(k+1)} + \alpha_{n2}x_2^{(k+1)} + \ldots + \alpha_{nn}x_n^{(k)}
\end{aligned}
, \; k = 0,1,\ldots .
$$

Если представить матрицу $\alpha$ как сумму двух треугольных: $\alpha = \alpha_1 + \alpha_2$, где

$$
\alpha_1 = \begin{bmatrix} 0 & 0 & \ldots & 0 \\ \alpha_{21} & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ \alpha_{n1} & \alpha_{n2} & \ldots & 0 \end{bmatrix}, \; \alpha_2 = \begin{bmatrix} 0 & \alpha_{12} & \ldots & \alpha_{1n} \\ 0 & 0 & \ldots & \alpha_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & 0 \end{bmatrix},
$$

то можно доказать, что $\left\| x^{(k)} - x^* \right\| \le \dfrac{\left\|\alpha_2\right\|}{1 - \left\|\alpha\right\|} \left\| x^{(k)} - x^{(k-1)} \right\|$. Из этого следует критерий окончания итерационного процесса:

$$
\left\| x^{(k-1)} - x^{(k)} \right\| \le \frac{1 - \left\|\alpha\right\|}{\left\|\alpha_2\right\|} \varepsilon , \tag{2.10}
$$

где ε - необходимая точность решения.

Если $A$ – симметричная положительно определенная матрица, то метод Зейделя сходится при любом начальном приближении без каких-либо дополнительных условий.

### 2.5 Задание к лабораторной работе

1. Подготовить программную реализацию метода Гаусса без выбора главного элемента, использовать 32 битное представление вещественных чисел. Реализацию выполнить так, чтобы ей было возможно воспользоваться при разработке программ для последующих лабораторных работ. С помощью полученной программы решить соответствующую варианту систему СЛАУ 1 из таблицы 2.1. Добавить в программу выбор главного элемента заданным способом. Решить ту же систему с выбором главного элемента. Объяснить полученные результаты.

2. Подготовить программную реализацию приведения системы уравнений к виду (2.3) и решения системы заданным итерационным методом. Выполнение

условия сходимости перед началом итерационного процесса не контролировать, в качестве критерия остановки использовать (2.9). Попытаться решить соответствующую варианту СЛАУ 2 из таблицы 2.1. Вручную преобразовать систему уравнений так, чтобы гарантировать сходимость итерационного процесса. Добавить в программу опциональную проверку условия (2.5) и проверку окончания в соответствии с (2.8) или (2.10) в зависимости от метода. Решить преобразованную систему с точностью $10^{-3}$. Экспериментально подобрать минимальное значение заданной погрешности, при которой вычисления сходятся к конечному значению (не возникает переполнения) и завершаются корректно (программа не входит в бесконечный цикл). Объяснить все полученные результаты.

3. Подготовить программный модуль позволяющий решать систему с трехдиагональной матрицей произвольной размерности. Реализацию выполнить так, чтобы ей было удобно воспользоваться при разработке программ для последующих лабораторных работ. Провести тестирование на собственных примерах.

При разработке всех алгоритмов стараться по возможности **минимизировать вычислительные затраты.**

Отчёт по лабораторной работе должен содержать:

- исходные и преобразованные системы уравнений;
- тексты разработанных программ;
- результаты вычислений, полученные с помощью разработанных программ;
- объяснение полученных результатов.

Таблица 2.1

| № варианта | СЛАУ 1: $Ax=b$ | СЛАУ 2: $Ax=b$ |
|---|---|---|
| 1 | $A=\begin{pmatrix}5 & 2.833333 & 1\\ 3 & 1.7 & 7\\ 1 & 8 & 1\end{pmatrix}, b=\begin{pmatrix}11.666666\\ 13.4\\ 18\end{pmatrix}$ | $A=\begin{pmatrix}1 & 1 & 15\\ 15 & 0 & 1\\ 4 & 15 & 1\end{pmatrix}, b=\begin{pmatrix}17\\ 16\\ 20\end{pmatrix}$ |
| 2 | $A=\begin{pmatrix}2.83333 & 5 & 1\\ 1.7 & 3 & 7\\ 8 & 1 & 1\end{pmatrix}, b=\begin{pmatrix}11.66666\\ 13.4\\ 18\end{pmatrix}$ | $A=\begin{pmatrix}2 & 15 & 1 & 1\\ 3 & -1 & 2 & 20\\ 10 & 1 & 1 & 1\\ 3 & 2 & 14 & -2\end{pmatrix}, b=\begin{pmatrix}21\\ 27\\ 23\\ 20\end{pmatrix}$ |
| 3 | $A=\begin{pmatrix}5 & 3 & 1\\ 3 & 1.79999 & 7\\ 1 & 8 & 1\end{pmatrix}, b=\begin{pmatrix}12\\ 13.59998\\ 18\end{pmatrix}$ | $A=\begin{pmatrix}2 & 15 & -1\\ 12 & 2 & 3\\ 1 & 2 & 16\end{pmatrix}, b=\begin{pmatrix}29\\ 25\\ 53\end{pmatrix}$ |
| 4 | $A=\begin{pmatrix}3 & 5 & 1\\ 1.799999 & 3 & 7\\ 8 & 1 & 1\end{pmatrix}, b=\begin{pmatrix}12\\ 13.599998\\ 18\end{pmatrix}$ | $A=\begin{pmatrix}5 & 30 & 6\\ -3 & 4 & 20\\ 10 & 2 & 1\end{pmatrix}, b=\begin{pmatrix}53\\ 61\\ 15\end{pmatrix}$ |
| 5 | $A=\begin{pmatrix}10 & 20 & 30\\ 40 & 80.00001 & 60\\ 5 & -15 & 25\end{pmatrix}, b=\begin{pmatrix}60\\ 180.00001\\ 15\end{pmatrix}$ | $A=\begin{pmatrix}1 & 2 & 15 & 1\\ -1 & 3 & 2 & 20\\ 10 & 1 & 1 & 1\\ 2 & 14 & 3 & -2\end{pmatrix}, b=\begin{pmatrix}19\\ 24\\ 13\\ 17\end{pmatrix}$ |
| 6 | $A=\begin{pmatrix}2 & 1.99999 & 3 & 1\\ 1 & 1 & 1 & 1\\ 1 & -1 & -2 & 1\\ 1 & 2 & 1 & 1\end{pmatrix}, b=\begin{pmatrix}7.99997\\ 2\\ -1\\ 5\end{pmatrix}$ | $A=\begin{pmatrix}1 & 9 & 1\\ 2 & 2 & 11\\ 10 & 2 & 1\end{pmatrix}, b=\begin{pmatrix}12\\ 26\\ 14\end{pmatrix}$ |
| 7 | $A=\begin{pmatrix}10 & 2.0001 & 2 & 0\\ 5 & 1 & -2 & 4\\ 3 & 5 & 1 & -1\\ 0 & 6 & -2 & 2\end{pmatrix}, b=\begin{pmatrix}21.0001\\ 14\\ 10\\ 8\end{pmatrix}$ | $A=\begin{pmatrix}3 & 1 & 5\\ 0 & 7 & 3\\ 8 & 1 & 1\end{pmatrix}, b=\begin{pmatrix}12\\ 10\\ 18\end{pmatrix}$ |
| 8 | $A=\begin{pmatrix}2.00001 & 2 & 3 & 1\\ 1 & 1 & 1 & 1\\ -1 & 1 & -2 & 1\\ 2 & 1 & 1 & 1\end{pmatrix}, b=\begin{pmatrix}8.00003\\ 2\\ -1\\ 5\end{pmatrix}$ | $A=\begin{pmatrix}3 & 1 & 10\\ 14 & 2 & 3\\ 2 & 12 & -3\end{pmatrix}, b=\begin{pmatrix}18\\ 35\\ 25\end{pmatrix}$ |
| 9 | $A=\begin{pmatrix}15 & 20 & 30 & 40\\ 1 & 1.333333 & 1 & 1\\ 4 & 3 & 2 & 1\\ -1 & 1 & -1 & 1\end{pmatrix}, b=\begin{pmatrix}105\\ 4.333333\\ 10\\ 0\end{pmatrix}$ | $A=\begin{pmatrix}3 & 4 & 20\\ 5 & 30 & 6\\ 10 & 2 & 1\end{pmatrix}, b=\begin{pmatrix}27\\ 41\\ 13\end{pmatrix}$ |

| | | |
|---|---|---|
| 10 | $A = \begin{pmatrix} 20 & 15 & 30 & 40 \\ 1.33333 & 1 & 1 & 1 \\ 3 & 4 & 2 & 1 \\ 1 & -1 & -1 & 1 \end{pmatrix}, b = \begin{pmatrix} 105 \\ 4.33333 \\ 10 \\ 0 \end{pmatrix}$ | $A = \begin{pmatrix} 12 & 0 & 1 \\ 3 & 3 & 13 \\ 3 & 11 & 1 \end{pmatrix}, b = \begin{pmatrix} 26 \\ 38 \\ 30 \end{pmatrix}$ |

## 3. АППРОКСИМАЦИЯ ФУНКЦИЙ

Во многих задачах возникает проблема замены некоторой функции, заданной аналитически или таблично, другой, близкой, но более простой и удобной при вычислениях. Другая задача – восстановление функции на некотором отрезке по заданным на этом отрезке значениям функции в дискретном множестве точек.

В общем случае при постановке задачи аппроксимации необходимо прежде всего выбрать класс приближающих функций и определиться с критерием близости исходной и приближающей функции. В качестве критерия можно выбрать, например, точное совпадение приближаемой и приближающей функций в узловых точках (интерполяция); минимум суммы квадратов отклонения в узловых точках (метод наименьших квадратов) или другие. Указать правило, позволяющее получить значение функции в промежутках между узлами, ответить на вопросы, какие узлы использовать и как их расположить. На рисунке 3.1 приведен пример: звёздочками выделены узловые точки, сплошная линия – это интерполяция, пунктир – среднеквадратичное приближение.

