ИНСТИТУТ ТРАНСПОРТА И СВЯЗИ

Конспект лекций
по предмету

Численные методы

Последнее обновление:
15 января 2018 г.
15:13

Содержание

1 Введение

1.1 Процесс исследования

Наблюдать – понимать – использовать

1. Постановка задачи (дано, найти, ограничения);
2. Построение модели (например: система уравнений. Зависит от «дано» и «найти»);
3. Выбор численного метода;
4. Алгоритм и программа;
5. Отладка и тестирование;
6. Валидация модели;
7. Расчёт / численный эксперимент;
8. Анализ, интерпретация, прогноз.
   (a) Анализ – объяснение текущего поведения;
   (b) Интерпретация – объяснение полученных результатов на основе той теории, на которой построена модель;
   (c) Прогноз – что будет дальше в системе, и что изменить для «улучшения».

1.2 Внутримашинное представление чисел и погрешности

Возможные представления дробных чисел	Пример
С фиксированной точкой	64.256
С плавающей точкой	$6.4256 \cdot 10^1 = 0.064256 \cdot 10^3$
Нормализованная форма с плавающей точкой	$0.64256 \cdot 10^2$

1.2.1 Нормализованная форма с плавающей точкой

$$x = \pm \underbrace{\left(\frac{d_1}{\beta^1} + \frac{d_2}{\beta^2} + \cdots + \frac{d_k}{\beta^k} \right)}_{\text{мантисса}} \cdot \beta^n \tag{1.1}$$

где: k – разрядность мантиссы;
d_i – i-ый разряд мантиссы; $d_i \in \overline{0, \beta - 1}$;
β – основание системы счисления;
n – порядок.

Пусть $L \leq n \leq U$ (т.е. L и U нижняя и верхняя границы порядка) тогда количество чисел которое можно представить в данной форме вычисляется по формуле:

$$N = 2 \cdot (\beta - 1) \cdot \beta^{k-1} \cdot (U - L + 1) + 1 \tag{1.2}$$

1.3 Алгебра погрешностей

1.3.1 Определения

Пусть x – какое-то приближённое значение, тогда

$$x = x_0 \pm \Delta x \tag{1.3}$$

где: x_0 – исходное (точное) значение;
Δx – абсолютная погрешность.

Таким образом, абсолютною погрешность Δx можно вычислить следующим образом:

$$\Delta x = |x - x_0| \tag{1.4}$$

А относительную погрешность δx:

$$\delta x = \frac{\Delta x}{x_0} \tag{1.5}$$

Примечание

В случае когда $x_0 = 0$, относительную погрешность не определяют, а пользуются абсолютной.

1.3.2 Алгебраические уравнения погрешностей

$$\Delta(a \pm b) = \Delta a + \Delta b \tag{1.6}$$
$$\delta(a \cdot b) = \delta a + \delta b \tag{1.7}$$
$$\delta(a/b) = \delta a + \delta b \tag{1.8}$$
$$\delta(a^n) = n \cdot \delta a \tag{1.9}$$
$$\Delta(f(x)) \approx \left|f'(x_0)\right| \cdot \Delta x \tag{1.10}$$

Таким образом, только в случае возведения в степень (1.9), и только при $|n| < 1$, погрешность уменьшается. Во всех остальных случаях – только накапливается.

1.4 Классификация вычислительных методов

1.4.1 Прямые

При прямых методах сложность вычислений задачи зависит только от сложности исходной задачи.

1.4.2 Итерационные

При итерационных методах задаётся требуемая точность ϵ, от которой начинает зависеть сложность вычислений задачи.

1.5 Понятие обусловленности задачи

Обусловленной называется задача, для которой справедливы следующие утверждения:

- Корректность – должно быть конечное множество решений.
- Устойчивость метода – малому изменению исходных данных соответствует относительно малое изменение результата.
- Сходимость метода (только для итерационных методов) – применяемый метод сходится к решению за конечное число итераций.

1.6 Обозначения матриц

Пусть имеется квадратная матрица $\boldsymbol{A}$ размерностью 5×5:

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{bmatrix}$$

Существуют следующие разновидности матриц (примеры даны для размерности 5×5):

1.6.0.1 Единичная матрица

$$\boldsymbol{E} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

1.6.0.2 Диагональная матрица

$$\boldsymbol{D} = \begin{bmatrix} a_{11} & 0 & 0 & 0 & 0 \\ 0 & a_{22} & 0 & 0 & 0 \\ 0 & 0 & a_{33} & 0 & 0 \\ 0 & 0 & 0 & a_{44} & 0 \\ 0 & 0 & 0 & 0 & a_{55} \end{bmatrix}$$

1.6.0.3 Верхняя треугольная матрица

$$\boldsymbol{TU} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ 0 & a_{22} & a_{23} & a_{24} & a_{25} \\ 0 & 0 & a_{33} & a_{34} & a_{35} \\ 0 & 0 & 0 & a_{44} & a_{45} \\ 0 & 0 & 0 & 0 & a_{55} \end{bmatrix}$$

1.6.0.4 Нижняя треугольная матрица

$$\boldsymbol{TL} = \begin{bmatrix} a_{11} & 0 & 0 & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 & 0 \\ a_{31} & a_{32} & a_{33} & 0 & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} & 0 \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{bmatrix}$$

1.6.0.5 Трёхдиагональная матрица матрица

$$\boldsymbol{TD} = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} \\ 0 & 0 & 0 & a_{54} & a_{55} \end{bmatrix}$$

2 Методы решения систем линейных алгебраических уравнений (СЛАУ)

$$\boldsymbol{AX} = \boldsymbol{B} \tag{2.1}$$

где: $\boldsymbol{A}$ – матрица коэффициентов;
$\boldsymbol{X}$ – вектор неизвестных;
$\boldsymbol{B}$ – вектор правой части.

$$\begin{cases} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2 \\ \vdots & + & \vdots & + & \ddots & + & \vdots & = & \vdots \\ a_{m1}x_1 & + & a_{m2}x_2 & + & \cdots & + & a_{mn}x_n & = & b_n \end{cases}$$

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \quad \boldsymbol{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} \quad \boldsymbol{B} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

Необходимое условие существования единственного решения – число уравнений равняется числу условий. Другими словами матрица $\boldsymbol{A}$ должна быть квадратной.

Достаточное условие – ни одно уравнение линейно не связано с остальными уравнениями. Данное условие выполняется если $\det(\boldsymbol{A}) \neq 0$ (определитель можно посчитать только у квадратной матрицы).

2.1 Прямые методы

2.1.1 Метод исключения Гаусса

Метод исключения Гаусса состоит из двух проходов:

1. Прямой ход – преобразование исходной матрицы $\boldsymbol{A}$ в матрицу вида $\boldsymbol{TU}$ с помощью линейных операций. Сложность $O(n^3)$.
2. Обратный ход – нахождение вектора неизвестных от x_m до x_1. Сложность $O(n^2)$.

Пример прямого хода для СЛАУ размера 3×3

$$\begin{cases} a_{11} \cdot x_1 + a_{12} \cdot x_2 + a_{13} \cdot x_3 = b_1 & (1) \\ a_{21} \cdot x_1 + a_{22} \cdot x_2 + a_{23} \cdot x_3 = b_2 & (2) \\ a_{31} \cdot x_1 + a_{32} \cdot x_2 + a_{33} \cdot x_3 = b_3 & (3) \end{cases}$$

$$(2') = (2) - (1) \cdot \frac{a_{21}}{a_{11}}$$

$$(3') = (3) - (1) \cdot \frac{a_{31}}{a_{11}}$$

$$\begin{cases} a_{11} \cdot x_1 + a_{12} \cdot x_2 + a_{13} \cdot x_3 = b_1 & (1) \\ 0 + a'_{22} \cdot x_2 + a'_{23} \cdot x_3 = b'_2 & (2') \\ 0 + a'_{32} \cdot x_2 + a'_{33} \cdot x_3 = b'_3 & (3') \end{cases}$$

$$(3'') = (3') - (2') \cdot \frac{a_{32}}{a_{22}}$$

$$\begin{cases} a_{11} \cdot x_1 + a_{12} \cdot x_2 + a_{13} \cdot x_3 = b_1 & (1) \\ 0 + a'_{22} \cdot x_2 + a'_{23} \cdot x_3 = b'_2 & (2') \\ 0 + 0 + a''_{33} \cdot x_3 = b''_3 & (3'') \end{cases}$$

Без модификаций данный метод **не безопасен** из-за возможного деления на ноль, или число близкое к нулю.

2.1.1.1 Выбор ведущего элемента

Для повышения точности, а главное, для избежания деления на ноль, при прямом проходе перед линейным преобразованием, выбирается наибольший по модулю элемент в столбце – **ведущий элемент**. Текущая строка меняется местами со строкой с ведущим элементом. Таким образом не только предотвращается деление на ноль, но и увеличивается точность вычислений.

2.1.1.2 Нахождение определителя матрицы $\det(\boldsymbol{A})$

Благодаря тому, что после прямого хода данного метода образуется матрица вида $\boldsymbol{TU}$. Определитель матриц данного вида вычисляется простым произведением элементов на главной диагонали.

$$\det(\boldsymbol{A}) = \det(\boldsymbol{A_{TU}}) = \det(\boldsymbol{A_{TL}}) = \prod_{i=1}^{m} x_{ii} \tag{2.2}$$

2.1.2 Метод Краута-Холецки

Также известен как **метод $\boldsymbol{LU}$-разложения**. Данный метод требует хорошо обусловленную матрицу, т.е. элементы матрицы должны быть одинаковых порядков (влияет на точность вычислений), а также элементы матрицы не должны равняться нулю.

Любую невырожденную матрицу (т.е. матрицу у которой определитель не равен нулю) можно представить в виде произведения двух матриц:

$$\forall \boldsymbol{A} : \det(\boldsymbol{A}) \neq 0 \Rightarrow \boldsymbol{A} = \boldsymbol{L} \cdot \boldsymbol{U} \tag{2.3}$$

где: $\boldsymbol{L}$ – нижняя треугольная матрица;
$\boldsymbol{U}$ – верхняя треугольная матрица, у которой на главной диагонали одни единицы;

2.1.2.1 LU-разложение

Пусть a, l и u элементы матриц $\boldsymbol{A}$, $\boldsymbol{L}$ и $\boldsymbol{U}$ соответственно, тогда

$$l_{ij} = a_{ij} - \sum_{k=1}^{j-1} l_{ik} u_{kj} \tag{2.4}$$

$$u_{ij} = \left(a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj} \right) \Big/ l_{ii} \tag{2.5}$$

Таким образом:

$$l_{i1} = a_{i1} \tag{2.6}$$

$$u_{1j} = a_{1j} / a_{11} \tag{2.7}$$

После $\boldsymbol{LU}$-разложения исходное уравнение можно преобразовать следующим образом:

$$\boldsymbol{AX} = \boldsymbol{B} \tag{2.8}$$

$$\boldsymbol{L} \cdot \underbrace{\boldsymbol{U} \cdot \boldsymbol{X}}_{\boldsymbol{Y}} = \boldsymbol{B} \tag{2.9}$$

$$\begin{cases} \boldsymbol{L} \cdot \boldsymbol{Y} & = \boldsymbol{B} \\ \boldsymbol{U} \cdot \boldsymbol{X} & = \boldsymbol{Y} \end{cases} \tag{2.10}$$

Получившиеся система линейных матричных уравнений решается двумя обратными проходами Гаусса.

$$\boldsymbol{L} \cdot \boldsymbol{Y} = \boldsymbol{B}$$

$$\begin{cases} l_{11}y_1 + 0 + \cdots + 0 = b_1 \Rightarrow y_1 = b_1/l_{11} \\ l_{21}y_1 + l_{22}y_2 + \cdots + 0 = b_2 \Rightarrow y_2 = \cdots \\ \vdots + \vdots + \ddots + \vdots = \vdots \Rightarrow \vdots = \cdots \\ l_{m1}y_1 + l_{m2}y_2 + \cdots + l_{mn}y_n = b_m \Rightarrow y_n = \cdots \end{cases}$$

$$\boldsymbol{U} \cdot \boldsymbol{X} = \boldsymbol{Y}$$

$$\begin{cases} x_1 + u_{12}x_2 + \cdots + u_{1n}x_n = y_1 \Rightarrow x_1 = \cdots \\ 0 + x_2 + \cdots + u_{2n}x_n = y_2 \Rightarrow x_2 = \cdots \\ \vdots + \vdots + \ddots + \vdots = \vdots \Rightarrow \vdots = \cdots \\ 0 + 0 + \cdots + x_n = y_m \Rightarrow x_n = y_m/u_{mn} \end{cases}$$

После $\boldsymbol{LU}$-разложения вычислительная сложность решения системы линейных матричных уравнений равняется двум обратным проходам Гаусса, т.е. $O(n^2)$. Таким образом данный метод быстрее метода Гаусса **только**, если необходим многократный расчёт $\boldsymbol{X}$ с одной и той-же $\boldsymbol{A}$ и разными $\boldsymbol{B}$ (иначе требуется производить повторное $\boldsymbol{LU}$-разложение, что медленнее прямого хода Гаусса).

2.1.3 Метод прогонки

Метод прогонки применим только для трёхдиагональных матриц ($\boldsymbol{TD}$) (т.е. для матриц у которых в двух строках и столбцах два ненулевых элемента, а во всех остальных – три).

$$\begin{cases} a_{11}x_1 + a_{12}x_2 = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2 \\ a_{32}x_2 + a_{33}x_3 + \ldots = b_3 \\ \vdots + \ddots = \vdots \\ a_{n,n-1}x_{n-1} + a_{nn}x_n = b_n \end{cases}$$

Из-за того что в первом уравнении всего два неизвестных x_1 и x_2, то одно можно выразить через другое и подставить в следующее уравнение. Подставив в следующее уравнение, в нём останется также два неизвестных. Так можно повторять для последующих уравнений, вплоть до последнего, в котором останется одно неизвестное, которое уже можно вычислить. Затем обратным ходом находятся остальные неизвестные.

$$
\begin{aligned}
x_1 =& \frac{b_1 - a_{12}x_2}{a_{11}} = \beta_1 - \alpha_1 x_2 \\
& a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2 \\
& a_{21}(\beta_1 - \alpha_1 x_2) + a_{22}x_2 + a_{23}x_3 = b_2 \\
& (a_{22} - a_{21}\alpha_1)x_2 = -a_{23}x_3 + b_2 - a_{21}\beta_1 \\
x_2 =& \frac{-a_{23}x_3 + b_2 - a_{21}\beta_1}{a_{22} - a_{21}\alpha_1} = \beta_2 - \alpha_2 x_3 \\
& \text{и т.д.}
\end{aligned}
$$

Таким образом получается следующий алгоритм:

1. Нахождение α_1, β_1

$$
\begin{aligned}
\alpha_1 &= \frac{a_{12}}{a_{11}} \\
\beta_1 &= \frac{b_1}{a_{11}}
\end{aligned}
\tag{2.11}
$$

2. Нахождение $\alpha_i, \beta_i \ \forall i \in \overline{2, n}$

$$
\alpha_k = \frac{a_{k,k+1}}{a_{kk} - a_{k,k-1} \cdot \alpha_{k-1}} \tag{2.12}
$$

$$
\beta_k = \frac{b_k - a_{k,k-1} \cdot \beta_{k-1}}{a_{kk} - a_{k,k-1} \cdot \alpha_{k-1}} \tag{2.13}
$$

3. $x_n = \beta_n$
4. «Обратный ход Гаусса»

$$
x_i = \beta_i - \alpha_i x_{i+1} \tag{2.14}
$$

2.2 Число обусловленности матрицы

Для сравнения многомерных величин существует понятие «мера сравнения».

2.2.1 Мера сравнения для векторов

Мерой сравнения для вектора является **норма вектора**.

Определение 2.2.1.1:

Норма вектора $\|\boldsymbol{X}\|$ – число, которое характеризует уровень всех элементов.

Евклидова норма $\|\boldsymbol{X}\| = \sqrt{\sum_{i=0}^{n} x_i^2}$

Манхэттенская норма $\|\boldsymbol{X}\| = \sum_{i=0}^{n} |x_i|$

2.2.2 Мера сравнения для матриц

Матрицу можно рассматривать как вектор векторов, поэтому принято считать нормой матрицы – максимальную норму её столбцов.

$$
\|\boldsymbol{A}\| = \max \left\| a_j \right\| \tag{2.15}
$$

2.2.2.1 Теоретическое определение

Таким образом, имея меру сравнения матриц, определяется **число обусловленности матрицы**:

$$
\operatorname{cond}(\boldsymbol{A}) = \|\boldsymbol{A}\| \cdot \left\| \boldsymbol{A}^{-1} \right\| \tag{2.16}
$$

Согласно определению число обусловленности принимает следующие значения:

$$
\operatorname{cond}(\boldsymbol{A}) \in [1; +\infty) \tag{2.17}
$$

2.2.2.2 Экспериментальное определение

Для СЛАУ в форме $\boldsymbol{AX} = \boldsymbol{B}$ число обусловленности матрицы $\boldsymbol{A}$ определяется следующим образом:

$$\frac{\|\Delta \boldsymbol{X}\|}{\|\boldsymbol{X}\|} \leq \operatorname{cond}(\boldsymbol{A}) \cdot \frac{\|\Delta \boldsymbol{B}\|}{\|\boldsymbol{B}\|} \tag{2.18}$$

либо

$$\operatorname{cond}(\boldsymbol{A}) \geq \frac{\|\Delta \boldsymbol{X}\|}{\|\boldsymbol{X}\|} \cdot \frac{\|\boldsymbol{B}\|}{\|\Delta \boldsymbol{B}\|} \tag{2.19}$$

где:

$\boldsymbol{B}$	Исходный вектор правой части;
$\boldsymbol{X}$	Найденное решение для данной СЛАУ с исходной $\boldsymbol{B}$.
$\Delta \boldsymbol{B}$	Вектор «искусственно» добавленной ошибкой для правой части. Обычно выбирается один наибольший по модулю элемент в $\boldsymbol{B}$, и в $\Delta \boldsymbol{B}$ как принято оставляют 1% от исходного значения. Т.е. только один элемент не равен нулю, а все остальные – нули.
$\Delta \boldsymbol{X}$	Разница между $\boldsymbol{X}$ и найденным решением для СЛАУ с $\boldsymbol{B} + \Delta \boldsymbol{B}$;
$\frac{\|\Delta \boldsymbol{X}\|}{\|\boldsymbol{X}\|}$	Относительная норма ошибки, другими словами: “отношение нормы ошибки решения к норме исходного решения”;
$\frac{\|\Delta \boldsymbol{B}\|}{\|\boldsymbol{B}\|}$	Относительная норма изменения правой части.

