1) Представление чисел в ЭВМ и связанные с этим погрешности

Существуют два способа представления чисел в памяти ЭВМ. Они называются так: форма с фиксированной точкой и форма с плавающей точкой. Форма с фиксированной точкой применяется к целым числам, форма с плавающей точкой — к вещественным числам (целым и дробным). Под точкой здесь подразумевается знак-разделитель целой и дробной части числа.

Целые отрицательные числа :

Для представления целого отрицательного числа используется дополнительный код.

Получить дополнительный код можно следующим путем:

1. записать внутреннее представление положительного числа X;
2. записать обратный код этого числа заменой во всех разрядах 0 на 1 и 1 на 0;
3. к полученному числу прибавить 1.

Целые положительные числа записываются без изменений.

Вещественные числа в памяти компьютера представляются в форме с плавающей точкой.

Форма с плавающей точкой использует представление вещественного числа R в виде произведения мантиссы m на основание системы счисления p в некоторой целой степени n, которую называют порядком:

$$R = m * p^n$$

Получается, что представление числа в форме с плавающей точкой неоднозначно? Чтобы не было неоднозначности, в ЭВМ ***используют нормализованное представление числа в форме с плавающей точкой.*** Мантисса в нормализован-ном представлении должна удовлетворять условию:

$$0{,}1_p <= m < 1_p.$$

Иначе говоря, мантисса меньше единицы и первая значащая цифра — не ноль. Значит для рассмотренного числа нормализованным представлением будет: 0.25324 * 102. В разных типах ЭВМ применяются различные варианты представления чисел в форме с плавающей точкой. Для примера рассмотрим один из возможных. Пусть в памяти компьютера вещественное число представляется в форме с плавающей точкой в двоичной системе счисления (p=2) и занимает ячейку размером 4 байта. В ячейке должна содержаться следующая информация о числе: знак числа, порядок и значащие цифры мантиссы. Вот как эта информация располагается в ячейке:

± машинный порядок	М А	Н Т И С	С А
1-й байт	2-й байт	3-й байт	4-й байт

В старшем бите 1-го байта хранится знак числа. В этом разряде 0 обозначает плюс, 1 — минус. Оставшиеся 7 бит первого байта содержат машинный порядок. В следующих трех байтах хранятся значащие цифры мантиссы.

2) Обусловленность задач и связанная с этим устойчивость алгоритмов

Обусловленность задачи
-Корректность задачи
-Устойчивость метода (чувствительность к изменению входных данных)
-Сходимость (в итерационных методах)

Метод называется устойчивым если ему соответствует относительно малое изменение результатов.

3) Операции с комплексными числами и функциями

Операции с комплексными числами:

операции над комплексными числами отличаются от обычных тем, что к комплексным числам добавляется мнимая единица $i=\sqrt{-1}$.

Сложение и вычитание.

Сложение и вычитание двух комплексных чисел определяются совершенно естественно

, $$z_1 \pm z_2 = (x_1 \pm x_2) + i(y_1 \pm y_2),$$

то есть надо сложить (или вычесть) отдельно вещественные и мнимые части чисел.

Умножение.

Умножение двух комплексных чисел производится как умножение обычных чисел, надо лишь помнить, что :

$$i^2 = -1:$$

$$z_1 \cdot z_2 = (x_1 x_2 - y_1 y_2) + i(x_1 y_2 + x_2 y_1)$$

.

Деление.