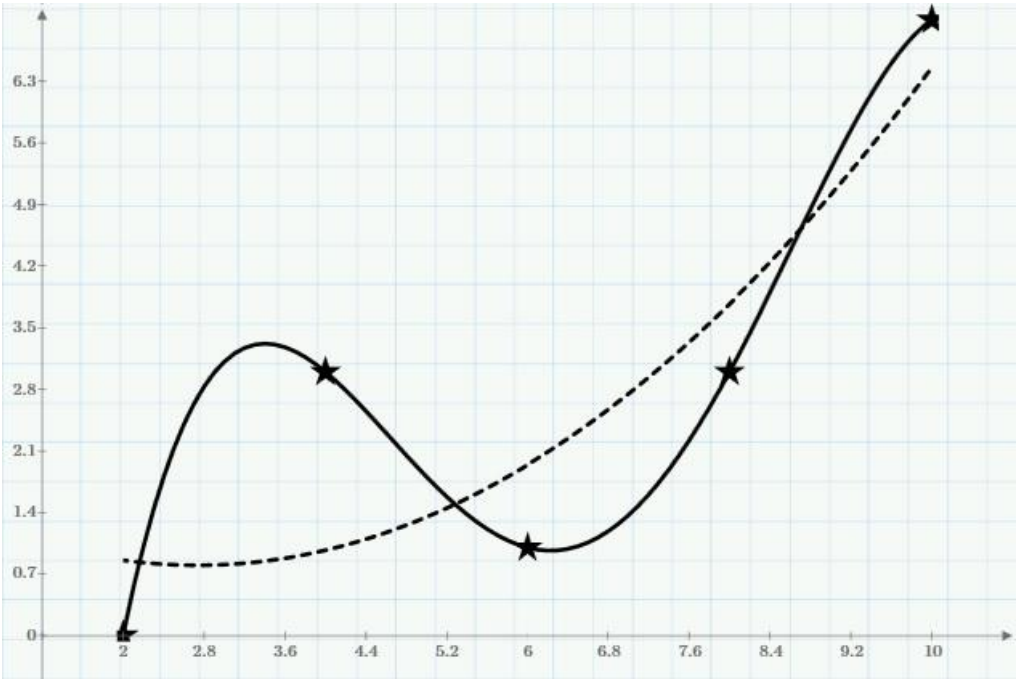

Рисунок 3.1

### 3.1 Интерполяция многочленом

Пусть на отрезке $[a, b]$ заданы $(n + 1)$ несовпадающих точек $a = x_0 < x_1 < \ldots < x_n = b$, известны значения функции $f(x)$ в этих точках: $f_i = f(x_i)$, $i = 0,1,\ldots,n$. Требуется по табличным значениям $(x_i, f_i)$ построить функцию $\varphi(x)$, такую, что значения $\varphi(x)$ легко вычисляются при любом $x \in [a, b]$ и при этом $\varphi(x_i) = f(x_i)$.

Приближающая функция может представлять собой линейную комбинацию некоторых базисных функций $\varphi_i(x)$: $\varphi(x) = \sum_{i=0}^{n} a_i \varphi_i(x)$, в качестве которых можно выбрать любую линейно независимую систему функций. При использовании степенных функций $1, x, x^2, \ldots, x^n$ функция $\varphi(x)$ представляет собой многочлен степени $n$:

$$P(x) = \sum_{i=0}^{n} a_i x^i .$$

Многочлены легко вычисляются, и теория интерполяции многочленами хорошо разработана. Для определения значений коэффициентов $a_i$ требуется решить систему уравнений $P(x_i) = f_i$; $i=0,\ldots,n$:

$$\begin{cases} a_0 + a_1 x_0 + \ldots + a_n x_0^n = f_0 \\ a_0 + a_1 x_1 + \ldots + a_n x_1^n = f_1 \\ \qquad\qquad \ldots \\ a_0 + a_1 x_n + \ldots + a_n x_n^n = f_n \end{cases} \qquad (3.1)$$

Определитель этой системы является ненулевым определителем Вандермонда:

$$\Delta = \begin{vmatrix} 1 & x_0 & \ldots & x_0^n \\ 1 & x_1 & \ldots & x_1^n \\ \ldots & \ldots & \ldots & \ldots \\ 1 & x_n & \ldots & x_n^n \end{vmatrix} = \prod_{j>i\geq 0}^{n} (x_j - x_i) \neq 0$$

Следовательно, решение системы существует и единственно, и для таблицы значений функции ($x_i$, $f_i$), $i$=0,…,$n$ существует единственный интерполяционный многочлен.

На практике для поиска коэффициентов интерполяционного многочлена систему уравнений (3.1) не применяют, так как обычно она плохо обусловлена. Другая причина состоит в том, что часто достаточно только одного или нескольких значений $P(x)$, поэтому явное вычисление коэффициентов $a_i$ избыточно.

### 3.2 Интерполяционный многочлен в форме Лагранжа

Интерполяционный многочлен, записанный в виде $L(x) = \sum_{i=0}^{n} f_i l_i(x)$, где $l_i(x)$ представляет собой многочлен степени $n$, удовлетворяющий условию $l_i(x_j) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$, называют интерполяционным многочленом в форме Лагранжа. Условие равенства нулю означает, что все $x_j$ за исключением $x_i$ являются корнями многочлена $l_i$ и по теореме о разложении многочлена по корням: $l_i(x) = C_i(x - x_0)\ldots(x - x_{i-1})(x - x_{i+1})\ldots(x - x_n)$. Из условия равенства $l_i$ единице при $i$=$j$ следует, что $C_i = \frac{1}{(x_i - x_0)\ldots(x_i - x_{i-1})(x_i - x_{i+1})\ldots(x_i - x_n)}$.

Таким образом, интерполяционный многочлен в форме Лагранжа будет выглядеть так: $L(x) = \sum_{i=0}^{n} f_i \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x - x_j}{x_i - x_j} = \sum_{i=0}^{n} f_i \frac{(x - x_0)\ldots(x - x_{i-1})(x - x_{i+1})\ldots(x - x_n)}{(x_i - x_0)\ldots(x_i - x_{i-1})(x_i - x_{i+1})\ldots(x_i - x_n)}$.

Вычислительная сложность этой формулы составляет O($n^2$) операций. Обычно ей пользуются напрямую – без запоминания промежуточных значений, то есть не преследуя целей ускорения повторных вычислений.

### 3.3 Интерполяционный многочлен в форме Ньютона

Интерполяционный многочлен в форме Ньютона естественным образом позволяет сохранить промежуточные данные для последующих вычислений. А

основное преимущество этой формы записи интерполяционного многочлена в сравнении с формой Лагранжа заключается в отсутствии необходимости перестраивать весь многочлен при добавлении новой точки в таблицу.

Для вывода интерполяционного многочлена в форме Ньютона понадобится представление о разделенных разностях.

Разделенные разности ***нулевого*** порядка совпадают со значениями функции. Разделенные разности ***первого*** порядка определяются соотношениями:

$$f(x_i; x_j) = \frac{f(x_i) - f(x_j)}{x_i - x_j}.$$

Разделенные разности ***второго*** порядка определяются через разделенные разности первого порядка:

$$f(x_i; x_j; x_k) = \frac{f(x_i; x_j) - f(x_j; x_k)}{x_i - x_k}.$$

Разделенная разность ***порядка n*** определяется соотношением:

$$f(x_i; x_j; x_k; \ldots; x_n) = \frac{f(x_i; x_j; x_k; \ldots; x_{n-1}) - f(x_j; x_k; \ldots; x_n)}{x_i - x_n}$$

Например, если функция задана таблицей значений в четырех точках:

$x_0$, $x_1$, $x_2$, $x_3$, то разделенные разности удобно вычислять в порядке представленном в таблице 3.1.

Таблица 3.1

<table>
<tr><td>$f(x_0)$</td><td rowspan="2">$f(x_0; x_1)$</td><td rowspan="2">$f(x_0; x_1; x_2)$</td><td rowspan="4">$f(x_0; x_1; x_2; x_3)$</td></tr>
<tr><td>$f(x_1)$</td></tr>
<tr><td>$f(x_2)$</td><td rowspan="2">$f(x_1; x_2)$</td><td rowspan="2">$f(x_1; x_2; x_3)$</td></tr>
<tr><td>$f(x_3)$</td><td>$f(x_2; x_3)$</td></tr>
</table>

Разделенные разности более высоких порядков равны нулю.

Вывод формулы интерполяционного многочлена в форме Ньютона.

Из определения разделенной разности первого порядка:

$$f(x) = f(x_0) + (x - x_0) f(x; x_0).$$

Из определения разделенной разности второго порядка:

$$f(x; x_0) = f(x_0; x_1) + (x - x_1) f(x; x_0; x_1).$$

подставляя выражение для $f(x; x_0)$ в предыдущее равенство получаем:

$$f(x) = f(x_0) + (x - x_0) f(x_0; x_1) + (x - x_0)(x - x_1) f(x; x_0; x_1).$$

Если функция задана таблицей из двух точек, на этом этапе следует остановиться. Последний член равен нулю, так как разделенная разность третьего порядка, в этом случае равна нулю.

Продолжая дальше, для таблицы из ($n$+1) точки, получим:

$$f(x) = f(x_0) + (x - x_0) f(x_0; x_1) + (x - x_0)(x - x_1) f(x_0; x_1; x_2) + \ldots +$$

$$+ (x - x_0)(x - x_1)\ldots(x - x_{n-1}) f(x_0; x_1; \ldots; x_n) + (x - x_0)(x - x_1)\ldots(x - x_n) f(x; x_0; \ldots; x_n).$$

Асимптотически вычислительная сложность в сравнении с формой Лагранжа не меняется и остаётся O($n^2$) операций. Но если оценить более точно, с учётом констант, то вычисление $f(x)$ с готовыми разделенными разностями будет выполняться в несколько раз быстрее. Разделенные разности можно вычислить заранее и сохранить для всех последующих вычислений. В случае добавления новой точки в исходную таблицу значений функции старые значения разделенных разностей останутся актуальными, потребуется добавить недостающие и можно продолжать вычисления $f(x)$.

### 3.4 Сглаживание методом наименьших квадратов

При проведении некоторых экспериментальных исследований наблюдаются частые и резкие изменения значений измеряемых величин. Чтобы избежать трудностей в изучении таких процессов и управлении ими, используют ***сглаживание***.

Применять сглаживание нужно осмотрительно. В некоторых случаях резко выделяющиеся точки могут характеризовать существенные качественные изменения, происходящие в исследуемом объекте. Сглаживая экспериментальные данные, можно легко утратить информацию о таких явлениях.

Сущность метода наименьших квадратов, для табличной функции состоит в том, чтобы отыскать такую аналитическую зависимость из некоторого класса функций, для которой сумма квадратов отклонений по всем точкам таблицы была бы минимальной.

Пусть связь между аргументами $x_i$ и значениями функции $f_i$, $i$=0,...,$n$, приближенно описывается выражением: $y = p(x, a_0, \ldots, a_k)$ с числовыми параметрами $a_0, \ldots, a_k$. Требуется определить такие значения этих параметров, при которых сумма квадратов отклонений $\sum_{i=0}^{n} \left( p(x_i, a_0, \ldots, a_k) - y_i \right)^2$ будет наименьшей.

Для многочлена $k$-й степени $P_k(x) = a_k x^k + \ldots + a_1 x + a_0$ сумма квадратов отклонений представляет собой неотрицательную функцию переменных $a_0, \ldots, a_k$:

$$F(a_0, \ldots, a_k) = \sum_{i=0}^{n} \left( a_k x_i^k + \ldots + a_1 x_i + a_0 - y_i \right)^2 .$$

Требующиеся наилучшие коэффициенты многочлена должны давать минимум функции $F$.