2.2.2.3 Интерпретация значений числа обусловленности матрицы

В лучшем случае $\operatorname{cond}(\boldsymbol{A}) \to 1$, тогда относительно небольшое изменение исходных данных даёт относительно небольшое изменение решения.

В худшем случае $\operatorname{cond}(\boldsymbol{A}) \to \infty$, тогда относительно небольшое изменение исходных данных приводит к значительному изменению результата.

$$\begin{gathered} \boldsymbol{D}, \boldsymbol{E} : \operatorname{cond}(\boldsymbol{A}) = 1 \\ \det(\boldsymbol{A}) = 0 : \operatorname{cond}(\boldsymbol{A}) \to \infty \end{gathered} \tag{2.20}$$

Таким образом принято различать следующие значения:

$1 \leq \operatorname{cond}(\boldsymbol{A}) < 10$	хорошо обусловленная
$10 \leq \operatorname{cond}(\boldsymbol{A}) < 1000$	зависит от требуемой точности
$1000 \leq \operatorname{cond}(\boldsymbol{A}) < \infty$	плохо обусловленная

2.3 Итерационные методы решения СЛАУ

Общий алгоритм для итерационных методов

1. Приведение к итерационному виду, т.е. виду, удобному для итераций;

$$\boldsymbol{AX} = \boldsymbol{B} \qquad \to \qquad \boldsymbol{X} = \boldsymbol{\alpha X} + \boldsymbol{\beta}$$

2. Выбор начального приближения $\boldsymbol{X}_{(0)}$ и требуемой точности ϵ. Если ничего неизвестно, то $\boldsymbol{X}_{(0)}$ можно выбрать 0 или 1.

3. Итерационный процесс следующего вида:

$$\boldsymbol{X}_{(k+1)} = \boldsymbol{\alpha X}_{(k)} + \boldsymbol{\beta}$$

Итерации продолжаются до тех пор пока не достигнута требуемая точность:

$$\max \left| x_{i_{(k+1)}} - x_{i_{(k)}} \right| \leq \epsilon \tag{2.21}$$

Примечание

В неравенстве (2.21) за требуемую точность берётся «текущая» точность (скорость изменения решения). Это делается за отсутствием эталонного x_0.

2.3.1 Достаточное условие сходимости итерационных методов

Каждый диагональный элемент по модулю должен быть больше или равен сумме всех остальных элементов в строке и столбце.

$$|a_{ii}| \geq \sum_{i \neq j} |a_{ij}| \quad \forall i \in \overline{1,n} \tag{2.22}$$

Данное условие является достаточным, но **не** является необходимым! (т.е. при его не выполнении итерационный метод может сходится)

2.3.2 Метод Якоби

Данный метод так-же известен как метод «простых итераций». Суть метода заключается в выражении неизвестного x_i из i-ого уравнения, для всех уравнений:

$$\begin{cases} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2 \\ \vdots & + & \vdots & + & \ddots & + & \vdots & = & \vdots \\ a_{n1}x_1 & + & a_{n2}x_2 & + & \cdots & + & a_{nn}x_n & = & b_n \end{cases}$$

$$\begin{aligned} x_1 &= \frac{b_1}{a_{11}} - \frac{a_{12}}{a_{11}}x_2 - \cdots - \frac{a_{1n}}{a_{11}}x_n \\ x_2 &= \frac{b_2}{a_{22}} - \frac{a_{21}}{a_{22}}x_1 - \cdots - \frac{a_{2n}}{a_{22}}x_n \\ &\vdots \\ x_n &= \underbrace{\frac{b_n}{a_{nn}}}_{\beta} \underbrace{- \frac{a_{n1}}{a_{nn}}x_1 - \cdots - \frac{a_{n,n-1}}{a_{nn}}x_{n-1}}_{\alpha} \end{aligned}$$

2.3.3 Метод Гаусса-Зейделя

В методе Якоби для вычисления x_k на i-ом шаге требуются значения $x_1, ..., x_k$ из предыдущего шага. Данный метод «ускоряет» метод Якоби таким образом, что на i-ом шаге очередной итерации, для расчёта x_k вместо значений $x_1, ..., x_k$ из предыдущей итерации берутся значения полученные при текущей итерации.

2.3.4 Метод последовательной нижней релаксации

Метод релаксаций используется для обеспечения сходимости. Данный метод использует предыдущие методы для нахождения $\boldsymbol{X}$, а в итерационном процессе используется следующее выражение:

$$\boldsymbol{X}_{(k+1)} = \boldsymbol{X}_{(k)} + \omega(\tilde{\boldsymbol{X}}_{(k+1)} - \boldsymbol{X}_{(k)}) \tag{2.23}$$

где: $\omega \in (0; 1]$ – коэффициент релаксации;
$\tilde{\boldsymbol{X}}_{(k+1)}$ – значения полученные с помощью других методов;

Стоит заметить, что

$$\omega = 1 \Rightarrow \boldsymbol{X}_{(k+1)} = \tilde{\boldsymbol{X}}_{(k+1)} \tag{2.24}$$

Таким образом при обнаружении расходимости значение ω понижают.

3 Приближение функций заданных в табличной форме

- Интерполяция / Экстраполяция – график функции проходит через все заданные точки;
- Аппроксимация – график функции проходит наиболее близко ко всем заданным точкам.

3.1 Глобальная интерполяция многочленами

Интерполяция – нахождение промежуточных значений функции по имеющемуся дискретному набору известных значений.

При интерполяции многочленами, интерполирующая функция представляется в виде линейной суперпозиции функций, называемых базисными:

$$\phi(x) = \sum_{m=0}^{k} a_m \cdot f_m(x) \tag{3.1}$$

где: k – порядок интерполяции;
f_m – базисы интерполяции;
a_m – неизвестные коэффициенты.

При интерполяции график функции ϕ проходит через все заданные точки ($\phi(x_i) = y_i$), т.е. все точки связаны некоторым законом.

3.1.1 Степенной базис

$$\begin{gathered} f_m(x) = a^m \\ \phi(x) = a_0 + a_1 x^1 + \cdots + a_k x^k \end{gathered} \tag{3.2}$$

$k = 0$ – константа;

$k = 1$ – прямая общего положения;

$k = 2$ – парабола;

$k = 3$ – кубическая парабола;

$\vdots$

Согласно следствию «Основной теоремы алгебры»: “Любой многочлен степени n имеет не более n корней, с учётом их кратности”. Если задана $n + 1$ точка, то достаточно полинома порядка n для проведения через них единственной кривой. Данный факт является общим случаем для известной из геометрии аксиомы: “через 2 точки можно провести единственную прямую”.

3.1.1.1 Метод заданных точек

В методе заданных точек составляется СЛАУ, в которой матрица неизвестных коэффициентов состоит из значений x_i заданных точек, а вектор правой части – значений y_i:

$$n+1 \left\{ \begin{array}{ccccccccc} a_0 & + & a_1 x_0^1 & + & \cdots & + & a_n x_0^n & = & y_0 \\ a_0 & + & a_1 x_1^1 & + & \cdots & + & a_n x_1^n & = & y_1 \\ \vdots & + & \vdots & + & \ddots & + & \vdots & = & \vdots \\ a_0 & + & a_1 x_n^1 & + & \cdots & + & a_n x_n^n & = & y_n \end{array} \right.$$

где: a_k – неизвестные коэффициенты;
x_k и y_k – табличные значения.

Таким образом в матричном виде матрица коэффициентов $\boldsymbol{A}$, вектор неизвестных $\boldsymbol{X}$ и вектор правой части $\boldsymbol{B}$ будут выглядеть следующим образом:

$$\boldsymbol{A} = \begin{bmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \cdots & x_n^n \end{bmatrix} \quad \boldsymbol{X} = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} \quad \boldsymbol{B} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Представленная матрица $\boldsymbol{A}$ называется **матрицей Вандермонда**, у которой при больших n $\operatorname{cond}(\boldsymbol{A}) \to \infty$. Поэтому метод заданных точек имеет ограниченное применение (не рекомендуется использовать при $n \geq 10$).

Пример

Пусть заданы следующие точки:

x	2	3	5
y	1	0	4

$$\begin{cases} a_0 + 2a_1 + 4a_2 & = 1 \\ a_0 + 3a_1 + 9a_2 & = 0 \\ a_0 + 5a_1 + 25a_2 & = 4 \end{cases}$$

При решении данной системы получится решение $\{9; -6; 1\}$. Следовательно, искомый полином является:

$$\phi(x) = x^2 - 6x + 9$$

3.1.1.2 Интерполяция Лагранжа

Данный метод интерполяции не требует решения СЛАУ. Интерполирующий полином задаётся с помощью следующей линейной суперпозиции:

$$L_n(x) = \sum_{k=0}^{n} y_k l_k(x) = y_0 l_0(x) + \cdots + y_n l_n(x) \tag{3.3}$$

где: $l_k(x)$ – локальные полиномы Лагранжа;
y_k – табличные значения y.

Из-за того что в правой части имеются y_k, то для полного соответствия полинома $L_n(x)$ табличным значениям ($L_n(x_k) = y_k$), требуется, чтобы для каждого i-ого табличного значения, i-ый локальный полином Лагранжа $l_i(x_i)$ был равен единице, а все остальные локальные полиномы – нулю. Данное достигается следующим определением $l_i(x_j)$:

$$l_i(x_j) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \tag{3.4}$$

Таким образом можно обобщить для всех остальных значений x:

$$l_i(x) = \frac{(x - x_0) \cdots (x - x_{i-1}) \cdot (x - x_{i+1}) \cdots (x - x_n)}{(x_i - x_0) \cdots (x_i - x_{i-1}) \cdot (x_i - x_{i+1}) \cdots (x_i - x_n)} \tag{3.5}$$

Следует заметить, что в знаменателе не может получится ноль из-за того, что по определению функции, 2 одинаковых x не может быть.

Данное определение $l_i(x)$ включает в себя первое, т.к.:

- если $x = x_i$, то числитель будет равен знаменателю, следовательно $l_i(x_i) = 1$;
- если $x = x_j, i \neq j$, то какой-то из множителей в числителе будет равняться нулю, следовательно $l_i(x_j) = 0$.

Пример

В качестве примера будет использоваться заданные x и y из предыдущего примера.

x	2	3	5
y	1	0	4

$$\begin{aligned} L_2(x) = & 1 \cdot \frac{(x-3)(x-5)}{(2-3)(2-5)} \\ & + 0 \cdot \frac{(x-2)(x-5)}{(3-2)(3-5)} \\ & + 4 \cdot \frac{(x-2)(x-3)}{(5-2)(5-3)} \\ = & x^2 - 6x + 9 \end{aligned}$$

3.1.1.3 Интерполяция Ньютона

$$N_n(x) = \sum_{i=0}^{n} \left(a_i \cdot \prod_{m=0}^{i-1} (x - x_m) \right) \\ = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n \cdot \prod_{m=0}^{n-1} (x - x_m) \tag{3.6}$$

где

$$\begin{aligned} a_0 &= y_0 \\ a_1 &= \Delta y_0 = \frac{y_1 - y_0}{x_1 - x_0} \\ a_2 &= \Delta^2 y_0 = \frac{\Delta y_1 - \Delta y_0}{x_2 - x_0} \\ &\vdots \\ a_n &= \Delta^n y_0 = \frac{\Delta^{n-1} y_1 - \Delta^{n-1} y_0}{x_n - x_0} \end{aligned} \tag{3.7}$$

Таким образом данный метод не требует полного пересчёта при добавлении точки в конец таблицы.

Пример

В качестве примера будет использоваться заданные x и y из предыдущих примеров.

x	2	3	5
y	1	0	4

Рассчитываем Δy и Δy^2:

x	y	Δy	$\Delta^2 y$
2	1	–	–
3	0	–	–
5	4	–	–

$\rightarrow$

x	y	Δy	$\Delta^2 y$
2	1	-1	–
3	0	–	–
5	4	–	–

$\rightarrow$

x	y	Δy	$\Delta^2 y$
2	1	-1	–
3	0	2	–
5	4	–	–

$\rightarrow$

x	y	Δy	$\Delta^2 y$
2	1	-1	1
3	0	2	–
5	4	–	–

... подставляем ...

$$N_2(x) = 1 - 1(x - 2) + 1(x - 2)(x - 3) = x^2 - 6x + 9$$

3.2 Локальная интерполяция

Локальная интерполяция является частным случаем глобальной интерполяции, когда вместо всех точек, за раз рассчитывается полином с ограниченным числом точек.

Локальную интерполяцию можно описать как глобальную интерполяцию применяемую за раз только к m точкам в т.н. «окне». Причём, «окно», как правило, сдвигается на 1 точку за раз, пока в окне не окажется последняя точка. Результатом интерполяции являются полученные интерполяции отдельных окон, “усреднённые” для каждой точки.

Метод сдвига окна, и метод “усреднения” зависит от конкретно применяемого метода.

3.2.1 Метод интерполяции кубическими сплайнами

Идея интерполяции с помощью кубических сплайнов заключается в вычислении функций – сплайнов $S_i(x)$ между каждыми двумя соседними точками, таким образом чтобы в местах соединения (заданных x) не наблюдалось сильных изгибов и разрывов. Последнее достигается с помощью задания зависимостей (условий интерполяции) между двумя сплайнами.

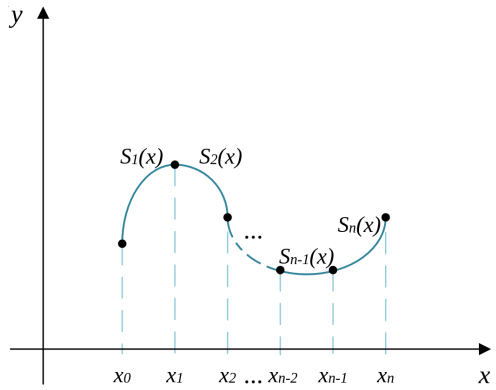
Пусть задана $n + 1$ точка:

x	x_0	x_1	$\cdots$	x_{n-1}	x_n
y	y_0	y_1	$\cdots$	y_{n-1}	y_n

тогда между ними можно построить n сплайнов.

Согласно определению данного метода, функция задающая сплайн должна быть (полиномом) третьего порядка:

$$S_i(x) = a_i + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2 + d_i(x - x_{i-1})^3 \tag{3.8}$$



где: $i \in \overline{1,n}$ – номер сплайна;
$S_i(x)$ – функция от x для i-ого сплайна;
x_{i-1} – начальная (левая) точка для i-ого сплайна;
a_i, b_i, c_i, d_i – неизвестные коэффициенты.

Таким образом данный метод интерполяции является локальным (кусочным), и не зависит от количества точек. Порядок $S_i(x)$ всегда будет равен трём.

При заданной $n+1$ точки, для интерполяции всей функции целиком, необходимо найти 4 коэффициента (a_i, b_i, c_i, d_i) для каждого сплайна $S_i(x)$. Всего таких сплайнов – n. Поэтому необходимо найти всего $4n$ неизвестных.

Обозначим расстояние между двумя соседними точками как h_i:

$$h_i = x_i - x_{i-1} \tag{3.9}$$

Для нахождения всех $4n$ неизвестных определим условия интерполяции:

Условия интерполяции

1. Каждый сплайн проходит через «левую» точку:

$$\begin{aligned} S_i(x_{i-1}) &= y_{i-1} \\ S_i(x_{i-1}) &\stackrel{\text{def}}{=} a_i + b_i(x_{i-1} - x_{i-1}) + c_i(x_{i-1} - x_{i-1})^2 + d_i(x_{i-1} - x_{i-1})^3 \\ &= a_i \\ &\Downarrow \\ y_{i-1} &= a_i \end{aligned} \tag{3.10}$$

   Данных уравнений можно составить n штук (для каждого сплайна).

2. Каждый сплайн проходит через «правую» точку:

$$\begin{aligned} S_i(x_i) &= y_i \\ S_i(x_i) &\stackrel{\text{def}}{=} a_i + b_i(x_i - x_{i-1}) + c_i(x_i - x_{i-1})^2 + d_i(x_i - x_{i-1})^3 \\ &= a_i + b_i h_i + c_i h_i^2 + d_i h_i^3 \\ &\Downarrow \\ y_i &= a_i + b_i h_i + c_i h_i^2 + d_i h_i^3 \end{aligned} \tag{3.11}$$

   Данных уравнений можно составить n штук (для каждого сплайна).

Первые два уравнения дают только условия прохождения через заданные точки, и они не задают, как будет проходить кривая через эти точки. Поэтому в заданных точках, без дополнительных условий, могут образовываться «негладкие» изгибы, или другими словами, производные двух сплайнов в одной точке будут различаться (разрыв первого рода).