Для деления комплексных чисел полезно запомнить следующее правило: чтобы разделить два комплексных числа друг на друга надо числитель и знаменатель умножить на число, комплексно сопряженное знаменателю. Тогда легко получить, что

.

$$\frac{z_1}{z_2} = \frac{x_1 x_2 + y_1 y_2}{x_2^2 + y_2^2} + i\,\frac{y_1 x_2 - x_1 y_2}{x_2^2 + y_2^2}.$$

4) Прямые методы решения систем линейных уравнений. Метод Гаусса

Метод Гаусса

Запишем систему $Ax=f,$ в развернутом виде

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1m}x_m = f_1,$$
$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2m}x_m = f_2,$$
$$\ldots\ldots\ldots \quad \ldots\ldots\ldots \quad \ldots\ldots\ldots \quad \ldots\ldots\ldots \quad \ldots\ldots\ldots \quad \ldots\ldots$$
$$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mm}x_m = f_m.$$

Метод Гаусса состоит в последовательном исключении неизвестных из этой системы. Предположим, что $a_{11} \neq 0$. Последовательно умножая первое уравнение на $-\frac{a_{i1}}{a_{11}}$ и складывая с i-м уравнение, исключим x_1 из всех уравнений кроме первого. Получим систему

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1m}x_m = f_1,$$
$$a_{22}^{(1)}x_2 + \ldots + a_{2m}^{(1)}x_m = f_2^{(1)},$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$a_{m2}^{(1)}x_2 + \ldots + a_{mm}^{(1)}x_m = f_m^{(1)},$$

где $a_{ij}^{(1)} = a_{ij} - \frac{a_{i1}a_{1j}}{a_{11}}, \quad f_i^{(1)} = f_i - \frac{a_{i1}f_1}{a_{11}}, \quad i,j = 2,3\ldots$

Аналогичным образом из полученной системы исключим x_2. Последовательно, исключая все неизвестные, получим систему треугольного вида

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1k}x_k + \ldots + a_{1m}x_m = f_1,$$
$$a_{22}^{(1)}x_2 + \ldots + a_{2k}^{(1)}x_k + \ldots + a_{2m}^{(1)}x_m = f_2^{(1)},$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$a_{m-1,m-1}^{(m-1)}x_{m-1} + a_{m-1,m}^{(m-1)}x_m = f_{m-1}^{(m-1)},$$
$$a_{m,m}^{(m-1)}x_m = f_m^{(m-1)}.$$

Описанная процедура называется прямым ходом метода Гаусса. Заметим, что ее выполнение было возможно при условии, что все $a_{i,i}^{(l)}$, $l=1,2\ldots m-1$ не равны нулю.

Выполняя последовательные подстановки в последней системе, (начиная с последнего уравнения) можно получить все значения неизвестных.

$$x_m = \frac{f_m^{(m-1)}}{a_{m,m}^{(m-1)}}, \qquad x_i = \frac{1}{a_{i,i}^{(i-1)}}\left(f_i^{(i-1)} - \sum_{j=i-1}^{m} a_{ij}^{(i-1)} x_j\right).$$

Эта процедура получила название обратный ход метода Гаусса..

Метод Гаусса может быть легко реализован на компьютере. При выполнении вычислений, как правило, не интересуют промежуточные значения матрицы *A*. Поэтому численная реализация метода сводится к преобразованию элементов массива размерности *(m×(m+1))*, где *m+1* столбец содержит элементы правой части системы.

Для контроля ошибки реализации метода используются так называемые контрольные суммы. Схема контроля основывается на следующем очевидном положении. Увеличение значения всех неизвестных на единицу равносильно замене данной системы контрольной системой, в которой свободные члены равны суммам всех коэффициентов соответствующей строки. Создадим дополнительный столбец, хранящий сумму элементов матрицы по строкам. На каждом шаге реализации прямого хода метода Гаусса будем выполнять преобразования и над элементами этого столбца, и сравнивать их значение с суммой по строке преобразованной матрицы. В случае не совпадения значений счет прерывается.

Один из основных недостатков метода Гаусса связан с тем, что при его реализации накапливается вычислительная погрешность. В книге [Самарский , Гулин] показано, что для больших систем порядка *m* число действий умножений и делений близко к $m^3/3$.