Необходимым условием экстремума дифференцируемой функции многих переменных является равенство нулю ее частных производных по всем переменным, следовательно, задача отыскания коэффициентов сводится к решению системы уравнений:

$$\begin{cases} \dfrac{\partial F}{\partial a_0} = 0, \\ \ldots \\ \dfrac{\partial F}{\partial a_k} = 0 \end{cases}$$

Дифференцируя $F$ по каждой переменной, получаем систему линейных уравнений:

$$\begin{cases} c_0 a_0 + c_1 a_1 + c_2 a_2 + \ldots + c_k a_k = d_0 \\ c_1 a_0 + c_2 a_1 + c_3 a_2 + \ldots + c_{k+1} a_k = d_1 \\ \ldots \\ c_k a_0 + c_{k+1} a_1 + c_{k+2} a_2 + \ldots + c_{2k} a_k = d_k \end{cases}, \qquad (3.2)$$

где

$$c_m = \sum_{i=0}^{n} x_i^m, m = 0,\ldots,2k \text{ , } d_j = \sum_{i=0}^{n} f_i x_i^j, j = 0,\ldots,k \text{ .}$$

Значения коэффициентов аппроксимирующего многочлена $a_i$ являются решением полученной системы. Если степень многочлена на единицу меньше количества точек ($k$=$n$), то будут получены коэффициенты интерполяционного многочлена.

### 3.5 Задание к лабораторной работе

1. Для функции заданной таблицей значений согласно варианту задания из таблицы 3.2 в любом доступном математическом программном пакете выполнить:
- расчет коэффициентов сглаживающих многочленов степеней 1, 2, 3 по методу наименьших квадратов;
- расчет коэффициентов интерполяционного многочлена с помощью решения системы уравнений (3.1);
- построение графиков полученных многочленов.

2. Разработать программную реализацию построения графиков
интерполяционного многочлена, полученного с помощью формулы Лагранжа, по произвольной таблице значений функции;
интерполяционного многочлена, полученного с помощью формулы Ньютона, по произвольной таблице значений функции;
сглаживающих многочленов 1, 2, 3 степени, полученных по методу наименьших квадратов, по произвольной таблице значений функции;
произвольного многочлена четвертой степени, **заданного набором коэффициентов** (*для выполнения следующего пункта задания*).

При разработке алгоритма стараться по возможности **минимизировать вычислительные затраты**.

Для решения системы (3.2) использовать реализацию метода Гаусса с выбором главного элемента **из соответствующей лабораторной работы**. Графики отображать как кусочно-линейную аппроксимацию с малым (~$10^{-2}$)

шагом. Стили отображения линий (штриховка, цвет, толщина и т.п.) должны **позволять видеть совпадающие графики**. Желательно отдельно **отобразить точки** исходной таблицы. Ограничений на среду разработки не накладывается, однако рекомендуется использовать современные высокоуровневые возможности, позволяющие не тратить время на программирование построения графиков (компонент Chart в C# Windows Forms, библиотека QCustomPlot в Qt/C++, компонент Tchart в Embarcadero C++ Builder и т.п.).

Способ ввода должен быть удобным и позволять корректировку отдельных элементов данных (поля ввода в окне, чтение файла и т.п.; только **не последовательный ввод в консоли**).

3. С помощью разработанной программы показать совпадение графиков многочленов, рассчитанных вручную в пункте 1 с реализованными программно.

Отчёт по лабораторной работе должен содержать:

- исходную таблицу значений функции;
- результаты «ручных» расчётов в сторонних программных пакетах (коэффициенты многочленов, графики);
- тексты разработанных программ;
- результаты, полученные с помощью разработанных программ (сравнения согласно пункту 3 задания);

Таблица 3.2

| № варианта | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
|---|---|---|---|---|---|
| | $f(x_1)$ | $f(x_2)$ | $f(x_3)$ | $f(x_4)$ | $f(x_5)$ |
| 1 | 2 | 4 | 5 | 6 | 7 |
| | 6 | 6 | 1 | -1 | 11 |
| 2 | -1 | 1 | 2 | 3 | 4 |
| | -5 | -3 | 18 | 6 | -2 |
| 3 | -2 | 0 | 1 | 3 | 5 |
| | 7 | 6 | 10 | 9 | 10 |
| 4 | -2 | 0 | 2 | 3 | 4 |

| | | | | | |
|---|---|---|---|---|---|
| | 18 | 12 | 7 | -1 | 0 |
| 5 | 5 | 7 | 9 | 11 | 12 |
| | 3 | -2 | -2 | 4 | 15 |
| 6 | 8 | 10 | 12 | 14 | 16 |
| | 4 | 9 | 5 | 1 | 16 |
| 7 | -1 | 0 | 1 | 2 | 3 |
| | -2 | -2 | -7 | 1 | 14 |
| 8 | -1 | 0 | 2 | 3 | 4 |
| | -1 | 2 | 10 | 10 | 15 |
| 9 | 7 | 9 | 10 | 12 | 14 |
| | 8 | 11 | 7 | 2 | -5 |
| 10 | 2 | 3 | 4 | 6 | 7 |
| | -1 | -6 | 7 | 8 | 2 |

### 3.6 Интерполяция сплайнами

Поведение интерполяционных многочленов высоких степеней между узловыми точками может оказаться неприятным сюрпризом. Дело в том, что с увеличением степени появляются выраженные колебания большой амплитуды. На рисунке 3.2 показан пример интерполяционного многочлена, построенного по таблице значений функции соответствующей таблице 3.3.

В большинстве случаев для большой таблицы значений функции более пригодна кусочно-полиномиальная аппроксимация, при которой различным интервалам соответствуют различные аппроксимирующие многочлены.

Таблица 3.3

| $x$ | 0 | 1 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $f(x)$ | 1 | 2 | 0 | 5 | 10 | 0 | 9 | 5 | 3 | 4 | 2 |


Рисунок 3.2

Сплайном степени $m$ называется функция $S_m(x)$, обладающая следующими свойствами:

- она непрерывна на отрезке $[x_0, x_n]$ вместе со своими производными до некоторого порядка $p$;

- на каждом частичном отрезке $[x_i, x_{i+1}]$ совпадает с некоторым многочленом степени $m$.

Разность между степенью сплайна и наивысшим порядком непрерывной на отрезке $[x_0, x_n]$ производной $(m - p)$ называют дефектом сплайна.

Можно сказать, что кусочно-линейная функция является сплайном первой степени с дефектом, равным единице.

### 3.7 Кубический сплайн

Наиболее распространенный вид сплайн интерполяции – это кубический сплайн с дефектом 1. Пусть аппроксимирующая $f(x)$ кривая обозначена $\varphi(x)$, а на каждом интервале $[x_{i-1}, x_i]$, $i = 1,...,n$ она совпадает с многочленом третьей степени $\varphi_i(x)$.

Пусть $\varphi_i(x) = a_i + b_i x + c_i x^2 + d_i x^3$,

где $a_i$, $b_i$, $c_i$, $d_i$ - коэффициенты сплайна, определяемые из следующих условий:

- равенство значений $\varphi_i(x)$ и аппроксимируемой функции $f(x)$ в узлах:

$\varphi_i(x_{i-1}) = y_{i-1}$, $\varphi_i(x_i) = f_i$;

- непрерывность первой и второй производных от сплайнов во внутренних узлах ($i$=1,...,$n$-1):

$\varphi_i'(x_i) = \varphi_{i+1}'(x_i)$, $\varphi_i''(x_i) = \varphi_{i+1}''(x_i)$.

Перечисленных условий недостаточно, необходимы ещё два, в качестве которых выступают граничные (краевые) условия. Значение второй производной в крайних точках таблицы принимают равными нулю:

$\varphi_1''(x_0) = 0$, $\varphi_n''(x_n) = 0$.

Из этих соотношений складывается полная система содержащая 4$n$ линейных алгебраических уравнений, решением которой будут все искомые коэффициенты. Матрица такой системы будет сильно разряженной, что очевидно уже из постановки задачи: отдельные сплайны непосредственно влияют только на соседние. Систему уравнений можно существенно изменить в лучшую сторону, если в ней останутся только $c_i$, а остальные коэффициенты будут вычисляться на основе их значений по некоторым простым формулам. Для этого $\varphi_i(x)$ нужно определить так:

$$\varphi_i(x) = a_i + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2 + d_i(x - x_{i-1})^3$$

Тогда условия равенства $\varphi_i(x)$ и $f(x)$ в узловых точках примут вид:

$$a_i = f_{i-1}, \tag{3.3}$$

$$a_i + b_i h_i + c_i h_i^2 + d_i h_i^3 = f_i, \tag{3.4}$$

где $h_i = x_{i-1} - x_i$.

С учётом выражений для производных -

$$\varphi'(x) = b_i + 2c_i(x - x_{i-1}) + 3d_i(x - x_{i-1})^2,$$

$$\varphi''(x) = 2c_i + 6d_i(x - x_{i-1})$$

из условий их непрерывности в точке $x_i$ получатся следующие соотношения:

$$b_i + 2c_i h_i + 3d_i h_i^2 = b_{i+1}, \tag{3.5}$$

$$c_i + 3d_i h_i = c_{i+1}. \tag{3.6}$$

Из граничных условий на основании выражения для второй производной можно записать, что

$$c_1 = 0\,, \tag{3.7}$$

$$c_n + 3d_n h_n = 0\,. \tag{3.8}$$

На этом этапе коэффициенты $a_i$ и $c_1$ уже найдены. Для построения системы относительно оставшихся $c_i$, $i = 2,\ldots n$ Из уравнения (3.6) нужно выразить $d_i$:

$$d_i = \frac{c_{i+1} - c_i}{3h_i} \tag{3.9}$$

Объединяя уравнения (3.3), (3.4) с соотношением (3.9) коэффициенты $b_i$ также выражаются через $c_i$:

$$b_i = \frac{f_i - f_{i-1}}{h_i} - \frac{(c_{i+1} + 2c_i)h_i}{3} \tag{3.10}$$

После подстановки выражений (3.9) и (3.10) в соотношение (3.5) будет получено уравнение, в которое входят только неизвестные коэффициенты $c_i$. Для симметричности записи в полученном уравнении значение индекса $i$ уменьшено на единицу:

$$h_{i-1}c_{i-1} + 2(h_{i-1} + h_i)c_i + h_i c_{i+1} = 3\left[\frac{(f_i - f_{i-1})}{h_i} - \frac{(f_{i-1} - f_{i-2})}{h_{i-1}}\right],\ i = 2,\ldots n \tag{3.11}$$

При $i=n$, учитывая граничное условие (3.6), в (3.11) следует считать

$$c_{n+1} = 0 \tag{3.12}$$

Таким образом, $n$-1 уравнение вида (3.11) вместе с (3.7) и (3.12) образует систему линейных алгебраических уравнений для определения коэффициентов $c_i$.

При условии постоянного шага т.е. $h_i = h_{i-1} = h$, для всех $i$=2,...,$n$, систему можно упростить:

$$hc_{i-1} + 4hc_i + hc_{i+1} = \frac{3}{h}(f_i - 2f_{i-1} + f_{i-2})\,,\ i = 2,\ldots n\,.$$

Коэффициенты $b_i$ и $d_i$ вычисляются по (3.9), (3.10), с учетом, (3.12).