Математическое определение гладкой прямой – “бесконечно дифференцируемая функция во всех точках”. Поэтому, чтобы сплайн был гладкий, и чтобы найти оставшиеся $2n$ неизвестных, вводятся еще два условия относительно производных S_i:

3. У соседних сплайнов, в точке соединения, производные первого порядка должны равняться между собой:

$$\begin{aligned} S_i'(x_i) &= S_{i+1}'(x_i) \\ S_i'(x) &= 0 + b_i + 2c_i(x - x_{i-1}) + 3d_i(x - x_{i-1})^2 \\ S_i'(x_i) &= 0 + b_i + 2c_i(x_i - x_{i-1}) + 3d_i(x_i - x_{i-1})^2 \\ &= b_i + 2c_i h_i + 3d_i h_i^2 \\ S_{i+1}'(x_i) &= 0 + b_{i+1} + 2c_{i+1}(x_i - x_i) + 3d_{i+1}(x_i - x_i)^2 \\ &= b_{i+1} \\ &\Downarrow \\ b_i + 2c_i h_i + 3d_i h_i^2 &= b_{i+1} \end{aligned} \tag{3.12}$$

4. У соседних сплайнов, в точке соединения, производные второго порядка должны равняться между собой:

$$\begin{aligned} S_i''(x_i) &= S_{i+1}''(x_i) \\ S_i''(x) &= 0 + 2c_i + 6d_i(x - x_{i-1}) \\ S_i''(x_i) &= 0 + 2c_i + 6d_i(x_i - x_{i-1}) \\ &= 2c_i + 6d_i h_i \\ S_{i+1}''(x_i) &= 0 + 2c_{i+1} + 6d_{i+1}(x_i - x_i) \\ &= 2c_{i+1} \\ &\Downarrow \\ 2c_i + 6d_i h_i &= 2c_{i+1} \quad | : 2 \\ c_i + 3d_i h_i &= c_{i+1} \end{aligned} \tag{3.13}$$

Данные требования дают еще $2 \cdot (n-1)$ уравнения, таким образом не хватает два уравнения для нахождения всех $4n$ неизвестных. Поэтому существует две разновидности сплайнов: с жёстким и со свободным закреплением концевых точек.

С жёстким закреплением концевых точек

Данный метод применим только если известна постановка исходной задачи, из которой можно определить первые или вторые производные для концевых точек.

Со свободным закреплением концевых точек

При «свободном» закреплении предполагается, что за пределами заданных точек функция ведёт себя линейно.

Таким образом задаётся еще одно условие, из которого получаются два недостающих уравнения:

$$\begin{aligned} S_1''(x_0) &= 0 \\ S_1''(x_0) &= 2c_1 + 6d_1(x_0 - x_0) \\ &= 2c_1 \\ &\Downarrow \\ 2c_1 &= 0 \end{aligned} \qquad \text{и} \qquad \begin{aligned} S_n''(x_n) &= 0 \\ S_n''(x_n) &= 2c_n + 6d_n(x_n - x_{n-1}) \\ &= 2c_n + 6d_n h_n \\ &\Downarrow \\ 2c_n + 6d_n h_n &= 0 \end{aligned}$$

Формирование алгоритма

1. Подставляем равенство (3.10) в уравнение (3.11)

$$\begin{aligned} y_{i-1} + b_i h_i + c_i h_i^2 + d_i h_i^3 &= y_i \\ &\Downarrow \\ b_i h_i + c_i h_i^2 + d_i h_i^3 &= y_i - y_{i-1} \end{aligned} \tag{3.14}$$

2. Из равенства (3.13) полученного из второй производной S_i можно выразить d_i:

$$d_i = \frac{c_{i+1} - c_i}{3h_i} \tag{3.15}$$

3. В равенство (3.12) полученное из первой производной S_i подставляется d_i (3.15):

$$\begin{gathered} b_i + 2c_i h_i + 3\frac{c_{i+1} - c_i}{3h_i} h_i^2 = b_{i+1} \\ \Downarrow \\ b_i + 2c_i h_i + (c_{i+1} - c_i) h_i = b_{i+1} \\ \Downarrow \\ b_i + (c_{i+1} + c_i) h_i = b_{i+1} \end{gathered} \tag{3.16}$$

4. В равенство (3.14) подставляется d_i (3.15):

$$\begin{gathered} b_i h_i + c_i h_i^2 + \frac{c_{i+1} - c_i}{3h_i} h_i^3 = y_i - y_{i-1} \\ \Downarrow \\ b_i h_i + c_i h_i^2 + \frac{1}{3}(c_{i+1} - c_i) h_i^2 = y_i - y_{i-1} \\ \Downarrow \\ b_i h_i + \frac{1}{3}(c_{i+1} + 2c_i) h_i^2 = y_i - y_{i-1} \end{gathered} \tag{3.17}$$

5. Из только что полученного равенства (3.17) выражается b_i:

$$\begin{aligned} b_i &= \frac{y_i - y_{i-1} - \frac{1}{3}(c_{i+1} + 2c_i) h_i^2}{h_i} \\ &= \frac{y_i - y_{i-1}}{h_i} - \frac{1}{3}(c_{i+1} + 2c_i) h_i \end{aligned} \tag{3.18}$$

6. Наконец, полученное b_i (3.18) подставляется в левую и правую часть равенства (3.16), с учётом изменения индекса в правой части:

$$\left[\frac{y_i - y_{i-1}}{h_i} - \frac{1}{3}(c_{i+1} + 2c_i) h_i\right] + (c_{i+1} + c_i) h_i = \left[\frac{y_{i+1} - y_i}{h_{i+1}} - \frac{1}{3}(c_{i+2} + 2c_{i+1}) h_{i+1}\right]$$

$$-\frac{1}{3}(c_{i+1} + 2c_i) h_i + (c_{i+1} + c_i) h_i + \frac{1}{3}(c_{i+2} + 2c_{i+1}) h_{i+1} = \frac{y_{i+1} - y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i} \quad \Big| \cdot 3$$

$$-(c_{i+1} + 2c_i) h_i + 3(c_{i+1} + c_i) h_i + (c_{i+2} + 2c_{i+1}) h_{i+1} = 3 \cdot \left(\frac{y_{i+1} - y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i}\right)$$

$$-h_i c_{i+1} - 2h_i c_i + 3h_i c_{i+1} + 3h_i c_i + h_{i+1} c_{i+2} + 2h_{i+1} c_{i+1} = 3 \cdot \left(\frac{y_{i+1} - y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i}\right)$$

$$h_i c_i - h_i c_{i+1} + 3h_i c_{i+1} + 2h_{i+1} c_{i+1} + h_{i+1} c_{i+2} = 3 \cdot \left(\frac{y_{i+1} - y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i}\right)$$

$$h_i c_i + 2h_i c_{i+1} + 2h_{i+1} c_{i+1} + h_{i+1} c_{i+2} = 3 \cdot \left(\frac{y_{i+1} - y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i}\right)$$

$$h_i c_i + 2(h_i + h_{i+1}) c_{i+1} + h_{i+1} c_{i+2} = 3 \cdot \left(\frac{y_{i+1} - y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i}\right)$$

Переиндексировав на единицу меньше (от всех i отняв 1), получается:

$$h_{i-1} c_{i-1} + 2(h_{i-1} + h_i) c_i + h_i c_{i+1} = 3 \cdot \left(\frac{y_i - y_{i-1}}{h_i} - \frac{y_{i-1} - y_{i-2}}{h_{i-1}}\right) \tag{3.19}$$

Можно заметить, что в полученном уравнении, неизвестные переменные: c_{i-1}, c_i и c_{i+1}. Таким образом, используя данное уравнение, как будет показано далее, можно составить трёхдиагональную СЛАУ, которая быстро решается методом прогонки.

Алгоритм

1. $a_i = y_{i-1}$;
2. Составляется СЛАУ для c_i, используя уравнение (3.19):

$$\begin{aligned}
&\text{при } i=2 \to 2(h_1+h_2)c_2 + && h_2c_3 && && = 3\left(\frac{y_2-y_1}{h_2} - \frac{y_1-y_0}{h_1}\right)\\
&\text{при } i=3 \to && h_2c_2 + 2(h_2+h_3)c_3 + && h_3c_4 && = 3\left(\frac{y_3-y_2}{h_3} - \frac{y_2-y_1}{h_2}\right)\\
&\text{при } i=4 \to && && h_3c_3 + 2(h_3+h_4)c_4 + && h_4c_5 = 3\left(\frac{y_4-y_3}{h_4} - \frac{y_3-y_2}{h_3}\right)\\
&\text{при } i=5 \to && && && h_4c_4 + 2(h_4+h_5)c_5 + h_5c_6 = 3\left(\frac{y_5-y_4}{h_5} - \frac{y_4-y_3}{h_4}\right)\\
&\quad\vdots
\end{aligned}$$

   Можно заметить, что получается СЛАУ относительно c_i с трёхдиагональной хорошо обусловленной матрицей коэффициентов, для которой подходит метод прогонки.

3. Вычисляются c_i:

$$\begin{bmatrix} \ddots & \ddots & & \\ \ddots & \ddots & \ddots & \\ & \ddots & \ddots & \ddots \\ & & \ddots & \ddots \end{bmatrix} \cdot \begin{bmatrix} c_2 \\ \vdots \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} b_2 \\ \vdots \\ \vdots \\ b_n \end{bmatrix}$$

4. Получив c_i, вычисляются d_i с помощью выражения (3.15);
5. Имея c_i и d_i, вычисляются последние неизвестные коэффициенты – b_i с помощью выражения (3.18).

На выходе получается таблица коэффициентов для каждого сплайна:

№ сплайна	a_i	b_i	c_i	d_i
1	y_0	b_1	0	d_1
2	y_1	b_2	c_2	d_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	y_{n-1}	b_n	c_n	d_n

3.3 Аппроксимация

Задачей аппроксимации является построение кривой, наиболее близко к заданным точкам. Для этого необходимо определить, что такое «близость к точкам». Самый тривиальный способ определения близости к точке – вычисление разности между значением на построенной кривой ($\varphi(x_i)$) и исходным значением в соответствующей точке ($y(x_i)$).

$$\Delta_i = \varphi(x_i) - y_i$$

Однако, значения Δ_i могут быть как положительными, так и отрицательными, по этой причине данный критерий не подходит в качестве критерия минимизации (в сумме Δ_i может быть 0). Следовательно, следует взять модуль:

$$\Delta_i = \left|\varphi(x_i) - y_i\right|$$

Данный критерий возможно использовать для минимизации. Но его сложно дифференцировать, что важно для нахождения экстремума – минимума.

По этой причине самый простой и распространённый критерий для минимизации – сумма квадратов разностей.

3.3.1 Метод наименьших квадратов (МНК)

$$\sum_{i=0}^{n} \big(\varphi_k(x_i) - y_i\big)^2 = \sum_{i=0}^{n} \Delta_i^2 \to \min \tag{3.20}$$

где: $\varphi_k(x)$ – функция аппроксимации порядка k.

В качестве функции аппроксимации используется линейная суперпозиция:

$$\varphi_k(x) = \sum_{m=0}^{k} a_m f_m(x) = a_0 f_0(x) + \cdots + a_k f_k(x) \tag{3.21}$$

где: f_m – система базисных функций;
k – порядок аппроксимации.

После выбора базиса задача сводится к нахождению коэффициентов.

$$S = \sum_{i=0}^{n} \Delta_i^2 = \sum_{i=0}^{n} \left[\left(\sum_{m=0}^{k} a_m f_m(x) \right) - y_i \right]^2 \to \min \tag{3.22}$$

$$= \sum_{i=0}^{n} \big(a_0 f_0(x_i) + \cdots + a_k f_k(x_i) - y_i\big)^2 \to \min \tag{3.23}$$

$$= S(\underbrace{a_0, ..., a_k}_{k+1}) \tag{3.24}$$

Можно заметить, что график функции S – парабола, ветви которой направленны вверх, а следовательно, в данном уравнении есть всего один экстремум, который и будет являться точкой минимума.

В данной задаче минимизации S зависит от $k+1$ неизвестного коэффициента, которые можно найти приравняв каждую частную производную от S к нулю (для нахождения минимума). При этом получается $k+1$ уравнение – СЛАУ, решением которого будут найденные коэффициенты для базисных функций аппроксимации.

$$k+1 \begin{cases} \dfrac{\partial S}{\partial a_0} = 2\sum\limits_{i=0}^{n} \Big[\big(a_0 f_0(x_i) + \cdots + a_k f_k(x_i) - y_i\big) \cdot f_{\boldsymbol{0}(x_i)} \Big] = 0 \\ \dfrac{\partial S}{\partial a_1} = 2\sum\limits_{i=0}^{n} \Big[\big(a_0 f_0(x_i) + \cdots + a_k f_k(x_i) - y_i\big) \cdot f_{\boldsymbol{1}(x_i)} \Big] = 0 \\ \quad \vdots \\ \dfrac{\partial S}{\partial a_k} = 2\sum\limits_{i=0}^{n} \Big[\big(a_0 f_0(x_i) + \cdots + a_k f_k(x_i) - y_i\big) \cdot f_{\boldsymbol{k}(x_i)} \Big] = 0 \end{cases} \tag{3.25}$$

$$\begin{cases} \boldsymbol{a_0} \sum\limits_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{f_0^2(x_i)} + a_1 \sum\limits_{i=0}^{n} f_1(x_i) f_0(x_i) + \cdots + a_k \sum\limits_{i=0}^{n} f_k(x_i) f_0(x_i) = \sum\limits_{i=0}^{n} y_i f_0(x_i) \\ a_0 \sum\limits_{i=0}^{n} f_0(x_i) f_1(x_i) + \boldsymbol{a_1} \sum\limits_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{f_1^2(x_i)} + \cdots + a_k \sum\limits_{i=0}^{n} f_k(x_i) f_1(x_i) = \sum\limits_{i=0}^{n} y_i f_1(x_i) \\ \quad \vdots \qquad\qquad \vdots \qquad \ddots \qquad \vdots \qquad\qquad \vdots \\ a_0 \sum\limits_{i=0}^{n} f_0(x_i) f_k(x_i) + a_1 \sum\limits_{i=0}^{n} f_1(x_i) f_k(x_i) + \cdots + \boldsymbol{a_k} \sum\limits_{\boldsymbol{i=0}}^{\boldsymbol{n}} \boldsymbol{f_k^2(x_i)} = \sum\limits_{i=0}^{n} y_i f_k(x_i) \end{cases} \tag{3.26}$$

Полученная СЛАУ – универсальная система по методу наименьших квадратов.

Алгоритмическая сложность МНК

Сложность составления матрицы: $O(n^3)$
Сложность вычисления СЛАУ: $O(n^3)$

3.3.1.1 Степенной базис

$$f_m(x) = x^m \tag{3.27}$$

Степенной базис используется для слабо нелинейных функций (слегка отличающихся от линейных), из-за того что, как выяснится дальше, матрица получаемой СЛАУ относится к классу матриц Вандермонда.

* графики: линейных, слабо нелинейных, не линейных функций *

Примеры

В качестве примера будет использоваться таблица с заданными x и y:

x	2	3	5
y	1	0	4

- $k = 0 \Rightarrow \varphi_0(x) = a_0$

$$(n+1)a_0 = \sum_{i=0}^{n} y_i \Rightarrow a_0 = \frac{\sum_{i=0}^{n} y_i}{n+1}$$

Другими словами – при порядке аппроксимации равном нулю, аппроксимирующая функция является средним арифметическим.

$$3a_0 = 5 \Rightarrow a_0 \approx 1.667$$

- $k = 1 \Rightarrow \varphi_1(x) = a_0 + a_1 x$

$$\begin{cases} 3a_0 + 10a_1 = 5 \\ 10a_0 + 38a_1 = 22 \end{cases}$$

$$\varphi_1(x) = -\frac{15}{7} + \frac{8}{7}x$$

- $k = 2 \Rightarrow \varphi_2(x) = a_0 + a_1 x + a_2 x^2$

$$\begin{cases} 3a_0 + 10a_1 + 38a_2 = 5 \\ 10a_0 + 38a_1 + 160a_2 = 22 \\ 38a_0 + 160a_1 + 722a_2 = 104 \end{cases}$$

$$\varphi_2(x) = 9 - 6x + x^2$$

Таким образом, как можно было заметить, при задании порядка аппроксимации (k) равному на единицу меньше количества точек (n), аппроксимация переходит в её частный случай – интерполяцию.

$$\text{аппроксимация} \xrightarrow{k=n} \text{интерполяция} \tag{3.28}$$

Данный факт также объясняется тем, что при увеличении количества слагаемых в функции, увеличивается количество её степеней свободы. А когда степеней свободы достаточно, график функции проходит через все заданные точки.

С другой стороны, если на вход подаются зашумлённые данные, то можно подобрать необходимый порядок аппроксимации $k \ll n$ (значительно меньший количества точек), для выделения тренда.

Также стоит заметить, что используя степенной базис, элементы матрицы коэффициентов в (3.26) возрастают с лева на право по квази степенному закону (близко к степенному). Следовательно получаемые матрицы коэффициентов в данном базисе относятся к классу матриц Вандермонда, которые плохо обусловлены уже начиная с 10-ого порядка.

$$\text{при } k \geq 10, \quad \operatorname{cond}(\boldsymbol{A}) \to \infty$$

Это обстоятельство является основным ограничителем степенной аппроксимации, из-за чего чаще используются другие базисы.

3.3.1.2 Аппроксимация в базисе взаимно ортогональных функций

Определение 3.3.1.1:

Функции $f_i(x)$ и $f_j(x)$ взаимно ортогональны, если:

$$\int_a^b f_i(x) f_j(x)\,dx = \begin{cases} \delta, & i = j \\ 0, & i \neq j \end{cases} \tag{3.29}$$

где: δ – некоторое число.

Общий алгоритм аппроксимации в базисе взаимно ортогональных функций

В данном алгоритме вместо интеграла используется сумма.
При большом количестве точек $\int \approx \sum$.

1. Масштабирование исходных значений x_i в интервал базиса (в котором не нарушается взаимная ортогональность):

$$[x_0; x_n] \to [a_0; a_n] \tag{3.30}$$

где: a_0 и a_n – левая и правые границы области значений функций базиса, при которых сохраняется взаимная ортогональность.