Для того, чтобы уменьшить рост вычислительной погрешности применяются различные модификации метода Гаусса. Например, метод Гаусса с выбором главного элемента по столбцам, в этом случае на каждом этапе прямого хода строки матрицы переставляются таким образом, чтобы диагональный угловой элемент был максимальным. При исключении соответствующего неизвестного из других строк деление будет производиться на наибольший из возможных коэффициентов и следовательно относительная погрешность будет наименьшей.

Существует метод Гаусса с выбором главного элемента по всей матрице. В этом случае переставляются не только строки, но и столбцы. Использование модификаций метода Гаусса приводит к усложнению алгоритма увеличению числа операций и соответственно к росту времени счета. Поэтому целесообразность выбора того или иного метода определяется непосредственно программисто.

Выполняемые в методе Гаусса преобразования прямого хода, приведшие матрицу ***A*** системы к треугольному виду позволяют вычислить определитель матрицы

$$\det A = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ 0 & a_{22}^{(1)} & \dots & a_{2m}^{(1)} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & a_{m,m}^{(m-1)} \end{vmatrix} = a_{11} \cdot a_{22}^{(1)} \dots a_{m,m}^{(m-1)}.$$

Метод Гаусса позволяет найти обратную матрицу. Для этого необходимо решить матричное уравнение

$$A \cdot X = E,$$

где *E*– единичная матрица. Его решение сводится к решению *m* систем

$$A\bar{x}^{(j)} = \bar{\delta}^{(j)}, \qquad j = 1,2,..., m,$$

у вектора $\bar{\delta}^{(j)}$ *j* –я компонента равна единице, а остальные компоненты равны нулю

5.Обусловленность матрицы и точность решения системы.

Числом (или мерой) обусловленности матрицы называют величину равную произведению норм матрицы A и ее обратной и обозначают cond(A) . Матрицы с большим числом обусловленности называются плохо обусловленными и, наоборот, матрицы с малым значением называются хорошо обусловленными. Для норм справедливо Cond(A) ≥ 1 .

Cond(A)=||A|| * $||A^{-1}||$

$$\frac{||\Delta X||}{||X||} \leq Cond(A) * \frac{||\Delta B||}{||B||}$$

Где:

B - вектор правой части уравнений;

ΔB - вектор правой части уравнений с внесенной ошибкой;

X – вектор ответов;

ΔX – вектор ответов, полученных при внесении ошибки.

Нормы каждого из векторов считаются по одной из следующих формул:

Норма Эвклида:

$$||X|| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

Манхеттенская норма:

$$||X|| = |x_1| + |x_2| + \cdots + |x_n|$$

6.Итерационные методы решения систем линейных уравнений.

Метод последовательных приближений.

1)AX=B приведение системы к виду, удобному для итераций. $X = \alpha X + \beta$

Выражаем X в каждом уравнении:

$$x_1 = \frac{b_1}{a_{11}} - \frac{a_{12}}{a_{11}} x_2 - \frac{a_{13}}{a_{11}} x_3$$

$$x_2 = \frac{b_2}{a_{22}} - \frac{a_{21}}{a_{22}} x_2 - \frac{a_{23}}{a_{22}} x_3$$

$$x_3 = \frac{b_3}{a_{33}} - \frac{a_{31}}{a_{33}} x_2 - \frac{a_{32}}{a_{33}} x_3$$

$$\begin{array}{ccc} | & | & | \\ \beta & \alpha & \alpha \end{array}$$

Матрица:

$$h = \begin{matrix} 0 & \alpha_{12} & \alpha_{13} \\ \alpha_{21} & 0 & \alpha_{23} \\ \alpha_{31} & \alpha_{32} & 0 \end{matrix}$$

2)Выбрать начальное приближение $X(0) = \begin{matrix} 0 \\ \vdots \\ 0 \end{matrix}$

3)$X_{(k+1)} = \alpha X_{(k)} + \beta$

4) if $\max|Xi_{(k+1)} - Xi_{(k)}| < \varepsilon$, then stop

Else goto (3)

Где ε –точность.