Ниже показан пример вычисления коэффициентов $c_i$ для значений функции, приведенных в таблице 3.4.

Таблица 3.4

| $x$ | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| $f(x)$ | 1 | 3 | 1 | 4 | 2 |

С учётом постоянного шага $h = x_i - x_{i-1}$, $i = 1,\ldots,4$ и того, что $c_1$=0, система уравнений (3.11) принимает вид:

$$\begin{pmatrix} 4h & h & 0 \\ h & 4h & h \\ 0 & h & 4h \end{pmatrix} \cdot \begin{pmatrix} c_2 \\ c_3 \\ c_4 \end{pmatrix} = \begin{pmatrix} \frac{3}{h}(f_2 - 2f_1 + f_0) \\ \frac{3}{h}(f_3 - 2f_2 + f_1) \\ \frac{3}{h}(f_4 - 2f_3 + f_2) \end{pmatrix}.$$

После подстановки значений, в матричной форме получается:

$$\begin{pmatrix} 4 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 4 \end{pmatrix} \cdot \begin{pmatrix} c_2 \\ c_3 \\ c_4 \end{pmatrix} = \begin{pmatrix} 3(1-6+1) \\ 3(4-2+3) \\ 3(2-8+1) \end{pmatrix},$$

а в виде отдельных уравнений:

$$\begin{aligned} 4c_2 + c_3 &= -12 \\ c_2 + 4c_3 + c_4 &= 15 \\ c_3 + 4c_4 &= -15 \end{aligned}.$$

Ниже приведено решение системы, округленное до третьего знака:

$c_2$ = -4.554

$c_3$ = 6.214

$c_4$ = -5.304.

## 4 ЧИСЛЕННОЕ ДИФФЕРЕНЦИРОВАНИЕ

Производная функции есть предел отношения приращения функции к приращению аргумента при стремлении к нулю приращения аргумента:

$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{\Delta f}{\Delta x}$, где $\Delta f = f(x + \Delta x) - f(x)$.

При численном нахождении производной отношение бесконечно малых приращений функций и аргумента заменяют отношением конечных разностей. Очевидно, что чем меньше будет приращение аргумента, тем точнее численное значение производной.

Приращение аргумента задается тремя способами, откладывая $\Delta x = h$ вправо, влево и в обе стороны от исследуемой точки. Соответственно получается три двухточечных метода численного дифференцирования:

$$\frac{df(x)}{dx} \approx \frac{\Delta f}{\Delta x} = \frac{f(x + \Delta x) - f(x)}{\Delta x} = \frac{f_{i+1} - f_i}{h}, \tag{4.1}$$

$$\frac{dy(x)}{dx} \approx \frac{\Delta f}{\Delta x} = \frac{f(x) - f(x - \Delta x)}{\Delta x} = \frac{f_i - f_{i-1}}{h}, \tag{4.2}$$

$$\frac{df(x)}{dx} \approx \frac{\Delta f}{\Delta x} = \frac{f(x + \Delta x) - f(x - \Delta x)}{2\Delta x} = \frac{f_{i+1} - f_{i-1}}{2h}, \tag{4.3}$$

где $h = x_{i+1} - x_i$.

Аналогичные формулы можно получить дифференцированием интерполяционного многочлена первой степени, построенного по двум точкам.

Вторая производная вычисляется как первая производная от первой производной. Расчетная формула имеет вид:

$$\frac{d^2 f(x)}{dx^2} = \frac{d}{dx}\left(\frac{df(x)}{dx}\right) \approx \frac{\frac{f_{i+2} - f_i}{2h} - \frac{f_i - f_{i-2}}{2h}}{2h} = \frac{f_{i+2} - 2f_i + f_{i-2}}{4h^2} \tag{4.4}$$

или $\frac{d^2 f(x)}{dx^2} \approx \frac{f_{i+2} - 2f_i + f_{i-2}}{\tilde{h}^2}$, где $\tilde{h} = x_{i+2} - x_i$.

Формулу для приближенного вычисления второй производной можно получить, дважды дифференцируя интерполяционный многочлен в форме Ньютона, построенный по трем последовательным равноотстоящим точкам. Слагаемые содержащие $x$ в первой степени обратятся в ноль, останется взять производную члена содержащего $x^2$. С учётом выражения для разделенной разности второго порядка будет получено соотношение идентичное (4.4):

$$f''(x) \approx \frac{d^2\left(x^2 f\left(x_0; x_1; x_2\right)\right)}{dx^2} = 2\,\frac{\dfrac{f(x_0) - f(x_1)}{x_0 - x_1} - \dfrac{f(x_1) - f(x_2)}{x_1 - x_2}}{x_0 - x_2}.$$

### 4.1 Погрешности вычисления производных

Поведение погрешности вычисления производной на отрезке $[x_0;x_1]$ можно проанализировать глядя на рисунок 4.1. В середине отрезка хорда почти совпадает с касательной, и погрешность значительно меньше чем на концах. Очевидно, что погрешность уменьшается при уменьшении шага $h=x_1-x_0$.

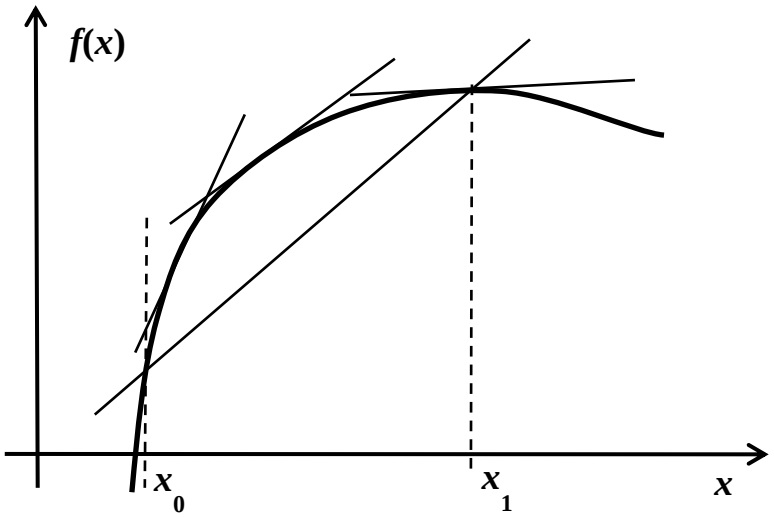


Рисунок 4.1

Используя разложение $f(x)$ в ряд Тейлора по степеням $h$ можно записать:

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2!}f''(x) + \frac{h^3}{3!}f'''(x) + \frac{h^4}{4!}f^{(4)}(x) + \ldots, \qquad (4.5)$$

$$f(x-h) = f(x) - hf'(x) + \frac{h^2}{2!}f''(x) - \frac{h^3}{3!}f'''(x) + \frac{h^4}{4!}f^{(4)}(x) - \ldots. \qquad (4.6)$$

Из (4.5) для первой производной вычисленной по (4.1) следует:

$$f'(x) = \frac{f(x+h) - f(x)}{h} - \frac{h}{2}f''(x) - \ldots$$

Из (4.6) для первой производной вычисленной по (4.2) следует:

$$f'(x) \approx \frac{f(x) - f(x-h)}{h} + \frac{h}{2}f''(x) + \ldots.$$

Вычитанием (4.6) из (4.5) первой производной вычисленной по (4.3) можно получить:

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} + \frac{h^2}{6} f'''(x) + \ldots.$$

Для второй производной сложением (4.5) и (4.6) можно получить:

$$f''(x) \approx \frac{f(x-h) - 2f(x) + f(x+h)}{h^2} + \frac{h^2}{12} f^{(4)}(x) + \ldots.$$

Говорят, что односторонние формулы для первой производной имеют ***первый порядок*** точности ***относительно шага*** т.е. погрешность есть **O(*h*)**. Погрешность симметричной формулы и формулы для второй производной **O($h^2$)**.

Слишком сильное уменьшение шага может привести не к ожидаемому, согласно предыдущим рассуждениям, уменьшению погрешности, а наоборот – к её росту. Дело в том, что свою роль в общей оценке погрешности должна играть погрешность вычисления функции. Пусть $f'(x) \approx \frac{f(x_1) - f(x_0)}{h}$, тогда погрешность метода $r_1$ можно оценить как $r_1 \le \frac{M_2 h}{2}$, где $M_2 = \max\limits_{x_0 \le x \le x_1} \left| f''(x) \right|$.

Если вычисления $f(x)$ содержат погрешности не превосходящие $E$, то $f'(x)$ будет дополнительно содержать погрешность $r_2 \le \frac{2E}{h}$. Общую погрешность $r$ можно оценить неравенством:

$$\left|r\right| \le \left|r_1\right| + \left|r_2\right| \le g(h) = \frac{M_2 h}{2} + \frac{2E}{h}$$

Как показано на рисунке 4.2, дальнейшее уменьшение шага после $h_{\text{опт}}$ приводит к возрастанию погрешности.

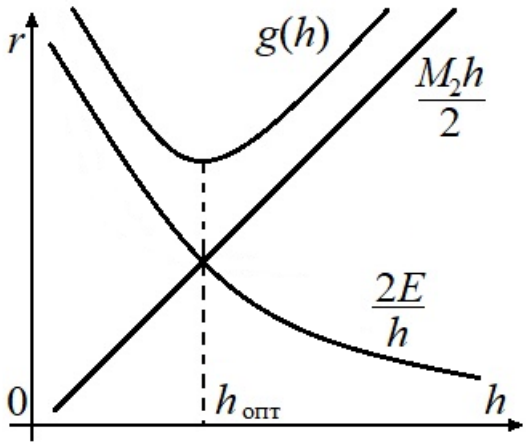


Рисунок 4.2

### 4.2 Задание к лабораторной работе

1. Запрограммировать приближенные вычисления первой производной функции $f(x) = v \cdot x^2$, где $v$ – номер варианта, в точках $10^{-2}$,$10^{-1}$,1,10,$10^{2}$. В программе использовать 32 битное представление вещественных чисел. Формула вычисления производной задаётся преподавателем. Сделать вычисления с различным шагом: $10^{-1}$, $10^{-2}$,...,$10^{-8}$. Экспериментально подобрать оптимальный шаг – шаг позволяющий добиться наилучшей точности. Объяснить полученные результаты.

2. Для функции заданной таблицей значений согласно варианту задания в таблице 3.2 выполнить «вручную» (в любом доступном математическом программном обеспечении) расчеты коэффициентов кубического сплайна по формулам (3.3), (3.7), (3.9)-(3.11). Для рассчитанных сплайнов построить графики.

3. Разработать программную реализацию построения кубического сплайна по формулам (3.3), (3.7), (3.9)-(3.11) для произвольной таблицы значений функции. Сравнить результаты работы программы с графиками, построенными в пункте 2. Рассчитать и построить графики первой и второй производной полученного приближения функции. Для первой производной использовать выражение (4.3). *Непрерывность производных дополнительно подтвердит отсутствие ошибок в реализации сплайна*.