2. Из-за взаимной ортогональности функций, матрица коэффициентов СЛАУ получается диагональной. По этой причине расчёт коэффициентов можно производить с помощью следующей формулы:

$$a_m = \frac{\sum_{i=0}^{n} y_i f_m(x_i)}{\sum_{i=0}^{n} f_m^2(x_i)} \tag{3.31}$$

Базис Фурье

$$\frac{1}{2}, \cos(x), \cos(2x), \quad \cdots, \quad \cos\big((n-1)x\big), 0, \sin(x), \sin(2x), \quad \cdots, \quad \sin\big((n-1)x\big) \tag{3.32}$$

$$x \in [-\pi; +\pi]$$

Полиномы Чебышёва

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{m+1}(x) &= 2xT_m(x) - T_{m-1}(x) \end{aligned} \tag{3.33}$$

$$x \in [-1; +1]$$

Полиномы Бесселя (полиномы Чебышёва 2-ого рода)

$$\begin{aligned} U_0(x) &= 1 \\ U_1(x) &= \mathbf{2}x \\ U_{m+1}(x) &= 2xU_m(x) - U_{m-1}(x) \end{aligned} \tag{3.34}$$

$$x \in [-1; +1]$$

Полиномы Лежандра

$$\begin{aligned} P_0(x) &= 1 \\ P_1(x) &= x \\ P_{m+1}(x) &= \frac{(2m+1)xP_m(x) - mP_{m-1}(x)}{m+1} \end{aligned} \tag{3.35}$$

$$x \in [-1; +1]$$

Полиномы Лагерра

$$\begin{aligned} L_0(x) &= 1 \\ L_1(x) &= 1 - x \\ L_{m+1}(x) &= \frac{(2m+1-x)L_m(x) - mL_{m-1}(x)}{m+1} \end{aligned} \tag{3.36}$$

$$x \in [-1; +1]$$

3.3.2 Аппроксимация кривых

Каждая функция является кривой, но не каждая кривая является функцией (т.к. по определению функции, одному значению x соответствует не более одного значения y).

При аппроксимации функций порядок задания точек не имеет значения, однако, при аппроксимации кривых порядок важен: от этого зависит вид получаемой кривой.

Таким образом кривые параметризуется: вводится дополнительный параметр t, т.е. кривая задаётся системой из функций, где каждая координата точки кривой выражается как функция от t.

Пример

В случае окружности (кривой второго порядка):

$$x^2 + y^2 = R^2$$

$$\begin{cases} x = \cos(t) \\ y = \sin(t) \end{cases}$$

3.3.2.1 Аппроксимация Безье

$$\begin{cases} \beta_n(t)_x = \sum\limits_{i=1}^{n} x_i \Phi_{i,n}(t) \\ \beta_n(t)_y = \sum\limits_{i=1}^{n} y_i \Phi_{i,n}(t) \end{cases} \tag{3.37}$$

где: $t \in [0; 1]$ – параметр;
i – порядковый номер опорной вершины;
n – количество точек;
$\Phi_{i,n}(t)$ – полином Бернштейна:

$$\Phi_{i,n}(t) = C_n^i t^i (1-t)^{n-1} \tag{3.38}$$

$$C_n^k = \frac{n!}{k!(n-k)!}$$

Как можно заменить, вычисление функции Ф требует вычисление чисел большого порядка, а так же для вычисления позиции точки нужно суммировать n таких различных функций, что делает расчёт кривой Безье очень ресурсоёмкой. По этой причине используются другие, более вычислительно устойчивые алгоритмы.

Алгоритм де Кастельжо

Алгоритм де Кастельжо является «графическим» алгоритмом, в котором нет необходимости производить трудоёмкие вычисления.

3.4 Сглаживающая интерполяция (аппроксимация)

С помощью метода наименьших квадратов, найденная кривая проходит через все точки. Получаемая кривая может быть очень сильно изогнута (за изогнутость отвечает вторая производная функции). Поэтому для её сглаживания в критерий минимизации МНК вводится дополнительное слагаемое:

$$S = \sum_{i=0}^{n} \big(\varphi_k(x_i) - y_i\big)^2 + \rho \int_{x_0}^{x_n} \big(\varphi_k''(x)\big)^2 \, dx \tag{3.39}$$

где: ρ – коэффициент гладкости;
k – порядок аппроксимации (для интерполяции $k = n$).

Пример

В случае аппроксимации 4-ого порядка:

$$\begin{aligned} \varphi_4(x) &= a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 \\ \varphi_4'(x) &= a_1 + 2a_2 x + 3a_3 x^2 + 4a_4 x^3 \\ \varphi_4''(x) &= 2a_2 + 6a_3 x + 12a_4 x^2 \end{aligned} \tag{3.40}$$

$$\int_{x_0}^{x_n} 2a_2 + 6a_3 x + 12a_4 x^2 \, dx = 2a_2(x_n - x_0) + 3a_3(x_n^2 - x_0^2) + 4a_4(x_n^3 - x_0^3) \tag{3.41}$$

В результате частные производные от S будут выглядеть следующим образом:

$$\begin{cases} \dfrac{\partial S}{\partial a_0} = 2\displaystyle\sum_{i=0}^{n} \Big[\big(a_0 f_0(x_i) + \cdots + a_k f_k(x_i) - y_i\big) \cdot f_{\mathbf{0}(x_i)}\Big] + 0 = 0 \\ \dfrac{\partial S}{\partial a_1} = 2\displaystyle\sum_{i=0}^{n} \Big[\big(a_0 f_0(x_i) + \cdots + a_k f_k(x_i) - y_i\big) \cdot f_{\mathbf{1}(x_i)}\Big] + 0 = 0 \\ \dfrac{\partial S}{\partial a_2} = 2\displaystyle\sum_{i=0}^{n} \Big[\big(a_0 f_0(x_i) + \cdots + a_k f_k(x_i) - y_i\big) \cdot f_{\mathbf{2}(x_i)}\Big] + 2(x_n - x_0) = 0 \\ \dfrac{\partial S}{\partial a_3} = 2\displaystyle\sum_{i=0}^{n} \Big[\big(a_0 f_0(x_i) + \cdots + a_k f_k(x_i) - y_i\big) \cdot f_{\mathbf{3}(x_i)}\Big] + 3(x_n^2 - x_0^2) = 0 \\ \dfrac{\partial S}{\partial a_4} = 2\displaystyle\sum_{i=0}^{n} \Big[\big(a_0 f_0(x_i) + \cdots + a_k f_k(x_i) - y_i\big) \cdot f_{\mathbf{4}(x_i)}\Big] + 4(x_n^3 - x_0^3) = 0 \\ \dfrac{\partial S}{\partial a_5} = 2\displaystyle\sum_{i=0}^{n} \Big[\big(a_0 f_0(x_i) + \cdots + a_k f_k(x_i) - y_i\big) \cdot f_{\mathbf{5}(x_i)}\Big] = 0 \\ \quad \vdots \end{cases} \tag{3.42}$$

Вывод

Критерий сглаживания влияет только на правую часть универсальной СЛАУ МНК (3.26), причём только начиная со второго уравнения (т.к. производная от полинома первого порядка (от линейной функции) равна нулю), до уравнения с номером порядка аппроксимации.

4 Методы численного дифференцирования

$$y'(x) = \frac{dy}{dx} = \lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} \tag{4.1}$$

$$y'(x_0) = \operatorname{tg} \alpha \tag{4.2}$$

Если входные данные это аналитически заданная функция («$y = f(x)$»), то достаточно формул для нахождения производной. Если же входные данные это строго заданные значения функции и её аргументов (например, таблица значений x и y), то применяются методы численного дифференцирования, для нахождения производных в **заданных точках**.

В данной теме рассматриваются численные методы дифференцирования, которые подразумевают, что интервал между поступлением входных данных – постоянный:

$$\boldsymbol{h_i = x_i - x_{i-1} = \text{const}} \tag{4.3}$$

При выполнении условия (4.3) значения производной будут приблизительно равны отношению приращения функции к приращению аргумента, в силу того, что h является минимально доступным значением приращения аргумента:

$$y' \approx \frac{\Delta y}{h} \tag{4.4}$$

Численно не возможно добиться $\Delta x \to 0$, т.к. есть только заданные точки. Однако, если нет ограничений по времени можно предварительно по заданным точкам выполнить интерполяцию (или аппроксимацию), и тогда аналитически найти производную. Если же нужно быстрее, то существуют другие методы.

4.1 Метод конечной разности

В данном методе производная заменяется конечной разностью:

$$\begin{aligned} y'_k &\approx \frac{y_{k+1} - y_k}{h} \quad \text{“правая разность”} \\ y'_k &\approx \frac{y_k - y_{k-1}}{h} \quad \text{“левая разность”} \end{aligned} \tag{4.5}$$

Левая и правая разность имеют асимметрию, поэтому можно взять среднее значение между ними:

$$y'_k \approx \frac{y_{k+1} - y_{k-1}}{2h} \quad \text{“центральная разность”} \tag{4.6}$$

Для сравнения точности разностных схем используется следующее определение:

Определение 4.1.0.1:

Порядком точности разностной схемы называется степень (порядок) такого полинома, который с помощью данной разностной схемы дифференцируется без ошибки.

(добавить график левой и правой разности)

4.1.1 Вывод порядка точности разностных схем

Пусть y_k равняется значению функции в некоторой точке x, тогда используя ряд Тейлора, распишем чему будут равняться соседние значения y:

$$\begin{aligned} y_k &\to y(x) \\ y_{k+1} &\to y(x+h) = y(x) + y'(x)h + \frac{y''(x)h^2}{2!} + \cdots + \frac{y^{(n)}(x)h^n}{n!} + \cdots \\ y_{k-1} &\to y(x-h) = y(x) - y'(x)h + \frac{y''(x)h^2}{2!} + \cdots + (-1)^n \frac{y^{(n)}(x)h^n}{n!} + \cdots \end{aligned} \tag{4.7}$$

Далее, подставляем получившиеся значения в определённые выше (4.5 и 4.6) разностные схемы:

$$\begin{aligned} y'_k &\approx \frac{y_{k+1}-y_k}{h} &&= y'(x)+\frac{y''(x)}{2!}\boldsymbol{h}+\frac{y'''(x)}{3!}h^2+\cdots \\ y'_k &\approx \frac{y_k-y_{k-1}}{h} &&= y'(x)-\frac{y''(x)}{2!}\boldsymbol{h}+\frac{y'''(x)}{3!}h^2+\cdots \\ y'_k &\approx \frac{y_{k+1}-y_{k-1}}{2h} &&= y'(x)+\frac{y'''(x)}{3!}\boldsymbol{h^2}+\cdots \end{aligned} \tag{4.8}$$

По младшей степени h определяется степень точности разностной схемы. Таким образом получаем:

$$\begin{array}{ll} \text{Точность “левой” и “правой” разности:} & O(h) \\ \text{Точность “центральной разности”:} & O(h^2) \end{array} \tag{4.9}$$

4.1.2 Метод неопределённых коэффициентов

Данный метод используется для вывода формул разностных схем любой длины. В данном методе происходит поиск разностной схемы, которая абсолютно точно работает для полинома порядка n.

Рассмотрим пример для двух соседних точек (всего пяти точек).

$$y'_k = a_0 y_{k-2} + a_1 y_{k-1} + a_2 y_k + a_3 y_{k+1} + a_4 y_{k+2} \tag{4.10}$$

Пусть $y_k = y(x)$, тогда верно следующее выражение:

$$y_{k+m} = y(x + m \cdot h) \tag{4.11}$$

Для удобства x_k примем равным 0, тогда получаем:

$$y_{k+m} = y(m \cdot h) \tag{4.12}$$

Наконец, для определения неизвестных коэффициентов a, составляется СЛАУ необходимого размера:

$f(x)$	$f'(x)$
1	0
x	1
x^2	$2x$
x^3	$3x^2$
x^4	$4x^3$

$$\begin{cases} a_0\cdot 1 + a_1\cdot 1 + a_2\cdot 1 + a_3\cdot 1 + a_4\cdot 1 = 0 \\ a_0\cdot(-2h) + a_1\cdot(-h) + a_2\cdot 0 + a_3\cdot(h) + a_4\cdot(2h) = 1 \\ a_0\cdot(-2h)^2 + a_1\cdot(-h)^2 + a_2\cdot 0^2 + a_3\cdot(h)^2 + a_4\cdot(2h)^2 = 0 \\ a_0\cdot(-2h)^3 + a_1\cdot(-h)^3 + a_2\cdot 0^3 + a_3\cdot(h)^3 + a_4\cdot(2h)^3 = 0 \\ a_0\cdot(-2h)^4 + a_1\cdot(-h)^4 + a_2\cdot 0^4 + a_3\cdot(h)^4 + a_4\cdot(2h)^4 = 0 \end{cases} \tag{4.13}$$

Стоит заметить, что при взятии большего количества точек для формулы, увеличивается т.н. мёртвая зона – «окно», т.е. не возможно будет найти производную в начальных и конечных точках.

4.1.3 Теоретическая и практическая точность дифференцирования

С уменьшением h, теоретически, точность дифференцирования должна увеличиваться. Однако, из-за деления на малое число (h), при численном нахождении производной, погрешность вычислений начинает превышать выигрыш от уменьшения h (из-за ограниченности разрядной сетки).

(добавить график зависимости точности от h)

4.2 Вычисление старших производных

$$\begin{aligned} y''_k &= \frac{y'_{k+1}-y'_k}{h} \\ y'_{k+1} &= \frac{y_{k+1}-y_k}{h} \\ y'_k &= \frac{y_k-y_{k-1}}{h} \\ &\Downarrow \\ y''_k &= \frac{y'_{k+1}}{h}-\frac{y'_k}{h} = \frac{y_{k+1}-y_k}{h^2}-\frac{y_k-y_{k-1}}{h^2} = \frac{y_{k+1}-2y_k-y_{k-1}}{h^2} \end{aligned} \tag{4.14}$$

$$y'''_k = \frac{y_{k+2}-3y_{k+1}+3y_{k-1}-y_{k-2}}{h^3} \tag{4.15}$$

В силу того, что производные высших порядков зависят от выбора вида младшей производной, выходит, что для вычисления старших производных существует множество различных вариантов. Например, для второй производной в качестве y'_k можно взять как левую так и правую разность.

Жёсткого правила для выбора формулы производной высшего порядка – нет, поэтому выбор зависит от конкретно поставленной задачи.

5 Методы численного интегрирования

Интеграл от функции $f = y(x)$ можно интерпретировать различными образами:

- Геометрически – площадь криволинейной трапеции:

$$I = \int_a^b y(x)\,dx = F(b) - F(a) \tag{5.1}$$

- Первообразная функции (т.е. функция, производная от которой, даст подынтегральное выражение):

$$F(x) = \int y(x)\,dx + C \quad \Big| \quad \frac{dF(x)}{dx} = y(x) \tag{5.2}$$

В отличии от производной, не от всякой функции можно вычислить интеграл. Поэтому даже если функция задана аналитически, и её первообразную не возможно найти, то функцию искусственно приводят в табличный вид, теряя точность, но взамен получая возможность вычислить интеграл.

В случае вычисления определённого интеграла достаточно просуммировать площади S_k «бесконечно малые» (численно – зависит от требуемой точности) под криволинейной поверхностью функции:

$$\int_a^b y(x)\,dx = F(b) - F(a) \approx \sum_{k=1}^{n} S_k \tag{5.3}$$

В случае же аналитически заданной функции, численное интегрирование подразумевает переход к интегральной функции с переменным верхним пределом:

$$F(x) = \int_a^x y(x)\,dx = F(x) - F(a) \tag{5.4}$$

тогда значение $F(x)$ можно вычислить следующим образом:

$$\begin{cases} F(x_0) = 0, & \text{т.к. } \int_a^a \ldots dx = 0 \\ F(x_{k+1}) = F(x_k) + S_k \end{cases} \tag{5.5}$$

Методы вычисления S_k уже зависят от выбранного метода численного интегрирования.

Теорема 5.0.1 (Теорема Котельникова)**:**

Любую функцию $F(t)$ состоящую из частот от 0 до ΔF, можно непрерывно передавать дискретно с любой точностью через $1/2\Delta F$ секунд.

$$\Delta t = \frac{1}{2\Delta F} \tag{5.6}$$

5.1 Квадратурные формулы Ньютона-Котеса

Квадратурные формулы Ньютона-Котеса применимы только при фиксированном изменении x, и служат для оценки значения определённого интеграла.

Определение 5.1.0.1:

Квадратура – вычисление значения определённого интеграла.

5.1.1 Метод прямоугольников (0 порядок квадратуры Ньютона-Котеса)

$$S_k = h \cdot y_{k-1} \tag{5.7}$$

5.1.2 Метод трапеций (1 порядок квадратуры Ньютона-Котеса)

$$S_k = \frac{h}{2}\left(y_{k-1} + y_k\right) \tag{5.8}$$

При выполнении теоремы Котельникова (5.0.1) метода трапеций вполне хватает для получения достаточно точных результатов.

5.1.3 Метод Симпсона (2 порядок квадратуры Ньютона-Котеса)

$$S_k = \frac{h}{6}(y_{k-1} + 4y_k + y_{k+1}) \tag{5.9}$$

5.1.4 Квадратура третьего порядка

$$S_k = \frac{3h}{8}(y_{k-2} + 3y_{k-1} + 3y_{k+1} + y_{k+2}) \tag{5.10}$$

5.1.5 Квадратура порядка m

Квадратура любого порядка m получается с помощью вычисления определённого интеграла на отрезке $[x_{k-1}; x_k]$ от интерполирующего полинома порядка m:

$$S_k(x) = \int\limits_{x_{k-1}}^{x_k} y_m(x)\,dx \tag{5.11}$$

где: $y_m(x)$ – некоторый интерполирующий полином порядка m (например, рассчитанный методом Лагранжа).

Таким образом, в общем случае квадратура порядка m будет выглядеть следующим образом:

$$S_k = \int\limits_{x_{k-1}}^{x_k} y_m(x)\,dx = a_0 + a_1 x + \cdots + a_m x^m \tag{5.12}$$

5.2 Сплайн-квадратура

Для численного интегрирования с **не равномерным** шагом применяется метод «сплайн квадратуры».

$$h \neq \text{const}$$

Формула для i-ого сплайна (согласно определению (3.8)):

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

Как упоминалось ранее, в сплайнах значащий коэффициент – только c_i, поэтому при интегрировании остаётся только данный коэффициент:

$$S_k = \int\limits_{x_{k-1}}^{x_k} y(x)\,dx = \frac{h}{2}(y_{k-1} + y_k) - \frac{h^3}{24}(c_{k-1} + c_k) \tag{5.13}$$

5.3 Метод адаптивного интегрирования

Название адаптивное интегрирование происходит из того, что на каждом шаге итерации вычисляется площадь под криволинейной поверхностью с заданной точностью. Таким образом данный метод не зависит от «извилистости» кривой, и на каждой итерации вычисляется площадь, адаптируясь под заданную точность.