Для решения системы уравнений (3.11) использовать реализацию метода прогонки **из соответствующей лабораторной работы**. Графики отображать как

кусочно-линейную аппроксимацию с малым (~$10^{-2}$) шагом. **Точки исходной таблицы отобразить отдельно**. Ограничений на среду разработки не накладывается, однако рекомендуется использовать современные высокоуровневые возможности, позволяющие не тратить время на программирование построения графиков (компонент Chart в C# Windows Forms, библиотека QCustomPlot в Qt/C++, компонент Tchart в Embarcadero C++ Builder и т.п.).

Отчёт по лабораторной работе должен содержать:

- результаты, полученные в пункте 1, с объяснением;
- исходную таблицу значений функции;
- результаты и промежуточные данные «ручных» расчётов в сторонних программных пакетах (систему уравнений, коэффициенты сплайнов, графики);
- тексты разработанных программ;
- результаты, полученные с помощью разработанных программ (графики сплайнов и производных, с выделением точек таблицы).

# 5 ЧИСЛЕННОЕ ИНТЕГРИРОВАНИЕ

Пусть требуется найти значение $\int_a^b f(x)dx$, где $f(x)$ некоторая функция, непрерывная на интервале $(a, b)$. В большинстве существующих методов численного интегрирования для повышения точности исходный интервал $(a, b)$ разбивается на $n$ меньших интервалов $(x_i; x_{i+1})$, где $i=0,\ldots n$; $a=x_0\leq x_1\leq\ldots\leq x_n=b$. Интеграл при этом вычисляется как сумма площадей «полосок», получаемых при таком разбиении. Отличие методов состоит в способе аппроксимации $f(x)$ на отрезке $(x_i; x_{i+1})$. В методах ***Ньютона-Котеса*** для аппроксимации используются полиномы различных степеней. К этой группе относятся методы прямоугольников, трапеций, Симпсона.

## 5.1 Метод прямоугольников

Из геометрических соображений очевидно, что интеграл можно приближенно заменить площадью прямоугольника. Если площадь

прямоугольников определить как произведение значения подынтегральной функции и величины $h=(x_{i+1}-x_i)$, называемой шагом интегрирования, то будет получено приближенное равенство

$$\int_a^b f(x)dx \approx \sum_{i=0}^{n-1} h \cdot f(x_i) = h \cdot \sum_{i=0}^{n-1} f(x_i), \qquad (5.1)$$

называемое ***метод левых прямоугольников***. Иллюстрация работы метода показана на рисунке 5.1.

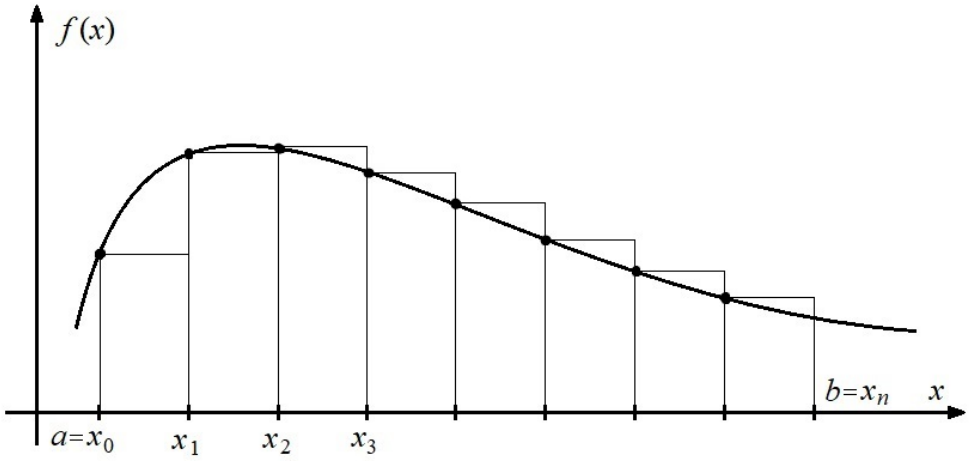


Рисунок 5.1

Соотношение (5.1) можно интерпретировать как замену функции $f(x)$ каждом интервале $(x_i; x_{i+1})$ интерполяционным многочленом нулевой степени.

Для интервала от $x_i$ до $x_i+h$, используя разложение $f(x)$ в ряд Тейлора вблизи точки $x_i$ можно записать:

$$f(x) = f(x_i) + (x - x_i) f'(x_i) + \frac{(x - x_i)^2}{2!} f''(x_i) + \ldots,$$

тогда

$$\int_{x_i}^{x_i+h} f(x)dx = x \cdot f(x_i)\Big|_{x_i}^{x_i+h} + \frac{(x - x_i)^2}{2} f'(x_i)\Big|_{x_i}^{x_i+h} + \frac{(x - x_i)^3}{3 \cdot 2!} f''(x_i)\Big|_{x_i}^{x_i+h} + \ldots =$$

$$= f(x_i)h + \frac{h^2}{2} f'(x_i) + O(h^3).$$

Это означает, что величина погрешности $R_i$ на отдельном шаге $(x_i; x_i+h)$ есть величина порядка $\frac{h^2}{2}$, а точнее $|R_i| < \frac{h^2}{2} M_{1,i}$, где $M_{1,i} = \max |f'(x)|_{(x_i, x_i+h)}$. Общая погрешность метода $R$ представляет собой сумму по всем шагам:

$R = \sum_{i=0}^{n-1} R_i$, для которой справедливо неравенство:

$$|R| \le \sum_{i=0}^{n-1} \frac{h^2}{2} M_1 = n \frac{h^2}{2} M_1 = \frac{h}{2}(b-a) M_1, \qquad (5.2)$$

где $M_1 = \max |f'(x)|_{(a,b)}$.

Если вычислить площадь прямоугольников как показано на рисунке 5.2, то метод следует назвать ***методом правых прямоугольников***. Оценка погрешности при этом не поменяется.

$$\int_a^b f(x)dx \approx h \cdot \sum_{i=1}^{n} f(x_i) \qquad (5.3)$$

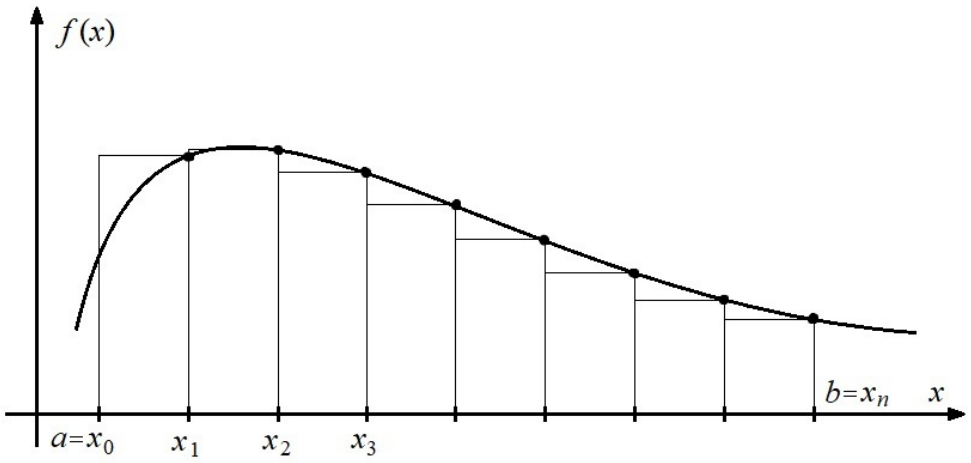


Рисунок 5.2

Метод прямоугольников можно существенно улучшить, если выбирать для аппроксимации значение $f(x)$ в середине интервала – в точке $\bar{x} = \frac{x_i + x_{i+1}}{2}$, в результате чего расчетная формула примет вид:

$$\int_a^b f(x)dx \approx \sum_{i=0}^{n-1} hf\left(\frac{x_i + x_{i+1}}{2}\right) \qquad (5.4)$$

Вычисления по формуле 5.4 называют ***методом средних прямоугольников***. Геометрическая интерпретация приведена на рисунке 5.3.

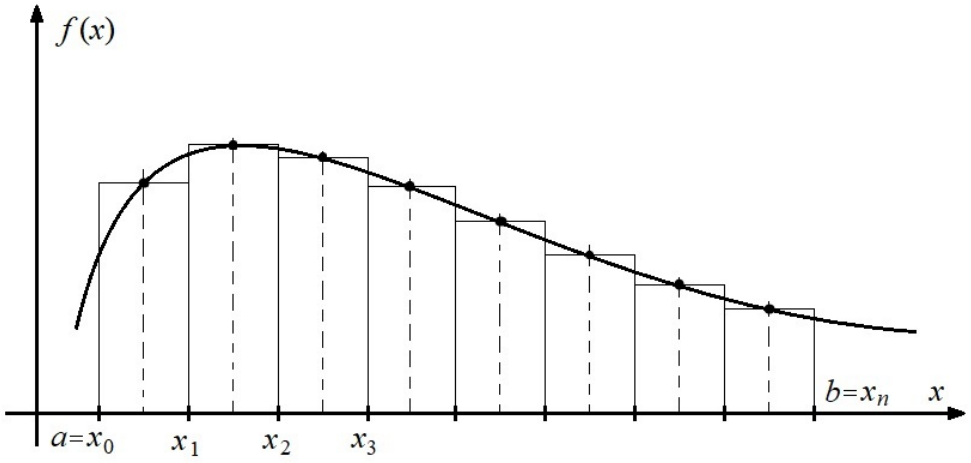


Рисунок 5.3

При анализе погрешности можно заметить, что второй член разложения $f(x)$ в ряд Тейлора вблизи точки $\bar{x} = \frac{x_i + x_{i+1}}{2} = x_i + \frac{h}{2}$ при вычислении интеграла сократится:

$$\int_{x_i}^{x_i+h} (x - \bar{x}) f'(\bar{x}) dx = \frac{(x - \bar{x})^2}{2} f'(\bar{x}) \Bigg|_{x_i}^{x_i+h} = f'(\bar{x}) \left( \frac{(x_i + h - \bar{x})^2}{2} - \frac{(x_i - \bar{x})^2}{2} \right) =$$

$$= f'(\bar{x}) \left( \frac{\left( x_i + h - x_i - \frac{h}{2} \right)^2}{2} - \frac{\left( x_i - x_i - \frac{h}{2} \right)^2}{2} \right) = 0 .$$

Вследствие этого оценка погрешности изменится в лучшую строну:

$$R \le \frac{h^2}{24} M_2 (b - a), \tag{5.5}$$

где $M_2 = \max \left| f''(x) \right|_{(a,b)}$.