В основе данного метода может быть любой из выше упомянутых методов, но в силу того, что в этом методе присутствуют явно выраженные итерации для нахождения одной из S_k, чаще всего используют наиболее простой и эффективный – метод трапеций.

В качестве входного параметра алгоритма необходим начальный шаг интегрирования h_0, хотя его можно брать как оставшуюся длину интервала, в силу того, что при каждой итерации будет найден такой шаг, при котором удовлетворяется заданная точность ε.

Алгоритм

1. Вычисление начального значения S_k:
$$S_k^{(0)} = S(h_0)$$
2. Деление шага интегрирования пополам:
$$h_i = \frac{h_0}{2}$$
3. Вычисление следующего значения S_k, как сумму двух площадей с половинным интервалом:
$$S_k^{(1)} = S_{k_1}(h_i) + S_{k_2}(h_i)$$
4. Если требуемая точность удовлетворена, то переход к следующей итерации ($k = k + 1$), иначе «дробление» шага h_i продолжается.

$$\text{Если} \quad \left|S_k^{(1)} - S_k^{(0)}\right| < \varepsilon, \quad \text{то } k = k + 1, \text{ иначе переход на шаг 2.}$$

5.4 Квадратура Гаусса

Квадратура Гаусса также носит название “квадратурная формула повышенной (удвоенной) точности”.

Данный метод применим только когда функция задана аналитически, и необходимо вычислить определённый интеграл.

$$I = \int_a^b y(x)\,dx \tag{5.14}$$

В данном методе интеграл рассматривается как сумма произведений значений самой функции $y(x_m)$ на какой-то коэффициент c_m:

$$I \approx \sum_{m=1}^{n} c_m \cdot y(x_m) \tag{5.15}$$

В силу того что, суммируются два неизвестных n раз, необходимо найти $2n$ неизвестных.

Предположим, что такая формула точна для любого полинома степени $m \in [0; 2n-1]$, при интегрировании от -1 до $+1$:

$$y(x) = x^m$$

Вычислим интеграл от t^m на отрезке $[-1; +1]$:

$$\begin{aligned}\int_{-1}^{+1} t^m\,dt = \left.\frac{t^{m+1}}{m+1}\right|_{-1}^{+1} = \frac{1^{m+1}}{m+1} - \frac{(-1)^{m+1}}{m+1} = \\ = \frac{1}{m+1} - \frac{(-1)^{m+1}}{m+1} = \begin{cases} \dfrac{2}{m+1}, & m \text{ – чётное} \\ 0, & m \text{ – не чётное} \end{cases}\end{aligned} \tag{5.16}$$

таким образом возможно составить следующую систему уравнений, решая которую будут найдены неизвестные коэффициенты:

$$\begin{cases} c_1 t_1^0 + c_2 t_2^0 + \cdots + c_n t_n^0 = 2 & \text{при } m = 0 \\ c_1 t_1^1 + c_2 t_2^1 + \cdots + c_n t_n^1 = 0 & \text{при } m = 1 \\ c_1 t_1^2 + c_2 t_2^2 + \cdots + c_n t_n^2 = 2/3 & \text{при } m = 2 \\ \vdots \quad\quad \vdots \quad\quad \ddots \quad\quad \vdots \quad\quad \vdots & \\ c_1 t_1^{2n-1} + c_2 t_2^{2n-1} + \cdots + c_n t_n^{2n-1} = 0 & \text{при } m = 2n-1 \end{cases} \tag{5.17}$$

Однако, данную систему нет необходимости решать, т.к., доказано, что коэффициенты t являются корнями полинома Лежандра (3.35) порядка n (приравненного к нулю):

$$P_n(t) = 0 \tag{5.18}$$

Из данного полинома вычисляются n корней $t_i | i \in [1, n]$. Далее вычисляются коэффициенты c_i используя полученные корни с помощью следующей формулы:

$$c_i = \frac{2}{(1 - t_i^2) \cdot P_n'(t_i)^2} \tag{5.19}$$

Наконец, для вычисления значения интеграла I необходимо произвести масштабирование из $[-1; +1]$ в исходные $[a; b]$ как самого значения I так и найденных корней t_i в «опорные точки» x_m:

$$x_m = \frac{b + a}{2} + \frac{b - a}{2} t_m \tag{5.20}$$

$$I \approx \frac{b - a}{2} \sum_{m=1}^{n} c_m \cdot y(x_m) \tag{5.21}$$

Пример

$$y(x) = 2x^2 + 5x$$

Порядок квадратуры равен 2, поэтому интеграл (5.15) приблизительно равен:

$$I \approx c_1 y(x_1) + c_2 y(x_2)$$

Полином Лежандра $P_2(t)$, и его производная:

$$P_2(t) = \frac{3t^2 - 1}{2}$$
$$P_2'(t) = 3t$$

Решаем уравнение $P_2(t) = 0$:

$$\frac{3t^2 - 1}{2} = 0$$
$$3t^2 = 1$$
$$t_{1,2} = \pm\frac{1}{\sqrt{3}}$$

Далее вычисляются коэффициенты c_i согласно формуле (5.19):

$$\begin{aligned} c_{1,2} &= \frac{2}{(1 - \frac{1}{3}) \cdot [3 \cdot (\pm\frac{1}{\sqrt{3}})]^2} \\ &= \frac{2}{\frac{2}{3} \cdot 3} \\ &= 1 \end{aligned}$$

Наконец можно вычислить значение интеграла, не забыв при этом произвести масштабирование от отрезка $[-1; +1]$ к $[a; b]$:

$$I = \frac{b - a}{2} \cdot \big(1 \cdot y(x_1) + 1 \cdot y(x_2)\big)$$

$$I \approx \frac{6-0}{2}\Big(1 \cdot y(x_1) + 1 \cdot y(x_2)\Big)$$

$$t_{1,2} = \pm\frac{1}{\sqrt{3}} = \pm 0.557$$

$$x_1 = \frac{6+0}{2} + \frac{6-0}{2} \cdot \left(-\frac{1}{\sqrt{3}}\right) = 1.268$$
$$x_2 = \frac{6+0}{2} + \frac{6-0}{2} \cdot \left(+\frac{1}{\sqrt{3}}\right) = 4.732$$

$$f(1.268) = 9.556$$
$$f(4.732) = 68.444$$

$$I \approx \frac{6-0}{2}\Big(9.556 + 68.444\Big) = 234$$

Проверка:

$$I = \int_0^6 (2x^2 + 5x)\,dx = \left(\frac{2}{3}x^3 + \frac{5}{2}x^2\right)\Big|_0^6 = 234$$

6 Методы решения нелинейных уравнений

6.0.1 Виды уравнений

Алгебраические уравнения $f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$

Трансцендентные уравнения $f(x) = e^x, \ln(x), \sin(x)$

6.0.2 Алгебраические уравнения

$$n = 1 \quad : \quad a_0 + a_1 x = 0 \quad \Rightarrow \quad x = -\frac{a_0}{a_1}$$

$$n = 2 \quad : \quad a_0 + a_1 x + a_2 x^2 = 0 \quad \Rightarrow \quad x_{1,2} = \frac{-a_1 \pm \sqrt{a_1^2 - 4a_0 a_2}}{2a_2}$$

$n = 3$:
$n = 4$: Существуют аналитические решения только для особых видов

$n = 5$:
$\vdots$ Согласно теореме Абеля не существует аналитического решения

6.0.3 Трансцендентные уравнения

Для трансцендентных уравнений в общем случае нет алгебраического решения.

6.1 Методы

Все методы решения нелинейных уравнений, в качестве входных данных, требуют как минимум, отрезок $[a; b]$, на котором может быть интересующий корень.

По первой теореме Коши, если $f(a) \cdot f(b) \leq 0$, то функция хотя бы один раз пересекает O_x, а, значит на отрезке $[a; b]$ есть корень. Однако, могут случатся ситуации, когда на отрезке $[a; b]$ находится кратное число корней, тогда произведение $f(a) \cdot f(b) > 0$.

Таким образом в случае необходимости нахождения всех корней на отрезке, требуется сделать поиск «окон», в которых могут находится корни.

Может случится следующая ситуация, когда значения $|y|$ достаточно близки к 0, однако значение x находится далеко от корня:

(вставить график)

Поэтому с заданной точностью сравнивают изменения значения x а не y:

$$|x_{k+1} - x_k| < \varepsilon \tag{6.1}$$

Условие сходимости итерационных методов решения уравнений:

$$\text{при } k \to \infty, x_k \to x^* \tag{6.2}$$

где: k – количество итераций
x_k – корень, найденный на итерации k
x^* – искомый корень.

Условие сходимости проверяется с помощью:

$$|x_{k+1} - x_k| < |x_k - x_{k-1}| \tag{6.3}$$

Также его можно записать следующим образом:

$$|x_{k+1} - x_k| = C \cdot |x_k - x_{k-1}|^p \tag{6.4}$$

где: $C < 1$ – некоторая константа;
p – **порядок скорости сходимости**.

При $p = 1$: линейная сходимость;
$p = 2$: квадратичная сходимость (удвоенная);

6.2 Методы с линейной сходимостью

6.2.1 Метод половинного деления

Также известен как метод бисекции.

6.2.1.1 Алгоритм

1. Проверка наличия корней
$$f(a) \cdot f(b) \leq 0$$
2. Вычисление середины отрезка по оси x:
$$x = \frac{a+b}{2}$$
3. – Если $f(a) \cdot f(x) \leq 0$, то $b = x$,
– иначе $a = x$.
4. Если $|b - a| \leq \varepsilon$, то корень с заданной точностью найден, иначе переход к шагу 2.

Метод половинного деления самый медленный, однако он обладает **абсолютной устойчивостью**, т.е. сходимость гарантирована. Причём можно спрогнозировать время вычисления.

Каждый раз отрезок на котором ищется корень делится пополам, а критерий остановки – длина отрезка меньше ε. Пусть длина отрезка станет меньше чем ε за k шагов:

$$|b - a| \cdot 2^{-k} \leq \varepsilon \tag{6.5}$$

тогда

$$k = \log_2 \frac{|b-a|}{\varepsilon} \tag{6.6}$$

6.2.2 Метод золотого сечения

Также известен как «метод Фибоначчи».

6.2.2.1 Определение золотого сечения

Пусть m есть малая часть целого, а n – большая. И пусть эти части относятся друг к другу следующим образом:

$$\frac{m}{n} = \frac{n}{n+m} \tag{6.7}$$

Тогда $n+m$ – целое, и обозначим целое за 1:

$$\begin{cases} n+m=1 \\ m = \dfrac{n \cdot n}{n+m} \end{cases} \text{ – выражено из (6.7)} \tag{6.8}$$

Если в $n+m=1$ подставить m из второго равенства:

$$\begin{aligned} n+m &= 1 \\ n + \frac{n \cdot n}{n+m} &= 1 \\ n + \frac{n^2}{1} - 1 &= 0 \end{aligned}$$

получается следующее квадратное уравнение:

$$n^2 + n - 1 = 0 \tag{6.9}$$

корни которого являются:

$$n_{1,2} = \frac{-1 \pm \sqrt{1+4}}{2} = \frac{\pm\sqrt{5}-1}{2} \tag{6.10}$$

Первый из полученных корней называется «золотым сечением»:

$$\Phi = \frac{\sqrt{5}-1}{2} \approx 0.618 \tag{6.11}$$

Тогда обратное число, обозначаемое как φ равняется:

$$\varphi = 1 - \Phi \approx 0.382 \tag{6.12}$$

От сюда можно сделать практический вывод: деление в пропорции золотого сечения – это деление какой либо величины в отношении $\approx 62\%$ и $\approx 38\%$.

6.2.2.2 Метод

Метод золотого сечения отличается от метода половинного деления, только тем, что отрезок на котором происходит поиск корня делится не пополам, а в пропорции золотого сечения.

Таким образом шаг 2 половинного деления заменяется на:

2. – Если $\big|f(a)\big| > \big|f(b)\big|$, то $x = a + \Phi|b-a|$,
   – иначе $x = a + \varphi|b-a|$.

В силу того, что на каждом шаге (в среднем) отсекается большая часть отрезка на котором нет корня, то данный метод быстрее метода половинного деления не более чем в полтора раза.

6.2.3 Метод хорд

Также известен как «метод секущих».

Метод хорд, как и метод золотого сечения, отличается от метода бисекции только шагом №2, на котором вычисляется новая оценка корня.

Основная идея: т.к. на отрезке $[a;b]$ есть корень, то функция подменяется хордой от a до b и в качестве корня выбирается пересечение хорды с осью O_x:

$$x = b - \frac{b-a}{f(b)-f(a)} f(b) \tag{6.13}$$

(добавить график)

Метод хорд не работает на монотонных функциях, т.к. может произойти ситуация когда $|b-a| > \varepsilon$, но a упирается в истинное значение. Поэтому за условие остановки следует взять изменение длины отрезка:

$$\Delta|b-a| < \varepsilon \tag{6.14}$$

6.3 Методы с квадратичной сходимостью

Методы с квадратичной сходимостью, в отличии от линейных, являются точечными, а не интервальными. Поэтому задание отрезка $[a;b]$ в данных методах необходимо только для выбора начальной точки.

6.3.1 Метод Ньютона

Также известен как «метод касательных».

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \tag{6.15}$$

Критерий остановки:

$$|x_{k+1} - x_k| \leq \varepsilon \tag{6.16}$$

Метод Ньютона чрезвычайно быстр, однако совершенно нестабилен.

6.3.2 Метод Ньютона с коррекцией

Для стабилизации метода Ньютона вводится коэффициент коррекции α. Изначально предполагается, что $\alpha = 1$. Далее вычисляется следующее значение x_k:

$$x_{k+1} = x_k - \alpha \cdot \frac{f(x)}{f'(x)} \tag{6.17}$$

В случае если новое значение $f(x_{k+1})$ больше старого значения $f(x_k)$, коэффициент коррекции уменьшается:

$$\text{Если } \big|f(x_{k+1})\big| \geq \big|f(x_k)\big|, \text{ то } \alpha = \frac{\alpha}{2} \tag{6.18}$$

6.3.3 Метод Парабол

Метод парабол основывается на вычислении точки пересечения параболы, проведённой через 3 заданные точки, с осью O_x.

Пусть изначально заданы левая и правая границы поиска корня $[a;b]$, и начальное приближение x_0:

x	a	x_0	b
y	$f(a)$	$f(x_0)$	$f(b)$

Тогда по заданным точкам, например, с помощью метода интерполяции Лагранжа, возможно построить параболу (проходящую через все 3 точки: a, x_0, b).

$$L_2(x) = a_0 + a_1 x + a_2 x^2 \tag{6.19}$$

Пусть парабола построена (найдены коэффициенты a_0, a_1, a_2 квадратного уравнения), тогда возможно вычислить точки пересечения параболы с осью O_x – это будут корни квадратного уравнения:

$$x_{1,2} = \frac{-a_1 \pm \sqrt{a_1^2 - 4a_0 a_2}}{2a_2} \tag{6.20}$$

Из двух корней необходимо выбрать тот, который принадлежит изначальному отрезку $[a;b]$, предположим, что это корень x_1, тогда:

– Если $x_1 \in [x_0;b]$, то $a = x_0$;
– иначе $b = x_0$.

Далее в качестве нового приближения x_0 выбирается x_1 (т.е. $x_0 = x_1$).

В качестве условия остановки, как и в остальных методах используется сравнение нового приближения со старым относительно требуемой точности:

$$|x_1 - x_0| \leq \varepsilon \tag{6.21}$$

Можно заметить что в методе парабол больше вычислений, чем в предыдущих квадратичных методах, однако, он обладает абсолютной устойчивостью.

6.3.4 Метод Якоби (простых итераций)

1. Приведение к удобному для итераций виду. Для этого к левой и правой частям уравнения достаточно добавить x:

$$\begin{gathered} f(x) + \boldsymbol{x} = 0 + \boldsymbol{x} \\ \Downarrow \\ x = \varphi(x) \end{gathered} \tag{6.22}$$

2. Выбор начального приближения для $x \in [a; b]$, например:

$$x_0 = \frac{a+b}{2}$$

3. Вычисление значения x на следующей итерации:

$$x_{k+1} = \varphi(x_k) \tag{6.23}$$

4. Если $|x_{k+1} - x_k| \leq \varepsilon$, то решение с заданной точностью найдено, иначе повторять шаг 3 пока не будет достигнута требуемая точность.

Метод простых итераций работает достаточно быстро, **если сходится**!
Условие сходимости:

$$\left|\varphi'(x)\right| < 1 \quad \forall\, x \tag{6.24}$$

А значит, как минимум:

$$\left|\varphi'(x_0)\right| < 1 \tag{6.25}$$

Пример

$x^2 - 4x + 3 = 0$

$$\begin{gathered} x^2 - 4x + 3 + \boldsymbol{x} = 0 + \boldsymbol{x} \\ \Downarrow \\ \varphi(x) = x^2 - 3x + 3 \end{gathered}$$

Пусть $x_0 = 5$, тогда ожидаемый корень, который должен быть найден $x^* = 3$.
Проверка условия сходимости:

$$\begin{gathered} \varphi'(x) = 2x - 3 \\ \left|\varphi(x_0)\right| = |2 \cdot 5 - 3| = 7 > 1 \Rightarrow \text{не сходится} \end{gathered}$$

Пусть $x_0 = 1.1$, тогда ожидаемый корень, который должен быть найден $x^* = 1$.
Проверка условия сходимости:

$$\left|\varphi(x_0)\right| = |2 \cdot 1.1 - 3| = 0.8 < 1 \Rightarrow \text{сходится}$$

$$\begin{aligned} x_0 &= 1.1 \\ x_1 &= 1.1^2 - 3 \cdot 1.1 + 3 = 0.91 \\ x_2 &= 0.91^2 - 3 \cdot 0.91 + 3 = 1.0981 \\ x_3 &= 1.0981^2 - 3 \cdot 1.0981 + 3 = 0.91152 \\ x_4 &= 0.91152^2 - 3 \cdot 0.91152 + 3 = 1.0963 \\ x_5 &= 1.0963^2 - 3 \cdot 1.0963 + 3 = 0.91297 \end{aligned}$$

Можно заметить, что x медленно приближается к ожидаемому корню $x^* = 1$.

6.3.4.1 Коррекция

Известно, что если левую и правую часть уравнения домножить на одно и тоже число, то его корни не изменятся:

$$\boldsymbol{k} \cdot f(x) = 0 \tag{6.26}$$

Таким образом $\varphi(x)$ и его производная будут:

$$\varphi(x) = x + kf(x) \tag{6.27}$$
$$\varphi'(x) = 1 + kf'(x) \tag{6.28}$$

Теперь, для того чтобы выполнялось условие сходимости $\left|\varphi'(x)\right| < 1$ можно подобрать значение k:

$$\left|\varphi'(x)\right| < 1$$
$$\left|1 + kf'(x)\right| < 1$$
$$\begin{cases} 1 + kf'(x) < 1 \\ 1 + kf'(x) > -1 \end{cases}$$

в случае $f'(x) > 0$

$$\begin{cases} k < 0 \\ k > -\dfrac{2}{f'(x)} \end{cases}$$

$$k \in \left(-\frac{2}{f'(x)}; 0\right)$$

или

в случае $f'(x) < 0$

$$\begin{cases} k > 0 \\ k < -\dfrac{2}{f'(x)} \end{cases}$$

$$k \in \left(0; -\frac{2}{f'(x)}\right)$$

7 Методы решения систем нелинейных уравнений

Пусть имеется система нелинейных уравнений:

$$\begin{cases} f_1(x_1, ..., x_n) = 0 \\ f_2(x_1, ..., x_n) = 0 \\ \quad\vdots \\ f_n(x_1, ..., x_n) = 0 \end{cases} \tag{7.1}$$

Данную систему можно представить в виде матричной функции:

$$F(X) = 0 \tag{7.2}$$
$$\tag{7.3}$$

тогда

$$X = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \tag{7.4}$$

где: F – матричная функция;
X – вектор (столбец) решений уравнений.

Существование решения сложно определить, а тем более область.

7.1 Метод простых итераций (метод Якоби)

1. Приведение к удобному для итераций виду (подобно, как это делалось ранее)

$$F(X) + \boldsymbol{X} = 0 + \boldsymbol{X}$$
$$\Downarrow$$
$$X = \Phi(X) \tag{7.5}$$

2. Выбор начального приближения, например $X_{(0)} = [0, ..., 0]$

3. Вычисление X на следующей итерации:

$$X_{(k+1)} = \Phi(X_{(k)})$$

4. Если $\max\left|x_{i_{(k+1)}} - x_{i_{(k)}}\right| < \varepsilon$, то решение с заданной точностью найдено, иначе повторять шаг 3, пока не будет достигнута требуемая точность.

Аналогично методу простых итераций для одного нелинейного уравнения, условие сходимости в данном случае:

$$\left|\left|W(X_{(0)})\right|\right| < 1 \tag{7.6}$$

Т.е. норма матрицы Якоби должна быть меньше единицы, где матрица Якоби:

$$W = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{7.7}$$

$\det(W)$ называется **якобианом**.

7.1.1 Коррекция

Аналогично коррекции, которая делается для одного уравнения, можно произвести коррекцию данного метода для систем нелинейных уравнений. Однако, в данном случае потребуется вектор коэффициентов K:

$$K = \begin{bmatrix} k_1 \\ \vdots \\ k_n \end{bmatrix}$$

Однако, если методом Якоби (без коррекции) невозможно решить систему нелинейных уравнений, то обычно используют другой метод вместо коррекции в силу большого числа дополнительных вычислений, связанных с введением коэффициентов K.

7.2 Метод Ньютона

Метод Ньютона в случае одного нелинейного уравнения (согласно определению (6.15)) выглядит следующим образом:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

В многомерном случае (для системы нелинейных уравнений) метод Ньютона определяется следующим образом:

$$X_{(k+1)} = X_{(k)} - W\left(X_{(k)}\right)^{-1} \cdot F\left(X_{(k)}\right) \tag{7.8}$$

7.3 Методы оптимизации

Из системы нелинейных функций (7.1) создаётся функционал F, минимизируя который, находится решение исходной системы:

$$F(x_1, ..., x_n) = f_1^2(x_1, ..., x_n) + \cdots + f_n^2(x_1, ...x_n) = \min \equiv 0 \tag{7.9}$$

Таким образом система нелинейных уравнений решается обходным путём – через задачу минимизации.

Задача минимизации решается с помощью методов оптимизации. Причём в отличии от решения системы уравнений, решение будет найдено в любом случае (даже если не равное нулю).

При попытке решения задачи минимизации напрямую (с использованием частных производных), получается система нелинейных уравнений эквивалентная начальной («попадаем туда от куда пытались уйти»):

$$\begin{cases} \dfrac{\partial F}{\partial x_1} = 0 \\ \vdots \\ \dfrac{\partial F}{\partial x_n} = 0 \end{cases} \Leftrightarrow \begin{cases} f_1(x_1, ..., x_n) = 0 \\ \vdots \\ f_n(x_1, ..., x_n) = 0 \end{cases} \tag{7.10}$$

7.3.1 Метод случайного поиска Монте-Карло

7.3.2 Методы спуска

7.3.2.1 Метод покоординатного поиска

Данный метод аналогичен настройке осциллографа – все «ручки» фиксируются в определённом положении, и далее крутится одна, выбирая «лучшее» положение.

Алгоритм

1. Задаётся начальная точность ε;
2. Выбирается начальное приближение $X_{(0)}$;
3. Поиск нового локального минимума (по одному из аргументов);
4. Фиксация данного аргумента;
5. Если $\left|F(X_{(i)}) - F(X_{(i-1)})\right| < \varepsilon$, то решение с требуемой точностью найдено (конец алгоритма); иначе
6. Выбирается другой аргумент, и заново проводится поиск минимума (переход на шаг 3).

7.3.2.2 Градиентный поиск

1. Выбор начального приближения $X_{(0)}$;
2. Расчёт градиента (направление в котором наиболее быстро растёт функция):

$$\vec{G} = \left(\frac{\partial F}{\partial x_1}, ..., \frac{\partial F}{\partial x_n}\right) = (f_1, ..., f_n) \tag{7.11}$$

3. Шаг длиной ε в противоположную сторону направления $\vec{G}$;
4. Повторяются предыдущие шаги, до достижения требуемой точности.

7.3.2.3 Метод наискорейшего поиска

Данный метод также известен как «метод корабельной навигации».

1. Выбирается начальное приближение $X_{(0)}$;
2. Расчёт $\vec{G}$;
3. Итерационно совершаются шаги длиной ε в направлении противоположном $\vec{G}$, до тех пор, пока значение функции на новом шаге не увеличится.
4. Если был сделан только один шаг, то ответ при данном ε был найден, иначе переход на шаг 2.

7.4 Эволюционные методы

7.4.1 Генетические алгоритмы

Генетические алгоритмы представляют собой **имитацию симуляций**, где каждый шаг (итерация) называется **поколением**, а каждая особь в поколении – **геномом**.

Алгоритм

1. Выбор начальной популяции (например, $N_0 > 500$). Важно иметь возможность собирать статистические данные.
2. "Кроссовер" (скрещивание). Пример:
   (a) Выбираются пары случайным образом;
   (b) Случайным образом выбирается точка скрещивания;
   (c) Для каждой пары создаются два ребёнка, которые берут начало у одного из родителей и дополняют вторую часть от второго родителя. В зависимости от типа задачи применяются различные способы для разрешения случая, когда в части генома, берущегося у второго родителя, присутствуют похожие элементы генома.
3. Селекция (отбор). В этом шаге для отбора наиболее приспособленных особей применяется фитнес функция F, которая является специфичной для решаемой задачи. Т.е. задачей данного шага является сокращение популяции.
4. Мутация (проверка устойчивости популяции случайными изменениями). У некоторых особей (случайно выбранных) элемент генома меняется случайным образом.
5. Если количество родственников в популяции (особей с одинаковым геномом) превышает, например, 85%, то можно останавливать алгоритм, иначе переход на шаг 2.

7.4.1.1 Методы отбора

- Элитный отбор – остаются наиболее приспособленные (может застрять в локальном минимуме);
- Турнирный отбор – попарно сравнивают особи; наиболее приспособленная выигрывает;
- "Рулетка" – данный метод аналогичен игре в рулетку. Для каждой особи выделяется свой сектор в рулетке, размеры которого зависят от фитнес функции (чем лучше особь приспособлена, тем шире сектор). Далее пока не отобрано достаточное количество особей, "крутится рулетка" – выбирается одна особь. Причём выбор одной и той-же особи несколько раз не является проблемой, ибо критерий остановки – большое количество родственников в популяции.

8 Методы решения дифференциальных уравнений

8.1 Введение в дифференциальные уравнения

Определение 8.1.0.1:

Дифференциальное уравнение – уравнение, в которое входят производные функции, и может входить сама функция, а также независимая переменная и параметры.

Решением ОДУ является функция $y(t)$.

Пример

Порядок дифференциального уравнения	Уравнение
1	$y' = f(t, y)$
2	$y'' = f(t, y, y')$
3	$y''' = f(t, y, y', y'')$

Определение 8.1.0.2:

Обыкновенное дифференциальное уравнение (ОДУ) – это дифференциальное уравнение для функции от одной переменной.

8.1.1 Дифференциальные уравнения частных производных (ДУЧП)

Определение 8.1.1.1:

Дифференциальное уравнение частных производных (ДУЧП) – это дифференциальное уравнение для функции от нескольких переменных.

Пример

$$\frac{\partial^2 u}{\partial x^2} = \frac{1}{v_0^2} \cdot \frac{\partial^2 u}{\partial t^2} \tag{8.1}$$

$$u_{xx} = \frac{1}{v_0^2} \cdot u_{tt} \tag{8.2}$$

Решением дифференциального уравнения частных производных (в данном случае) является функция $u(x,t)$.

8.1.2 Классификация дифференциальных уравнений

- Обыкновенное или с частными производными;
- Порядок уравнения;
- Линейное или нелинейное.

8.1.3 Вопрос о корректности задачи

$$y' = f(t,y) \tag{8.3}$$

$$\int y'\,dt = \int f(t,y)\,dt \tag{8.4}$$

$$y(t) = \int f(t,y)\,dt + \boldsymbol{C} \tag{8.5}$$

Таким образом из-за константы C для дифференциальных уравнений существует бесконечное множество решений. Поэтому **для однозначного решения** необходимо задать начальные условия (C), например в виде значения $y(0)$.

В случае более высокого порядка дифференциального уравнения необходимо задавать больше начальных условий. В общем случае: *каков порядок уравнения, столько условий должно быть задано.*

Уравнение			Начальные условия
y''	$=$	$f(t,y,y')$	$y(0), y'(0)$
y'''	$=$	$f(t,y,y',y'')$	$y(0), y'(0), y''(0)$
	$\vdots$		

Определение 8.1.3.1:
Задача Коши – ОДУ с n начальными условиями.

Теорема 8.1.1:
Для нахождения единственного решения ОДУ порядка n необходимо n начальных условий.

В случае ОДУ первого порядка, никаких вариантов нет, кроме как задать какое-либо значение $y(x)$, однако для старших порядков ОДУ существуют альтернативы:

Задача	Начальные условия
Краевая / граничная задача	$y(0)$, $y(T)$
Смешанная задача	$y(0)$, $y'(0)$, $y(T)$
	$\vdots$

8.1.4 Методы решения

8.1.4.1 Графические

8.1.4.2 Аналитические

$$\frac{dy}{dt} = f(t,y), \quad y(0) = y_0 \tag{8.6}$$

Попытаться растащить y и t по левую и правую сторону уравнения, так, чтобы потом можно было проинтегрировать:

$$f_1(y)dy = f_2(t)dt \tag{8.7}$$

Пример

$$y' = \frac{xy}{1+x^2}, \quad y(0) = 1$$

$$\begin{aligned}
\frac{dy}{dx} &= \frac{xy}{1+x^2}\Bigg| \cdot \frac{dx}{y} \\
\frac{dy}{y} &= \frac{x}{1+x^2}\,dx \\
\int \frac{dy}{y} &= \int \frac{x}{1+x^2}\,dx \\
\ln|y| &= \frac{1}{2}\ln\left(1+x^2\right) + C \\
y &= e^C \cdot \sqrt{1+x^2} \\
y(0) &= e^C \cdot \sqrt{1+0^2} = 1 \quad \Rightarrow C = 0 \\
&\Downarrow \\
\text{Ответ: } y(x) &= \sqrt{1+x^2}
\end{aligned}$$

8.2 Линейные дифференциальные уравнения

Существует аналитический интеграл.

$$y' = f(t,y), \quad y(0) = y_0 \tag{8.8}$$

8.2.1 Классический метод

Классический метод подразумевает линейное дифференциальное уравнение, приведённое к стандартному виду:

$$y^{(n)} + a_1 y^{(n-1)} + \cdots + a_{n-1}y' + a_n y = f(y) \tag{8.9}$$

Тогда решением будет:

$$y(t) = \sum_{i=1}^{n} A_i \cdot e^{p_i t} \tag{8.10}$$

где: p_i – i-ый корень характеристического уравнения.

8.2.1.1 Частный случай: $f(t) = \text{const}$

В случае $f(t) = \text{const}$, полное решение $y(t)$ можно рассмотреть как следующую сумму:

$$y(t) = y_0(t) + y(\infty) \tag{8.11}$$

где: $y(\infty)$ – принуждённая составляющая, к которой будет стремиться $y(t)$ в бесконечности, в силу того, что $f(t) = \text{const}$;
$y_0(t)$ – свободная составляющая, которая определяет переходный процесс к принуждённой составляющей.

Из-за того, что $f(t) = \text{const}$, в бесконечности все производные f' будут стремиться к нулю, а значит от исходного уравнения (8.9) останется только:

$$a_n y = f(t) \tag{8.12}$$

Тогда:

$$y(\infty) = \frac{f(t)}{a_n} = \frac{\text{const}}{a_n} \tag{8.13}$$

Для нахождения $y_0(t)$ используется однородное уравнение от исходного (8.9):

$$y^{(n)} + a_1 y^{(n-1)} + \cdots + a_{n-1}y' + a_n y = 0 \tag{8.14}$$

В свою очередь, для нахождения решения однородного уравнения используется **характеристическое уравнение**:

$$p^n + a_1 p^{n-1} + \cdots + a_{n-1} p + a_n = 0 \tag{8.15}$$

Тогда решением $y_0(t)$ будет:

$$y_0(t) = \sum_{m=1}^{n} A_m e^{p_m t} \tag{8.16}$$

При квадратном характеристическом уравнении

В случае кратных корней $p_1 = p_2$:

$$y_0(t) = (A_1 + A_2 t) e^{p_{12} t} \tag{8.17}$$

В случае комплексных корней $p = -b \pm \omega t$

$$y_0(t) = A_0 \cdot e^{-bt} \cdot \sin(\omega t + \varphi) \tag{8.18}$$

Пример

$$y' + 5y = 10, \quad y(0) = 1$$

$$y(t) = y_0(t) + y(\infty)$$

$$y(\infty) = \frac{10}{5} = 2$$

$$\begin{aligned} y' + 5y &= 0 && \text{– однородное уравнение} \\ p + 5 &= 0 && \text{– характеристическое уравнение} \\ p &= -5, && \text{тогда решение } y_0(t): \\ y_0(t) &= A \cdot e^{-5t}, && \text{в свою очередь } y(t): \\ y(t) &= A \cdot e^{-5t} + 2 \end{aligned}$$

$$\begin{aligned} y(0) &= 1 \\ A \cdot e^0 + 2 &= 1 && \Rightarrow A = -1 \\ \text{Ответ:} \quad y(t) &= -e^{-5t} + 2 \end{aligned}$$

Проверка:

$$y(0) = -e^{-5 \cdot 0} + 2 = 1$$

$$\begin{aligned} (-e^{-5t} + 2)' + 5(-e^{-5t} + 2) &= \\ = 5e^{-5t} - 5e^{-5t} + 10 &= \\ = 10 & \end{aligned}$$

Пример

$$y'' + 5y' + 4y = 20, \quad y(0) = 1,\ y'(0) = 2$$

$$\text{При } t \to \infty: \quad 4y(\infty) = 20 \ \Rightarrow \ y(\infty) = 5$$

$$\begin{aligned} p^2 + 5t + 4 &= 0 \\ p_1 &= -4 \\ p_2 &= -1 \end{aligned}$$

$$y_0(t) = A_1 \cdot e^{-t} + A_2 \cdot e^{-4t}$$

$$y(t) = A_1 \cdot e^{-t} + A_2 \cdot e^{-4t} + 5 \qquad y'(t) = -A_1 \cdot e^{-t} - 4A_2 \cdot e^{-4t}$$
$$y(0) = 1 \qquad y'(0) = 2$$
$$A_1 \cdot e^0 + A_2 \cdot e^0 + 5 = 1 \qquad -A_1 \cdot e^{-t} - 4A_2 \cdot e^{-4t} = 2$$
$$A_1 + A_2 + 5 = 1 \qquad -A_1 - 4A_2 = 2$$

$$\begin{cases} A_1 + A_2 + 5 = 1 \\ -A_1 - 4A_2 = 2 \end{cases}$$

$$-3A_2 = -2 \;\Rightarrow\; A_2 = \frac{2}{3}$$
$$A_1 = -4 - \frac{2}{3} = -\frac{14}{3}$$
$$\text{Ответ:} \quad y(t) = -\frac{14}{3}e^{-t} + \frac{2}{3}e^{-4t} + 5$$