### 5.2 Метод трапеций

В методе трапеций аппроксимация $f(x)$ осуществляется полиномом первой степени как показано на рисунке 5.4. Значение интеграла на каждом интервале $(x_i; x_{i+1})$ вычисляется как площадь прямоугольной трапеции с основаниями

равными значениям функции на границах интервала и высотой равной величине шага:

$$\int_{x_i}^{x_{i+1}} f(x)dx \approx \frac{h}{2}\left( f(x_i) + f(x_{i+1})\right).$$

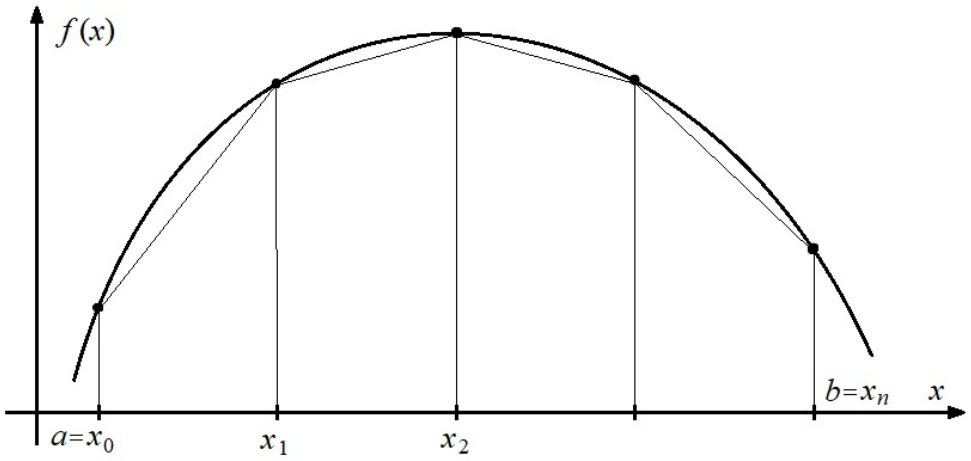


Рисунок 5.4

Значения функции на концах отрезка входят только в крайние трапеции, остальные учитываются в двух, таким образом приближенное значение интеграла вычисляется по формуле:

$$\int_a^b f(x)dx \approx \frac{h}{2}\left( f(x_0) + 2\sum_{i=1}^{n-1} f(x_i) + f(x_n)\right) \tag{5.6}$$

Погрешность метода трапеций того же порядка относительно шага интегрирования, как и у средних прямоугольников, но оценка выше, чем у метода средних прямоугольников:

$$\left|R\right| \le \frac{h^2}{12} M_2 (b-a). \tag{5.7}$$

### 5.3 Метод Симпсона

В методе Симпсона подынтегральная функция $f(x)$ заменяется интерполяционным полиномом второй степени $P_2(x)$ – параболой, проходящей через три точки $x_i$, $x_{i+1}$, $x_{i+2}$, как показано на рисунке 5.5.

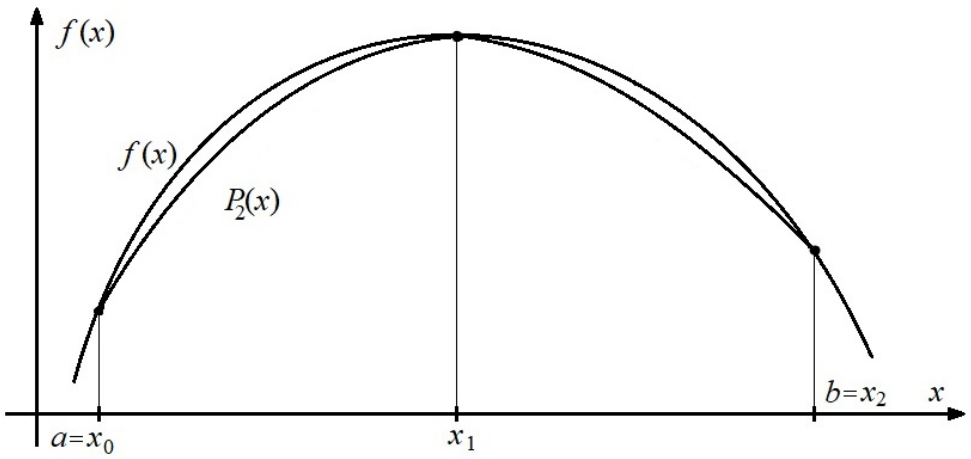


Рисунок 5.5

Выражение для $P_2(x)$ можно записать в форме Ньютона:

$$P_2(x) = f(x_i) + \frac{x - x_i}{h}\left(f(x_{i+1}) - f(x_i)\right) + \frac{(x - x_i)(x - x_{i+1})}{2h^2}\left(f(x_i) - 2f(x_{i+1}) + f(x_{i+2})\right).$$

Если обозначить $x - x_i = z$ тогда,

$$P_2(z) = f(x_i) + \frac{z}{h}\left(f(x_{i+1}) - f(x_i)\right) + \frac{z(z - h)}{2h^2}\left(f(x_i) - 2f(x_{i+1}) + f(x_{i+2})\right) =$$

$$= f(x_i) + \frac{z}{2h}\left(-3f(x_i) + 4f(x_{i+1}) - f(x_{i+2})\right) + \frac{z^2}{2h^2}\left(f(x_i) - 2f(x_{i+1}) + f(x_{i+2})\right).$$

Интеграл по интервалу $(x_i, x_{i+2})$ выражается через значения функции в точках:

$$\int_{x_i}^{x_{i+2}} P_2(x)dx = \int_0^{2h} P_2(z)dz =$$

$$= 2hf(x_i) + \frac{(2h)^2}{4h}\left(-3f(x_i) + 4f(x_{i+1}) - f(x_{i+2})\right) + \frac{(2h)^3}{6h^2}\left(f(x_i) - 2f(x_{i+1}) + f(x_{i+2})\right) =$$

$$= \frac{h}{3}\left(6f(x_i) - 9f(x_i) + 12f(x_{i+1}) - 3f(x_{i+2}) + 4f(x_i) - 8f(x_{i+1}) + 4f(x_{i+2})\right) =$$

$$= \frac{h}{3}\left(f(x_i) + 4f(x_{i+1}) + f(x_{i+2})\right).$$

С учетом полученного выражения, общая формула примет вид:

$$\int_a^b f(x)dx \approx \frac{h}{3}\left(f(x_0) + 4f(x_1) + 2f(x_2) + \ldots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n)\right) \quad (5.8)$$

На рисунке 5.6 проиллюстрировано применение формулы (5.5) при $n = 4$. При уменьшении $h$ следует учитывать, что для корректной работы метода $n$ может быть только чётным.

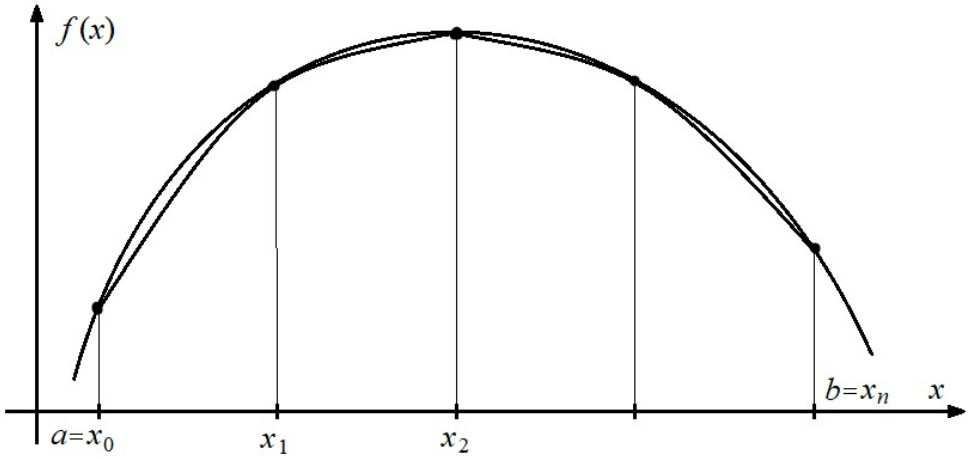


Рисунок 5.6

Метод Симпсона представляет собой метод второго порядка аппроксимирующего многочлена, при этом имеет четвертый порядок точности относительно шага. Оценка погрешности определяется неравенством:

$$\left|R\right| \le \frac{h^4}{180} M_4 (b - a), \tag{5.9}$$

где $M_4 = \max\limits_{a \le x \le b} \left| f^{(IV)}(x) \right|$.

### 5.4 Методы более высоких порядков

При аппроксимации подынтегральной кривой многочленом $k$-й степени требуется как минимум ($k$+1) точка, а формула для вычисления значения интеграла при минимально возможном $n=k$ будет иметь следующий вид:

$$\int_{x_0}^{x_k} f(x)dx \approx c_0 h \sum_{i=0}^{k} \omega_i f(x_i)$$

Коэффициенты методов Ньютона-Котеса представлены в таблице 5.1.

Таблица 5.1

| $k$ | $c_0$ | $\omega_0$ | $\omega_1$ | $\omega_2$ | $\omega_3$ | $\omega_4$ | $\omega_5$ |
|---|---|---|---|---|---|---|---|
| 1 | 1/2 | 1 | 1 | - | - | - | - |
| 2 | 1/3 | 1 | 4 | 1 | - | - | - |
| 3 | 3/8 | 1 | 3 | 3 | 1 | - | - |
| 4 | 2/45 | 7 | 32 | 12 | 32 | 7 | - |
| 5 | 5/288 | 19 | 75 | 50 | 50 | 75 | 19 |

При уменьшении изменении $h$ следует учитывать, что $n$ должно быть кратно $k$.

Метод Ньютона-Котеса 3-го порядка аппроксимации имеет 4-й порядок точности относительно шага интегрирования, 4-го и 5-го порядков - 6-й порядок точности.

### 5.5 Выбор шага интегрирования

В основе алгоритмов с автоматическим выбором шага лежит ***правило Рунге*** Пусть для вычисления некоторой функции $g(x)$ используется приближенная формула $\varphi(x,h)$ и

$$g(x)-\varphi(x,h)=M h^p, \tag{5.10}$$

где $M$ – константа.

При вычислении с шагом $rh$ получим:

$$g(x)-\varphi(x, rh)=M (rh)^p. \tag{5.11}$$

Тогда вычитая (5.9) из (5.11) получим:

$$M h^p = \frac{\varphi(x,h) - \varphi(x,rh)}{r^p - 1}.$$

Пусть при последовательном уменьшении шага приближение вычисленное с шагом $h$ обозначено $S_h$ , а $S_{h/r}$ – с шагом $h/r$ (в $r$ раз меньше), тогда процесс уменьшения $h$ можно прекратить если

$$\frac{\left|S_h - S_{h/r}\right|}{r^p - 1} < \varepsilon, \tag{5.12}$$

где $\varepsilon$ - заданная точность, $p$ – порядок точности метода.

### 5.6 Задание к лабораторной работе

1. Проанализировать поведение подынтегральной функции, заданной согласно варианту задания в таблице 5.2. Вычислить оценки шага интегрирования для заданных преподавателем методов и значений точности из таблицы 5.2 используя формулы (5.2), (5.5), (5.7), (5.9). Расчеты выполнить вручную или с использованием любых доступных математических программных пакетов.

2. Разработать программную реализацию вычисления интеграла заданными методами с контролем точности основанном на правиле Рунге с учётом порядка точности метода. Входной информацией для разработанной программы должны быть: пределы интегрирования и требуемая точность. Подынтегральная функция жестко задаётся в программе. Выходная информация: значение интеграла и шаг, при котором оно вычислено. При разработке алгоритма стараться по возможности **минимизировать вычислительные затраты.** Ограничений на среду разработки не накладывается.

3. Сравнить оценки шага со значениями шага, которые получены в результате работы программ, а значения интеграла полученные программно с результатом, полученным в стороннем математическом программном пакете с высокой точностью.

Отчёт по лабораторной работе должен содержать:

- расчёт оценок шага;
- тексты разработанных программ;
- результаты работы программы;
- анализ полученных результатов.

Таблица 5.2

| № варианта | $f(x)$ | $(a; b)$ | ε |
|---|---|---|---|
| 1 | $\frac{1+x}{(2+3x)^2}$ | 1;3 | $10^{-2}$; $10^{-3}$ |
| 2 | $\frac{1+x}{(2+3x)^2 \cdot \sqrt{2+x}}$ | 5;15 | $10^{-3}$; $10^{-4}$ |
| 3 | $\frac{(1+x)^2}{x^3 \cdot \sqrt{2+x}}$ | 2;5 | $10^{-2}$; $10^{-3}$ |
| 4 | $\frac{1+\sqrt{x}}{1+4x+3x^2}$ | 0.5;4.5 | $10^{-2}$; $10^{-3}$ |
| 5 | $\frac{2.5x^2-0.1}{\ln(x)+1}$ | 3;13 | 1; $10^{-1}$ |
| 6 | $\frac{\ln(x)}{\sqrt{1.2+0.3x}}$ | 5;10 | $10^{-1}$; $10^{-2}$ |
| 7 | $\frac{(x+1)^2}{\sqrt{\ln(x)}}$ | 3;7 | 1; $10^{-1}$ |
| 8 | $\sqrt{x}e^{-\frac{x}{2}}$ | 2;10 | $10^{-2}$; $10^{-3}$ |
| 9 | $\frac{2x^2\ln(x)}{10+x^2}$ | 1;8 | $10^{-1}$; $10^{-2}$ |
| 10 | $\frac{2x-\sin(x)}{10+x^2}$ | 1;5 | $10^{-2}$; $10^{-3}$ |

## 6 РЕШЕНИЕ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

Пусть известно $\frac{dy}{dx} = f(x, y)$ и начальная точка $y(x_0)=y_0$, задачу поиска кривой $y(x)$, удовлетворяющей перечисленным соотношениям, называют задачей Коши для дифференциального уравнения $y' = f(x, y)$ с начальными условиями $(x_0,y_0)$. При использовании численных методов значения непрерывной функции $y(x)$, являющейся решением, вычисляются на конечном дискретном множестве значений аргумента: $x_0<x_1<\ldots<x_n$.

Методы, которые сводят решение к набору рекуррентных соотношений, позволяющих по предыдущим уже вычисленным значениям $y(x)$ найти следующие называют ***явными***.

***Неявными*** называют методы, в которых для определения значения $y_{n+1}$ требуется решить в общем случае нелинейное, зависящее от вида функции $f(x,y)$ уравнение относительно $y_{n+1}$.

***Одношаговыми*** называют методы, в которых для вычисления $y_{n+1}$ требуется знать только одно предыдущее значение – $y_n$.

Если для вычисления $y_{n+1}$ требуется знать несколько предыдущих значений $y_n, y_{n-1},\ldots, y_{n-m}$, то такие методы называют ***многошаговыми***.

### 6.1 Метод Эйлера

Решение в точке $x_{n+1}$, полученное по методу Эйлера, совпадает с разложением $y(x)$ в окрестности этой точки в ряд Тейлора до членов порядка $h$, где $h = x_{n+1} - x_n$.

Суть метода легко пояснить графически: предположим, что точка $(x_n, y_n)$ известна, тогда следующей точкой решения можно считать точку на прямой $y = y_n + y'_n(x - x_n)$ с абсциссой $x_{n+1}$, как показано на рисунке 6.1.

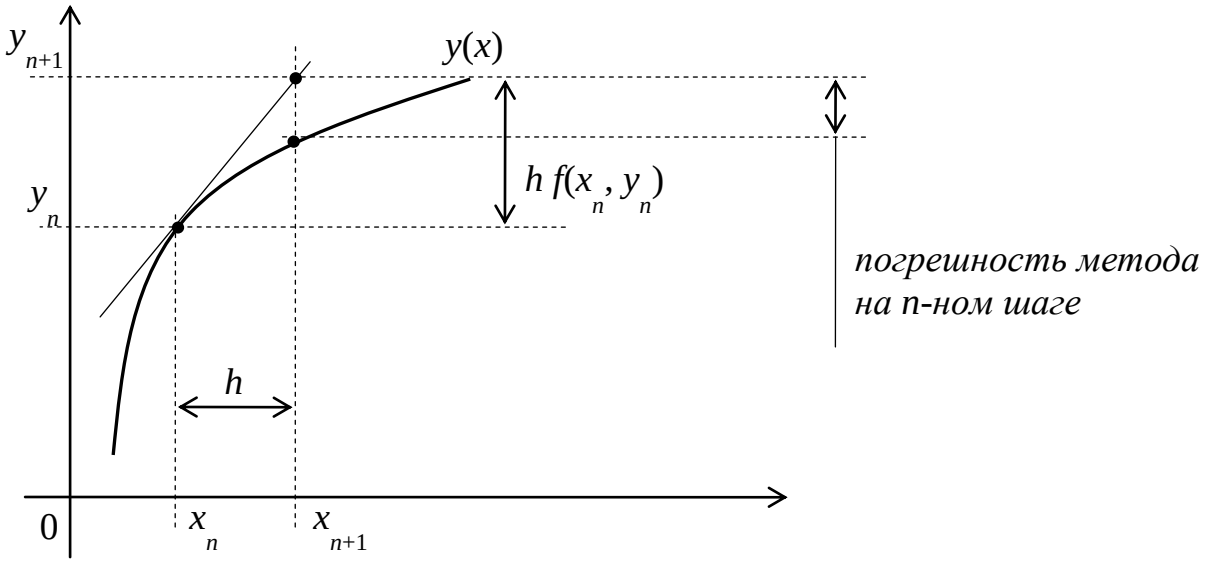


Рисунок 6.1

После замены производной в уравнении прямой на правую часть расчетная формула будет выглядеть следующим образом:

$$y_{n+1} = y_n + h\, f(x_n, y_n). \qquad (6.1)$$

Этот метод имеет довольно большую погрешность, кроме того, он часто оказывается неустойчивым – малая ошибка, возникшая на шаге, влияет на последующие шаги. Рост накопленной ошибки может быть настолько интенсивным, что погрешность начинает доминировать над решением, вплоть до переполнения разрядной сетки ЭВМ.

### 6.2 Модификации метода Эйлера

***Исправленный метод Эйлера*** – это модификация метода, увеличивающая порядок точности относительно шага до второго. При этом для вычисления решения в следующей точке требуется два вычисления правой части – в предыдущей точке $(x_n, y_n)$ и в точке, найденной по методу Эйлера: $\left(x_n + h, y_n + hf\left(x_n, y_n\right)\right)$.

Расчетная формула исправленного метода получается в результате подстановки усредненного значения в (6.1):

$$y_{n+1} = y_m + h\,\frac{f\left(x_n, y_n\right) + f\left(x_n + h, y_n + hf\left(x_n, y_n\right)\right)}{2} \qquad (6.2)$$

Графически это можно интерпретировать как усреднение угла наклона касательных, иллюстрация приведена на рисунке 6.2.

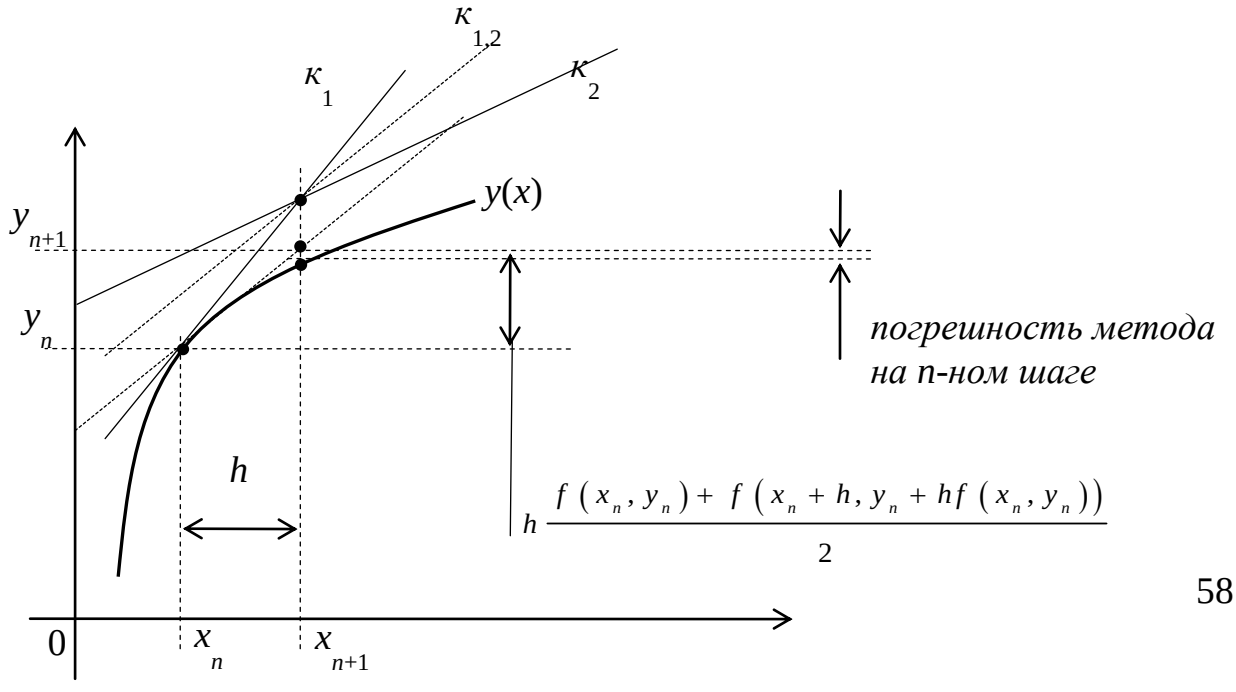

Рисунок 6.2

***Модифицированный метод Эйлера*** представляет собой другой способ увеличения порядка точности метода до второго. Угол наклона касательной вычисляется в средней точке $x_n + \frac{h}{2}$. Графическая интерпретация приведена на рисунке 6.3. Через точку ($x_n$, $y_n$) проведена прямая $K_0$ с тангенсом угла наклона, равным $f(x_n, y_n)$, она позволяет определить точку $\left( x_n + \frac{h}{2}, y_n + \frac{h}{2} f(x_n, y_n) \right)$. Расчетная формула метода получается в результате замены точки ($x_n$, $y_n$) в (6.1) этой средней точкой:

$$y_{n+1} = y_n + hf\left( x_n + \frac{h}{2}, y_n + \frac{h}{2} f(x_n, y_n) \right) \qquad (6.3)$$

Графически это можно представить как использование прямой $K_2$ параллельной $K_1$.