Пример

$$y'' + 10y' + 200y = 1000, \quad y(0) = 1,\ y'(0) = 1$$

$$y(t) = y_0(t) + y(\infty)$$

В бесконечности: $\qquad 0 + 0 + 200y(\infty) = 1000 \qquad \Rightarrow y(\infty) = \frac{1000}{200} = 5$

Характеристическое уравнение: $\qquad p^2 + 10p + 200 = 0$
$$p_{1,2} = -5 \pm 13.229i$$

Если решать, предположив, что $y_0(t) = A_1 e^{p_1 t} + A_2 e^{p_2 t}$:

$$\begin{aligned} e^{(-5\pm 13.229i)t} &= e^{-5t} \cdot e^{\pm 13.229it} \\ &= e^{-5t} \cdot (\cos(13.229t) \pm i\sin(13.229t)) \end{aligned}$$

$$\begin{aligned} A_1 e^{p_1 t} + A_2 e^{p_2 t} &= A_1 e^{-5t} \cdot (\cos(13.229t) - i\sin(13.229t)) \\ &+ A_2 e^{-5t} \cdot (\cos(13.229t) + i\sin(13.229t)) \\ &= e^{-5t}\left((A_1 + A_2)\cos(13.229t) + (-A_1 + A_2)i\sin(13.229t)\right) \end{aligned}$$

Таким образом решение с помощью такого $y_0(t)$ нецелесообразно, поэтому в случае комплексных корней в качестве $y_0(t)$ принято брать:

$$y_0(t) = A_0 e^{-bt} \cdot \sin(\omega t + \varphi)$$

$$y(t) = A_0 e^{-5t} \cdot \sin(13.229t + \varphi) + 5$$
$$y'(t) = -5A_0 e^{-5t} \cdot \sin(13.229t + \varphi) + A_0 e^{-5t} \cdot 13.229 \cdot \cos(13.229t + \varphi)$$

$$\begin{cases} A_0 \cdot \sin(\varphi) + 5 = 1 & \text{согласно } y(0) \\ -5A_0 \cdot \sin(\varphi) + 13.229A_0 \cdot \cos(\varphi) = 1 & \text{согласно } y'(0) \end{cases}$$

$$\begin{cases} A_0 \cdot \sin(\varphi) = -4 \\ -5\underbrace{A_0 \cdot \sin(\varphi)}_{-4} + 13.229A_0 \cdot \cos(\varphi) = 1 \end{cases}$$

$$\begin{cases} A_0 \cdot \sin(\varphi) = -4 \\ 20 + 13.229A_0 \cdot \cos(\varphi) = 1 \end{cases}$$

$$\begin{cases} A_0 \cdot \sin(\varphi) = -4 \\ A_0 \cdot \cos(\varphi) = {}^{-19}/_{13.229} = -1.436 \end{cases}$$

$$\operatorname{tg}(\varphi) = 2.785$$

$$\varphi = 1.226$$

$$A_0 = -\frac{4}{\sin(\varphi)} = -4.25$$

Ответ: $y(t) = 5 - 4.25e^{-5t}\sin(13.229t + 1.226)$

8.2.2 Операционный метод

Операционный метод основан на преобразовании Лапласа.

8.2.2.1 Свойства

$$L\{c\} = \frac{c}{p} \tag{8.19}$$

$$L\{cy(t)\} = cY(p) \tag{8.20}$$

$$L\{y'(t)\} = pY(p) - y(0) \tag{8.21}$$

$$L\{y''(t)\} = p^2Y(p) - y(0)p - y'(0) \tag{8.22}$$

$$L(p) = \int_0^\infty e^{pt} \cdot \left|y(t)\right| \, dt \tag{8.23}$$

Пример

$$y'' + 5y' + 4y = 20$$

$$p^2Y(p) - p - 2 + 5(pY(p) - 1) + 4Y(p) = \frac{20}{p}$$

$$p^2Y(p) - p - 2 + 5pY(p) - 5 + 4Y(p) = \frac{20}{p}$$

$$Y(p)\left(p^2 + 5p + 4\right) = \frac{20}{p} + p + 7$$

$$Y(p)\left(p^2 + 5p + 4\right) = \frac{p^2 + 7p + 20}{p}$$

$$Y(p) = \frac{p^2 + 7p + 20}{p(p^2 + 5p + 4)} = \frac{F_1(p)}{pF_2(p)}$$

$$Y(p) = \frac{p^2 + 7p + 20}{p(p+1)(p+4)}$$

$$Y(p) = \frac{A}{p} + \frac{B}{p+1} + \frac{C}{p+4}$$

$$y(t) = \frac{F_1(0)}{F_2(0)} + \sum_{i=1}^{n} \frac{F_1(p_i)}{p_i F_2'(p_i)} \cdot e^{p_i t} \tag{8.24}$$

$$F_1(p) = p^2 + 7p + 20$$
$$F_2(p) = 2p + 5$$

$$y(t) = 5 - \frac{14}{3}e^{-t} + \frac{2}{3}e^{-4t}$$

Пример

$$y'' + 10y' + 200y = 1000, \quad y(0) = 1,\ y'(0) = 1$$

$$\underbrace{p^2Y(p) - p - 1}_{y''} + \underbrace{10 \cdot (pY(p) - 1)}_{10y'} + \underbrace{200Y(p)}_{200y} = \underbrace{\frac{1000}{p}}_{1000}$$

$$Y(p)(p^2 + 10p + 200) = \frac{1000}{p} + p + 11$$

$$Y(p) = \frac{p^2 + 11p + 1000}{p(p^2 + 10p + 200)}$$

$$Y(p) = \frac{F_1(p)}{pF_2(p)}$$

$$y(t) = \frac{F_1(0)}{F_2(0)} + 2\mathrm{Re}\left\{\frac{F_1(p_1)}{pF_2'(p_1)}e^{p_1 t}\right\}$$

$$\frac{F_1(p_1)}{pF_2'(p_1)} = -2 + 0.718i = 2.125e^{2.8i}$$

???

$$y(t) = 5 + 2.25e^{-5t} \cdot \mathrm{Re}\left\{e^{13.229t + 2.8}\right\}$$
$$y(t) = 5 + 4.25e^{-5t}\cos(13.229t + 2.8)$$

8.2.3 Полуаналитические методы

Полуаналитические методы основаны на использовании рядов, в свою очередь, в которых, получаемая точность зависит от количества членов ряда.

8.2.3.1 Метод Пикара

Данный метод пригоден только для ОДУ первого порядка, и для функций f, от которых возможно вычислить интеграл **аналитически**.

В Методе Пикара обе части ОДУ интегрируются с помощью интеграла с переменным верхним пределом:

$$\underbrace{\int_0^t y'\,dt}_{y(t)-y(0)} = \int_0^t f(t,y)\,dt \tag{8.25}$$

Таким образом можно выразить $y(t)$:

$$y(t) = y(0) + \int_0^t f(t,y)\,dt \tag{8.26}$$

где: y из $f(t,y)$ на первом шаге берётся равным $y(0)$, а на каждом последующим – равным $y(t)$ полученным на предыдущем шаге.

Пример

$$y' = -3y + 2t^2, \quad y(0) = 13$$

$$\begin{aligned} y(t) &= 13 + \int_0^t \left(-3 \cdot 13 + 2t^2\right) dt \\ &= 13 - 39t + \frac{2}{3}t^3 \Big|_0^t \\ &= 13 - 39t + \frac{2}{3}t^3 \end{aligned}$$

$$\begin{aligned} y(t) &= 13 + \int_0^t \left(-3\left(13 - 39t + \frac{2}{3}t^3\right) + 2t^2\right) dt \\ &= 13 - 39t + \frac{117}{2}t^2 + \frac{2}{3}t^3 - \frac{1}{2}t^4 \end{aligned}$$

8.2.3.2 Метод последовательного дифференцирования

Данный метод лишён обоих ограничений метода Пикара, а именно, подходит для ОДУ любого порядка, а также нет необходимости в аналитическом вычислении интеграла.

Метод последовательного дифференцирования использует **многочлен Тейлора** для вычисления $y(t)$ с заданной точностью:

$$y(t) = y(0) + y'(0)t + \frac{y''(0)}{2!}t^2 + \cdots + \frac{y^{(n)}(0)}{n!}t^n \tag{8.27}$$

где:
- $y(0)$ – как правило задан в начальном условии (задачи Коши);
- y' – изначально дано (например, в ОДУ задачи Коши);
- y'', и последующие – можно найти путём вычисления производной от y'.

Пример

$$y' = -3y + 2t^2, \quad y(0) = 13$$

$$\begin{aligned} & & y'(0) &= -3y(0) + 2 \cdot 0^2 = -39 \\ y'' &= -3y' + 4t & y''(0) &= -3y'(0) + 4 \cdot 0 = 117 \\ y''' &= -3y'' + 4 & y'''(0) &= -3y''(0) + 4 = -347 \\ y^{(IV)} &= -3y''' & y^{(IV)}(0) &= -3y'''(0) = 1041 \end{aligned}$$

$$\text{В точности } t^4: \quad y(t) = 13 - 39t + \frac{117}{2}t^2 - \frac{347}{6}t^3 + \frac{1041}{24}t^4$$

8.2.4 Численные методы

В численных методах решения дифференциальных уравнений, подразумевается, что нужен численный результат, а не формула.

С помощью численных методов возможно решить любое ОДУ первого порядка и системы ОДУ первого порядка. А из этого следует, что с помощью данных методов возможно решить любое дифференциальное уравнение, т.к. любое уравнение возможно преобразовать в систему ОДУ первого порядка.

Таким образом численные методы:

- решают любое дифференциальное уравнение;
- дают решение в виде таблицы чисел (по которой возможно построить график);
- находят числовое значение на **прерывной** дискретной сетке.

8.2.4.1 Переход к численным методам

Для начала непрерывный параметр t дискретизируется с шагом h:

$$t \to t_k = k \cdot h, \quad k = \overline{0, n} \tag{8.28}$$

Далее левая и правая часть ОДУ интегрируется с помощью определённого интеграла от t_k до t_{k+1}:

$$\begin{gathered} y' = f(t, y) \\ \underbrace{\int_{t_k}^{t_{k+1}} y' \, dt}_{y_{k+1} - y_k} = \int_{t_k}^{t_{k+1}} f(t, y) \, dt \\ y_{k+1} - y_k = \int_{t_k}^{t_{k+1}} f(t, y) \, dt \end{gathered} \tag{8.29}$$

Универсальное основание численных методов

$$\begin{gathered} y_{k+1} = y_k + \int_{t_k}^{t_{k+1}} f(t, y) \, dt \\ y_{k+1} = y_k + \boldsymbol{S_k} \end{gathered} \tag{8.30}$$

где: S_k – площадь криволинейной трапеции $f(t, y)$ от t_k до t_{k+1}.

Получается, численный метод зависит от выбора метода численного интегрирования для расчёта S_k.

8.2.4.2 Классификация численных методов

Явные: следующее значение (y_{k+1}) однозначно вычисляется по предыдущему значению (y_k);
Неявные: для вычисления следующего значения (y_{k+1}) необходимо его уже знать.

Одношаговые: следующее значение рассчитывается по не более чем одному уже известному значению;
Многошаговые: метод, в котором для расчёта следующего значения используются несколько предыдущих значений.

Некоторые неявные методы можно свести к явным (в случае линейной зависимости), путём переноса неизвестных в левую часть. В случае, если это невозможно неявный численный метод являются итерационным, и для его расчёта можно использовать метод прогноз-коррекция.

Схема Прогноз-Коррекция

Прогноз-коррекция применяется в случае неявного метода.

Шаг «прогноз»: используется явный метод для расчёта изначального значения;
Шаг «коррекция»: используется неявный метод, в котором итерационно «корректируется» изначально найденное значение (до получения результата с требуемой точностью).

8.2.4.3 Метод Эйлера (явный, одношаговый)

В данном методе для расчёта S_k используется метод прямоугольников.

$$y_{k+1} = y_k + h \cdot f(t_k, y_k) \tag{8.31}$$

Пример

$$y' = -3y + 2t^2, \quad y(0) = 13$$

Пусть $t \in [0; 1]$, $h = 0.25$, тогда:

$$y_0 = 13$$
$$y_1 = 13 + 0.25f(0, 13) = 13 + 0.25(-3 \cdot 13 + 2 \cdot 0) = 13 - 9.75 = 3.25$$
$$y_2 = 3.25 + 0.25f(0.25, 3.25) = 3.25 + 0.25(-3 \cdot 3.25 + 2 \cdot 0.25^2) = 3.25 - 2.406 = 0.844$$
$$y_3 = 0.844 + 0.25f(0.50, 0.844) = 0.336$$
$$y_4 = 0.336 + 0.25f(0.75, 0.336) = 0.365$$

t	0.00	0.25	0.50	0.75	1.00
y	13	3.25	0.844	0.336	0.365
y реальное	13	6.15	2.96	1.54	1.01

Примечание

Таким образом метод Эйлера может потерять устойчивость, а также в нём **накапливается ошибка** (искажение информации).

8.2.4.4 Метод Эйлера-Коши (неявный, одношаговый)

В данном методе для подсчёта S_k используется метод трапеций.

$$y_{k+1} = y_k + \frac{h}{2}\left[f(t_k, y_k) + f(t_{k+1}, y_{k+1})\right] \tag{8.32}$$

8.2.4.5 Метод Адамса (явный, многошаговый)

В основе метода Адамса лежит квадратура Ньютона-Котеса 3-его порядка.

Для увеличения точности S_k строится интерполирующий полином 3-его порядка (например, методом Лагранжа) по 4-ём предыдущим точкам. В результате получается:

$$y_{k+1} = y_k + \frac{h}{24}\left[55f(t_k, y_k) - 59f(t_{k-1}, y_{k-1}) + 37f(t_{k-2}, y_{k-2}) - 9f(t_{k-3}, y_{k-3})\right] \tag{8.33}$$

8.2.4.6 Метод Башфорта (неявный, многошаговый)

$$y_{k+1} = y_k + \frac{h}{24}\left[9f(t_{k+1}, y_{k+1}) + 19f(t_k, y_k) - 5f(t_{k-1}, y_{k-1}) + f(t_{k-2}, y_{k-2})\right] \tag{8.34}$$

8.2.4.7 Метод Адамса-Башфорта (неявный, многошаговый)

Суть метода заключается в вычислении y_{k+1} с помощью метода Адамса (8.33), с дальнейшей подстановкой рассчитанного значения y_{k+1} в формулу (8.34) – метода Башфорта.

8.2.4.8 Методы Рунге-Кутты

Данные методы являются методами искусственного дробления шага, основная идея которых: для повышения точности интегрирования (расчёта S_k), вместо взятия предыдущих точек, генерируются новые точки внутри отрезка $[y_k; y_{k+1}]$.

Пример для IV порядка

$$
\begin{aligned}
f_1 &= f\left(t_k, y_k\right) && \text{левая точка} \\
f_2 &= f\left(t_k + \frac{h}{2}, y_k + \frac{h}{2} f_1\right) && \text{центральная точка (по методу Эйлера)} \\
f_3 &= f\left(t_k + \frac{h}{2}, y_k + \frac{h}{2} f_2\right) && \text{центральная точка (с коррекцией)} \\
f_4 &= f\left(t_{k+1}, y_k + h f_3\right) && \text{правая точка}
\end{aligned}
$$

Получается:

$$
y_{k+1} = y_k + \frac{h}{6}\left[f_1 + \underbrace{2f_2 + 2f_3}_{4 \cdot \frac{f_2 + f_3}{2}} + f_4\right] \tag{8.35}
$$

8.2.4.9 Методы Гира (неявные)

До сих пор описанные методы походят только для линейных, либо слабо нелинейных функций, которые теряют устойчивость для «жёстких» функций.

Суть методов Гира заключается в сочетании неявной схемы со сглаживающим фильтром.

Сглаживание по двум точкам

$$
y_{k+1} = \left[\frac{4}{3} y_k - \frac{1}{3} y_{k-1}\right] + \frac{2}{3} h \cdot f(t_{k+1}, y_{k+1}) \tag{8.36}
$$

Сглаживание по трём точкам

$$
y_{k+1} = \left[\frac{18}{11} y_k - \frac{9}{11} y_{k-1} + \frac{2}{11} y_{k-2}\right] + \frac{6}{11} h \cdot f(t_{k+1}, y_{k+1}) \tag{8.37}
$$

Можно заметить, что в случае не большого различия между точками y_k и предыдущими $(y_{k-1}, ...)$, сумма дробных коэффициентов равняется единице. Таким образом достигается «сглаживание».

8.2.4.10 Метод адаптивного интегрирования

В случае отсутствия эталонного значения, с которым можно было бы сравнить получаемые значения, существуют различные подходы, которые применяются в зависимости от обстоятельств. В данном случае используют два «родственных» метода для расчёта очередного значения, и в зависимости на сколько полученные значения отличаются друг от друга, уточняют полученное значение, либо рассчитывают следующее значение.

Данный метод предоставляет устойчивость и точность, взамен на скорость.

Алгоритм

1. Расчёт очередного значения y_k двумя методами, например:
   $y_{k_{(4)}}$ – методом Рунге-Кутты четвертого порядка;
   $y_{k_{(5)}}$ – методом Рунге-Кутты пятого порядка.
2. – Если $\Delta = \left|y_{k_{(4)}} - y_{k_{(5)}}\right| > \varepsilon$, то значит шаг от значения y_{k-1} слишком велик, и его следует уменьшить: $h_1 = {}^{h_0}/_2$, и произвести расчёт y_k заново;
   – Иначе, рассчитывается следующее значение.

Таким образом на выходе получаются пары значений: $[t, y]$, в силу неравномерности шага.

8.3 Решение систем дифференциальных уравнений

Любую систему дифференциальных уравнений можно преобразовать в систему ОДУ.

$$\begin{cases} y' = f_1(t,y,z), & y(0) \\ z' = f_2(t,y,z), & z(0) \end{cases} \quad \Leftrightarrow \quad \begin{cases} y' = z, & \text{тогда} \\ z' = f(t,y,z) & z(0) = y'(0) \end{cases}$$

Поэтому, для решения систем дифференциальных уравнений можно использовать все рассмотренные ранее методы. Причём, на каждом шаге, уравнения можно рассчитывать параллельно.

Пример в случае использования метода Эйлера

$$\begin{cases} y_{k+1} = y_k + hf_1(t_k, y_k, z_k) \\ z_{k+1} = z_k + hf_2(t_k, y_k, z_k) \end{cases} \tag{8.38}$$

8.4 Методы решения краевых задач

Определение 8.4.0.1:

Краевая задача – ОДУ второго порядка $y'' = f(t,y,y')$, в котором начальные условия (по сравнению с задачей Коши) заменены на краевые: $y(0), y(T)$.

$$y'' = f(t,y,y'), \qquad y(0) = y_0, y(T) = y_T \tag{8.39}$$

$$p(t)\cdot y'' + q(t)\cdot y' + r(t)\cdot y = f(t) \tag{8.40}$$

8.4.1 Численно-аналитические методы Галёркина

Главная идея – получить решение краевой задачи в виде приближённой аналитической функции. Для этого необходимо определить будущий вид функции $y(t)$:

$$y(t) = \varphi_0(t) + a_1\varphi_1(t) + \cdots + a_k\varphi_k(t) \tag{8.41}$$

где: $\varphi(t)$ – выбранные (известные) базисные (аналитические) функции.

Причём **базисные функции должны быть дифференцируемыми**, и удовлетворять следующим ограничениям:

$$\begin{aligned} \varphi_0(0) &= y_0 \\ \varphi_0(T) &= y_T \\ \varphi_i(0) = \varphi_i(T) &= 0 \ \forall i \in \overline{1,k} \end{aligned} \tag{8.42}$$

Таким образом между базисными функциями разделяют «обязанности»: за начальные условия ответственна только функция $\varphi_0(t)$.

Имея $y(t)$ (определён в (8.41)) возможно рассчитать $y'(t)$ и $y''(t)$, и подставить в исходное уравнение (8.40) краевой задачи, для расчёта невязки:

$$\begin{aligned} R(t,a_1,...,a_k) = p(t)\cdot &\left[\varphi_0''(t) + a_1\varphi_1''(t) + \cdots + a_k\varphi_k''(t)\right] \\ + q(t)\cdot &\left[\varphi_0'(t) + a_1\varphi_1'(t) + \cdots + a_k\varphi_k'(t)\right] \\ + r(t)\cdot &\left[\varphi_0(t) + a_1\varphi_1(t) + \cdots + a_k\varphi_k(t)\right] \\ - &f(t) \end{aligned} \tag{8.43}$$

Далее подбираются коэффициенты a_i, путём расчёта и минимизации невязки $R(t,a_1,...,a_k) \to \min$.

8.4.1.1 Минимизация R с помощью МНК (аппроксимация решения)

В силу непрерывности функции $y(t)$, напрямую МНК не применим, для этого функцию $y(t)$ искусственно дискретизируют. Тогда в качестве критерия минимизации используется

$$\sum_t R^2(t,a_1,...,a_k) \to \min \tag{8.44}$$

т.е. сумма квадратов отклонений по всем дискретизированным точкам t.

8.4.1.2 Метод коллокации (интерполяция решения)

Пусть приближенная функция $y(t)$ (определённая в (8.41)) равняется истинным значениям только в фиксированном числе точек, называемыми точками коллокации.

Количество точек коллокации выбирается в количестве k – равному количеству переменных a, тогда для каждого $i \in \overline{1,k}$ возможно записать по уравнению, таким образом получая СЛАУ:

$$R(t_i, a_1, ..., a_k) = 0 \, \forall i \in \overline{1,k} \tag{8.45}$$

$$\boldsymbol{AX} = \boldsymbol{B} \quad \Rightarrow \quad X = \begin{bmatrix} a_1 \\ \vdots \\ a_k \end{bmatrix} \tag{8.46}$$

8.4.2 Метод конечных разностей

Метод конечных разностей является численным методом, поэтому в качестве результата получается таблица чисел. Для этого функция дискретизируется с шагом h с помощью следующих обозначений:

$$y(t_k) = y_k \tag{8.47}$$

далее y' и y'' заменяются разностными схемами:

$$y'(t_k) \approx \frac{y_{k+1} - y_{k-1}}{2h} \tag{8.48}$$

$$y''(t_k) \approx \frac{y_{k+1} - 2y_k + y_{k-1}}{h^2} \tag{8.49}$$

Обозначенные y, y' и y'' можно подставить в исходное ОДУ (8.40), тогда для какого-то дискретного момента времени $k \in \overline{1, n-1}$ можно записать:

$$\frac{p(t_k)}{h^2}\left(y_{k+1} - 2y_k + y_{k-1}\right) + \frac{q(t_k)}{2h}\left(y_{k+1} - y_{k-1}\right) + r(t_k)y_k = f(t_k) \tag{8.50}$$

После можно сгруппировать коэффициенты при y_{k-1}, y_k и y_{k+1}:

$$\underbrace{\left(\frac{p(t_k)}{h^2} - \frac{q(t_k)}{2h}\right)}_{a_{k,k-1}} y_{k-1} + \underbrace{\left(r(t_k) - 2\frac{p(t_k)}{h^2}\right)}_{a_{k,k}} y_k + \underbrace{\left(\frac{p(t_k)}{h^2} + \frac{q(t_k)}{2h}\right)}_{a_{k,k+1}} y_{k+1} = \underbrace{f(t_k)}_{b_k} \tag{8.51}$$

Для каждого k от 1 до $n-1$ можно расписать уравнение. Причём y_0 и $y_n = y_T$ известны из начальных условий, поэтому их можно отнять от соответствующих правых частей b_1 и b_{n-1}, таким образом получается трёхдиагональная матрица:

$$\begin{bmatrix} \ddots & \ddots & 0 & 0 \\ \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & \ddots \\ 0 & 0 & \ddots & \ddots \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ \vdots \\ \vdots \\ y_{n-1} \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ \vdots \\ b_{n-1} \end{bmatrix} \tag{8.52}$$

Таким образом суть метода конечных разностей заключается в замене производных y в исходном уравнении на конечно разностные схемы, сводя задачу к решению СЛАУ с трёх диагональной матрицей.

Метод конечных разностей применим только для линейных задач, иначе не будет получатся трёхдиагональная матрица, которая решается за линейное время.

8.4.3 Метод «стрельбы»

Метод «стрельбы» является чисто численным методом, и подходит для любых ОДУ.

$$y'' = f(t, y, y'), \qquad y(0) = y_0, \quad y(T) = y_T$$

Если бы вместо изначального условия $y(T)$ в задаче было бы задано $y'(0)$, то это была бы задачей Коши, которую возможно решить любыми рассмотренными ранее методами.

Метод «стрельбы» основан на переходе к задаче Коши, т.е. отысканию такого начального условия $y'(0)$, при котором бы выполнялось условие $y(T) = y_T$.

Алгоритм

1. Выбор угла прицеливания $\alpha_1 \in [-90; +90]$. С помощью данного угла решается задача Коши, получая значение в точке T равным $y_T(\alpha_1)$:

$$y'(0, \alpha_1) = \mathrm{tg}(\alpha_1) \quad \xrightarrow{\text{Коши}} \quad y_T(\alpha_1) \tag{8.53}$$

2. Выбор второго угла прицеливания (пристрелка): $\alpha_2 = \alpha_1 + \Delta\alpha$. Опять решается задача Коши, получая значение в точке T равным $y_T(\alpha_2)$:

$$y'(0, \alpha_2) = \mathrm{tg}(\alpha_2) \quad \xrightarrow{\text{Коши}} \quad y_T(\alpha_2) \tag{8.54}$$

3. Вычисление чувствительности «выстрелов»:

$$\frac{\partial y_T}{\partial \alpha} \approx \frac{y_T(\alpha_2) - y_T(\alpha_1)}{\Delta\alpha} \tag{8.55}$$

4. Вычисление истинной коррекции:

$$\Delta\alpha^* = \frac{y_T - y_T(\alpha_1)}{\partial y_T / \partial \alpha} \tag{8.56}$$

5. Нахождение истинного угла прицеливания, и решение задачи Коши, в которой значение в точке T должно с численной точностью быть равным начальному условию y_T:

$$\alpha^* = \alpha_1 + \Delta\alpha^* \quad \rightarrow \quad y'(0, \alpha^*) = \mathrm{tg}(\alpha^*) \quad \xrightarrow{\text{Коши}} \quad y_T$$

Если точность после последнего шага не устраивает, «прицеливания» можно дальше уточнять, используя для шагов «вычисления чувствительности» и «вычисления истинной коррекции» последние два решения $\alpha_{k-1}, y_{T_{k-1}}$ и α_k, y_{T_k}.

Универсальность данного метода разменивается на его скорость (по сравнению со скоростью метода конечных разностей).

8.5 Методы решения дифференциальных уравнений в частных производных

8.5.1 Дифференциальные уравнения частных производных от двух переменных

Телеграфное уравнение

$$\frac{\partial^2 U}{\partial x^2} = \frac{1}{V_0^2} \cdot \frac{\partial^2 U}{\partial t^2} \quad \rightarrow \quad U(x, t) \tag{8.57}$$

Данное уравнение показывает движение импульса по проводу без искажений. В сокращённом виде:

$$U_{xx} = \frac{1}{V_0^2} U_{tt} \tag{8.58}$$

Чтобы уравнение решать численно, нужно свести хотя бы по одной переменной к ... .

8.5.1.1 Моделирование с помощью уравнений частных производных

Уравнение линейного переноса

$$\frac{\partial U}{\partial x} = -\frac{1}{V} \cdot \frac{\partial U}{\partial t} \tag{8.59}$$

В сокращённом виде

$$U_x = -\frac{1}{V} U_t \tag{8.60}$$

Начальные условия такого уравнения как правило изменение импульса по времени в начальной точке:

$$U(0, t) = U_0(t) \tag{8.61}$$

Линейный сдвиг

Исходное дифференцирование по расстоянию x можно заменить конечно разностной схемой, например правой разностью:

$$\frac{U_{i+1}(t) - U_i(t)}{\Delta x} \approx -\frac{1}{V} \cdot \frac{\partial U_i(t)}{\partial t} \tag{8.62}$$

тогда возможно выразить формулу $U(t)$ в новой точке $i+1$:

$$U_{i+1}(t) = U_i(t) - \frac{\Delta x}{V} \cdot \frac{\partial U_i(t)}{\partial t} \tag{8.63}$$

Полученное уравнение показывает задержку, которую потратила волна, двигаясь со скоростью V для прохождения участка длиной Δx. (???)

Затухание

$$U_x = -\gamma \cdot U \tag{8.64}$$

тогда

$$\begin{aligned} U_{i+1} &= U_i(t) - \gamma \cdot U_i(t) \\ U_{i+1} &= U_i(t)(1-\gamma) \end{aligned} \tag{8.65}$$

Дисперсия

Амплитуда уменьшается и при этом сам сигнал расширяется.

$$U_x = \beta U_{tt} \tag{8.66}$$

тогда

$$U_{i+1}(t) = U_i(t) + \beta \cdot U_{tt}$$

Пространственная дисперсия

Задний фронт становится резким, а передний – пологим.

$$U_x = \beta U_{ttt} \tag{8.67}$$

Нелинейность

$$U_x = \alpha U U_t \tag{8.68}$$

Нелинейность, на примере воды, отвечает за поверхностное натяжение: задний фронт становится пологим, а передний – «укручается» (вплоть до заворачивания, подобно выходу волны воды на берег).

Комбинация

Комбинируя данные эффекты можно рассчитывать изменения форм, моделировать турбулентность воздушных потоков, водяных потоков, и другие явления.

8.5.2 Метод конечных-разностей

Как и в других методах конечных-разностей (для других задач), главный приём – в самом уравнении производные заменяются на их приближённые конечно-разностные аналоги.

Данный метод подобен рассмотренным ранее одномерным методам, с отличием – двумерной дискретизации.

Пример для уравнения линейного переноса

Пусть необходимо решить уравнение:

$$U_x = -\frac{1}{V}U_t \tag{8.69}$$

с начальным условием (с каким-то шагом дискретизации Δt):

$$U(0,t) = U_0(t) \tag{8.70}$$

Следующий шаг – синтез сетки (дискретизация времени t и пространства x):

$$\Delta t = \text{const}, \qquad \Delta x = \text{const}$$
$$t_k = k \cdot \Delta t$$
$$x_i = i \cdot \Delta x$$
$$U(x_i, t_k) = U(i \cdot \Delta x, k \cdot \Delta t) = U_k^i$$

Строго правила для применения конечно разностных схем не существует, для каждой задачи выбирается наилучший метод, т.е. строится своя схема.

Пример составления разностной схемы правой и центральной разности:

$$\underbrace{\frac{U_k^{i+1} - U_k^i}{\Delta x}}_{\approx U_x} = -\frac{1}{V} \cdot \underbrace{\frac{U_{k+1}^i - U_{k-1}^i}{2\Delta t}}_{\approx U_t} \tag{8.71}$$

Далее можно выразить U_k^{i+1}:

$$U_k^{i+1} = U_k^i - \frac{\Delta x}{V} \cdot \frac{U_{k+1}^i - U_{k-1}^i}{2\Delta t} \tag{8.72}$$

Данная разностная схема является явной.

С помощью полученной формулы (т.е. разностной схемы), в случае задачи Дирихле ($U(x,0) = 0$), можно заполнить всю дискретную сетку имея только начальное условие ($U(0,t) = U_0(t)$).

Возможны и другие схемы, например двумерный аналог метода Рунге-Кутты – метод Кранка-Николсона.

Шаблон разностной схемы

Шаблон полученной схемы представляет собой T-образную форму. С помощью данной «формы» заполняется по очереди все i-слои, таким образом заполняя всю дискретную сетку.

Условие устойчивости разностной схемы Фон Неймана

Данное условие было выведено экспериментально.

Что-бы численное решение было устойчиво, шаг дискретизации по x следует выбирать не больше 0.01 произведения V на шаг дискретизации по времени:

$$\Delta x \leq 0.01 V \Delta t \tag{8.73}$$

где: V – оценочная скорость перемещения «волны».

8.5.3 Метод конечных элементов

Вся область нахождения решения разбивается на треугольники.

Для каждого треугольника (которые задаются координатами трёх вершин) записывается какой-то закон «сохранения», таким образом в дискретной форме приводит к описанию каждого треугольника одним линейным уравнением. Тогда сколько треугольников с сетке, столько получается уравнений в системе.

8.5.4 Метод граничных элементов

Данный метод предназначен для решения задач, когда в начальных условиях задачи заданы на некотором контуре. В данном методе используется кривоугольная сетка.

8.5.5 Морфологические методы

Пусть заданы несколько точек $U(t)$ по времени t, необходимо сделать прогноз (значение будущей точки).

Классические методы – аппроксимация или интерполяция.

Морфология – наука о форме (например, в данном случае интересует форма сигнала).

Если ориентируясь на дискретно-временной ряд удастся из заданных точек $U(t)$ «вытащить» зависимость:

$$U_{k+1} = f(U_k) \tag{8.74}$$

то знание функции f даёт идеальный компактный прогноз.

Если данная функция f справедлива на первом наборе точек $U(t)$ (на которых происходит обучение), то данный ряд возможно продолжить дальше.

8.5.5.1 Примеры применения

Упаковка, шифрование данных для передачи или хранения.

Например в GSM, вместо передачи полностью всей речи, передаётся некоторое стартовое значение и коэффициенты регрессии (например от 5 до 7). Таким образом передаётся 8 чисел вместо 1000. На приёмном конце берётся стартовое значение и с помощью коэффициентов регрессии распаковывается весь временной ряд, а далее воспроизводится как цифровой звуковой сигнал.

Если перенести это на плоскость, то возможно описать яркость одного пикселя через яркость пикселей его окружающих ближайших соседей. Таким образом появляется морфологический набор операций для обработки изображений.

Обработка изображений

Оптическое распознавание символов (OCR – optical character recognition) – перевод изображения с текстом в текстовые данные.

Базовые понятия

Инициатор – начальное состояние (изображение).

Генератор – правила, по которым каждый элемент изображения обрабатывается.

Маска – некоторое двоичное изображение (геометрическая форма). Он может быть произвольного размера и формы. Чаще всего используются симметричные элементы, как прямоугольник или круг. В каждом элементе выделяется особая точка, называемая начальной (origin). Она может быть расположена в любом месте элемента, хотя в симметричных масках это обычно центральный пиксель.

Алгоритм

В начале результирующая поверхность заполняется 0, образуя полностью чёрное изображение. Затем осуществляется зондирование или сканирование исходного изображения пиксель за пикселем с помощью маски. Для зондирования каждого пикселя на изображение «накладывается» маска так, чтобы совместились зондируемая и начальные точки. Затем проверяется некоторое условие на соответствие пикселей маске и точек изображения «под ним». Если условие выполняется, то на результирующем изображении в соответствующем месте ставится 1 (в некоторых случая будет добавляться не один единичный пиксель, а все единички из маски).

Морфологические операции

Дилатация (наращивание)	– заполнение «дырок».
Эрозия (сужение)	– удаление «шума».
Замыкание (склейка)	– дилатация, после которой производится эрозия – сглаживает контуры и заполняет промежутки.
Размыкание	– эрозия, после которой производится дилатация – сглаживает контуры ликвидирует узкие перешейки и выступы.

8.5.5.2 Клеточные автоматы

На основе перечисленных операций возможно моделировать колонии бактерий, поведение отдельных машин в транспортном потоке, и многое другое.

Определение 8.5.5.1:

Клеточный автомат – дискретная модель. Включает решётку ячеек, каждая из которых может находится в одном из конечного множества состояний.

Один из знаменитых клеточных автоматов – «Conway's Game of Life».