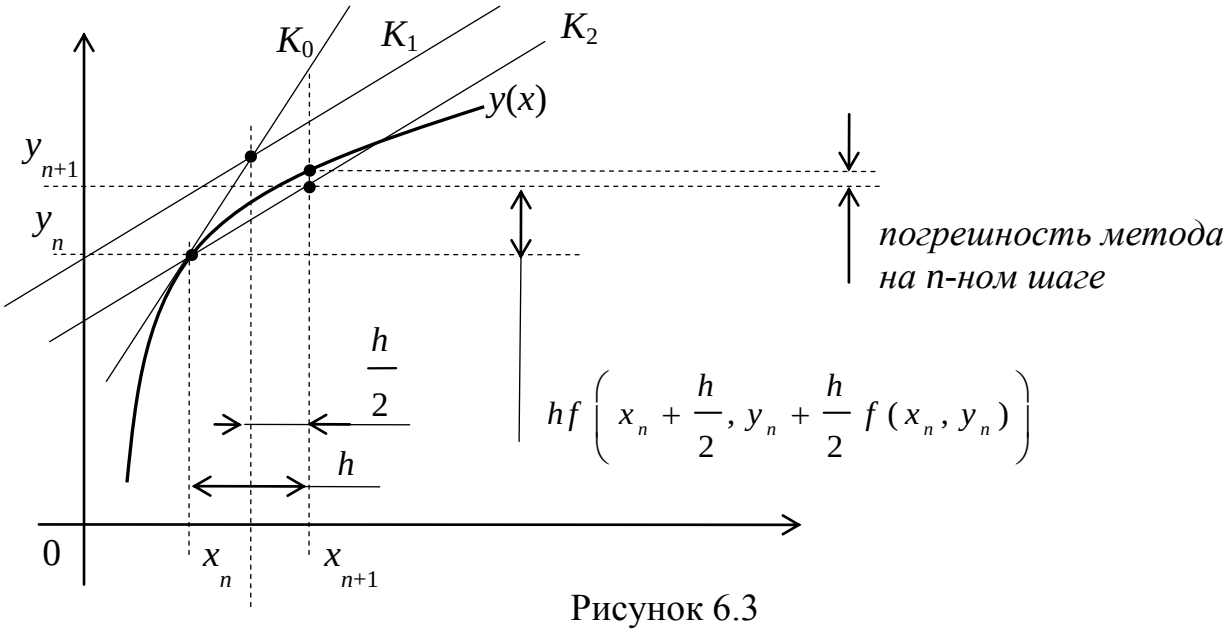


Рисунок 6.3

### 6.3 Метод Рунге-Кутты четвертого порядка

Перечисленные методы являются частными случаями ***методов Рунге-Кутты*** различного порядка точности, это одношаговые явные методы. Наиболее часто на практике применяется метод четвертого порядка, описанный в большинстве книг по методам вычислений. Он обеспечивает малую погрешность для широкого класса уравнений и, при этом, довольно экономичен (на каждом шаге интегрирования требуется вычисление четырех значений функции правой части $f(x,y)$):

Расчетная формула выглядит так:

$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4), \text{ где}$$

$$k_1 = f(x_n, y_n)$$

$$k_2 = f(x_n + h/2, y_n + hk_1/2)$$

$$k_3 = f(x_n + h/2, y_n + hk_2/2)$$

$$k_4 = f(x_n + h, y_n + hk_3).$$

### 6.4 Метод Рунге–Кутты–Мерсона (с контролем точности)

Метод Мерсона представляет собой алгоритм ***с автоматическим изменением*** шага сетки, основанный на контроле точности вычислений. Для каждой точки правая часть вычисляется пять раз, это позволяет оценить погрешность текущего шага.

Алгоритм метода можно записать так:

**Шаг 1.** Вычисляются $k_i$:

$$k_1 = hf(x_n, y_n)$$

$$k_2 = hf\left(x_n + \frac{1}{3}h, y_n + \frac{1}{3}k_1\right)$$

$$k_3 = hf\left(x_n + \frac{1}{3}h, y_n + \frac{1}{6}k_1 + \frac{1}{6}k_2\right)$$

$$k_4 = hf\left(x_n + \frac{1}{2}h, y_n + \frac{1}{8}k_1 + \frac{3}{8}k_3\right)$$

$$k_5 = hf\left( x_n + h, y_n + \frac{1}{2}k_1 - \frac{3}{2}k_3 + 2k_4 \right)$$

**Шаг 2.** Находится оценка локальной ошибки

$$\delta_n = \frac{1}{30}(2k_1 - 9k_3 + 8k_4 - k_5)$$

**Шаг 3.** Если выполняется неравенство $|\delta_n| \geq \varepsilon$, где ε – заданная точность то *h* уменьшается в 2 раза и управление передается на шаг 1.

**Шаг 4.** Вычисляется решение $y_{n+1} = y_n + \frac{1}{6}k_1 + \frac{2}{3}k_4 + \frac{1}{6}k_5$

**Шаг 5.** Если выполняется неравенство $|\delta_n| \leq \frac{\varepsilon}{32}$, то *h* увеличивается в 2 раза.

**Шаг 6.** Выполняется переход к следующей точке.

### 6.4 Семейство явных многошаговых методов Адамса

Пусть известно приближенное решение в четырех первых *k* точках. Тогда в этих точках $f(x,y)$ можно рассматривать как функцию единственного аргумента *x:* $f(x,y)=F(x)$. Если заменить $F(x)$ интерполяционным многочленом, построенным по этим известным точкам, то

$$y_{n+1} = y_n + \int_{x_n}^{x_{n+1}} F(x)dx \approx y_n + \int_{x_n}^{x_{n+1}} P_{k-1}(x)dx .$$

Пусть $k$=4, а расстояние между соседними $x_i$ одинаково и обозначается $h$, тогда после интегрирования на отрезке $(x_n;x_{n+1})$ получим:

$$y_{n+1} \approx y_n + \frac{h}{24}(55F_n - 59F_{n-1} + 37F_{n-2} - 9F_{n-3}).$$

После вычисления очередного значения $y_{n+1}$ можно определить $F_{n+1}= f(x_{n+1}, y_{n+1})$ и перейти к следующей точке ($n$+2). При этом следует сохранить три последних значения $F_{n-2}$, $F_{n-1}$, $F_n$, чтобы не возникло необходимости вычислять их вновь на следующем ($n$+2) шаге. Значение $F_{n-3}$ больше не понадобится, память в которой оно было записано можно использовать повторно. При разработке программы удобно хранить значения *F* в массиве, и сдвигать их на каждой итерации. В этом случае для получения одной точки решения

потребуется всего одно вычисление правой части уравнения. Порядок точности рассмотренного метода относительно шага равен четырем.

Ниже представлены формулы для методов Адамса второго, третьего и пятого порядков:

$$y_{n+1} = y_n + \frac{h}{2}\left(3F_n - F_{n-1}\right),$$

$$y_{n+1} = y_n + \frac{h}{12}\left(23F_n - 16F_{n-1} + 5F_{n-2}\right),$$

$$y_{n+1} = y_n + \frac{h}{720}\left(1901F_n - 2774F_{n-1} + 2616F_{n-2} - 1274F_{n-3} + 251F_{n-4}\right).$$

Метод *k*-го порядка требует предварительного вычисления решения в *k* точках. Если по условиям задачи известно только одно начальное значение, то для вычисления оставшихся приближений рекомендуется воспользоваться одношаговым методом того же порядка точности.

### 6.5 Задание к лабораторной работе

1. Аналитическими методами или сторонними программными средствами найти точное решение задачи Коши для дифференциального уравнения $y'=f(x,y)$. Правая часть уравнения, начальная и конечная точки заданы в таблице 6.1.

2. Разработать программную реализацию вычисления решения задачи Коши методами, заданными преподавателем. Входной информацией для разработанной программы должны быть: координаты начальной точки, ордината конечной точки, шаг интегрирования. Правая часть уравнения и точное решение жестко задаются в программе. Выходная информация: графики численных решений, полученных заданными методами, и график точного решения.

3. Провести эксперименты с различными значениями шага: сравнить результаты с точным решением. Экспериментально подобрать максимальные значения шага, при которых приближенные решения сходятся к точному, минимальные значения шага, при котором приближенные решения расходятся. Подобрать наибольшие значения шага, при которых результаты методов становятся визуально неотличимыми от точного решения.

При разработке алгоритма стараться по возможности **минимизировать вычислительные затраты**.

Ограничений на среду разработки не накладывается, однако рекомендуется использовать современные высокоуровневые возможности, позволяющие не тратить время на программирование построения графиков (компонент Chart в C# Windows Forms, библиотека QCustomPlot в Qt/C++, компонент Tchart в Embarcadero C++ Builder и т.п.).

Отчёт по лабораторной работе должен содержать:

- результаты экспериментов: графики приближенных решений, полученные разными методами, с разным шагом, точное решение;
- анализ полученных результатов;
- тексты разработанных программ.

Таблица 6.1

| № варианта | $f(x, y)$ | $x_0$ | $y_0$ | $x_n$ |
|---|---|---|---|---|
| 1 | $10 - 2y - x$ | 5 | 0 | 10 |
| 2 | $\sin(x) - 2y$ | 0 | 5 | 10 |
| 3 | $\sin(x) - y$ | 2 | 10 | 10 |
| 4 | $\sin(x) - y + 10$ | 0 | 0 | 10 |
| 5 | $x^2 - 5y$ | 0 | 10 | 6 |
| 6 | $y - x^2$ | 0 | 1 | 3 |
| 7 | $2y - yx$ | 0 | 5 | 5 |
| 8 | $y - 2yx$ | 0 | 1 | 3 |
| 9 | $\cos(x) - y + 5$ | 0 | 1 | 7 |
| 10 | $x^2 - x + y$ | 0 | 0 | 2 |

## ЛИТЕРАТУРА


1. Амосов А.А., Дубинский Ю. А., Копченова Н. В. Вычислительные методы: учебное пособие. - 4-е изд., стер. - СПб.: Лань, 2014. - 672 с.
2. Бахвалов Н.С. Численные методы. - М.: Наука, 1973. - 631 с.
3. Калиткин Н.Н. Численные методы. - М.: Наука, 1978. - 512 с.
4. Мудров А. Е. Численные методы для ПЭВМ на языках Бейсик, Фортран и Паскаль. - Томск: МП "РАСКО", 1991. - 272 с.
5. Пирумов У.Г. Численные методы: учебник и практикум для академического бакалавриата. - 5-е изд., перераб. и доп. - М: Юрайт, 2014. - 421 с.
6. Самарский А.А. Введение в численные методы. - СПб.: Лань, 2009. - 288 с.