На каждой итерации считается весь вектор X. То есть сразу x1 x2 x3. И потом все три значения используются для вычисления нового x1 x2 x3.

Метод Гаусса-Зейделя.

Новое вычисляем через старое.

Дана система:

$$\begin{cases} 5x_1 + 3x_2 = 8 \\ x_1 - 4x_2 = -3 \end{cases}$$

Преобразовываем так, чтобы выделить иксы в левой части:

$$\begin{cases} x_1 = 1.6 - 0.6x_2 \\ x_2 = 0.75 + 0.25x_1 \end{cases}$$
Задаем точность ε.
$$X(0) = \begin{matrix} 0 \\ 0 \end{matrix}$$

Решение:
1)Подставляем нули в иксы правой части, получаем новые значения для x_1 и x_2:
$$X(1) = \begin{matrix} 1.6 \\ 0.75 \end{matrix}$$
$$\Delta Xmax = 1.6$$
2)Подставляем полученные иксы:
$$\begin{cases} x_1 = 1.6 - 0.6 * 0.75 \\ x_2 = 0.75 + 0.25 * 1.6 \end{cases}$$
Получаем:
$$X(2) = \begin{matrix} 1.15 \\ 1.15 \end{matrix}$$
$$\Delta Xmax = 0.45$$
3)Продолжаем до получения необходимой точности.
$\Delta Xmax$ – это max($ax_1, bx_2, \dots, zx_n$) т.е. на шаге (2) это max($0.6 * 0.75, 0.25 * 1.6$)

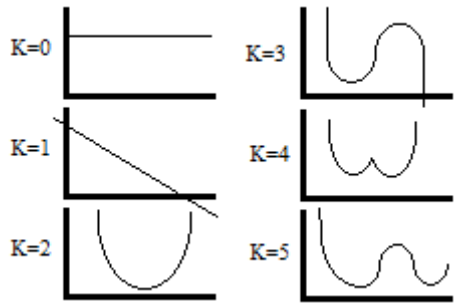
На каждой итерации считается только X1 или X2 или X3. Получается, что каждый раз они заного пересчитываются, чтобы ускорить сходимость. Считается только x1. Затем он подставляется в след решение, находим x2, затем подставляет x1 и x2 для нахождения x3 и затем подставляем сразу x1 x2 и x3.

Метод релаксаций.
$$X_{(k+1)} = X_{(k)} + \omega\ (\tilde{X}_{(k+1)} - X_{(k)})$$
$$0 < \omega < 2$$
Где:
ω - коэффициент релаксаций, подбираемый экспертно от 0 до 2. Нижняя релаксация: 0-1, верхняя релаксация: 1-2;
$\tilde{X}_{(k+1)}$ - вычисляется методом Гаусса-Зейделя.
Метод позволяет регулировать в ручную, увеличить или уменьшить скорость приближения к ответу. Сначала знаем разницу от идеального решения до X. Ищем, чтобы эта разница была 0.

7.Интерполяция и аппроксимация. Интерполяция по Лагранжу.

Интерполяция – нахождение промежуточный точек между заданными.
Интерполяция глобальная, если обрабатываются сразу все точки таблицы.
Интерполяция локальна, если берется несколько точек и восстанавливается некоторый интервал – хорошая скорость вычислений.

Аппроксимация – нахождение аналитического выражения функции по таблице точек, находящегося наиболее близко к точкам.
$$\varphi(x) = \sum_{k=0}^{n} a_k * f_k(x)$$
Где:
a_k - неизвестные коэффициенты;
$f_k(x)$ - базис интерполяции/аппроксимации.
$\varphi(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$
Полином на единицу меньше, чем количество точек.



Интерполяция Лагранжа.

$L_n = y_0 * l_0(x) + y_1 * l_1(x) + \cdots + y_n * l_n(x)$

Локальные полиномы $l_i(x_j) = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases}$

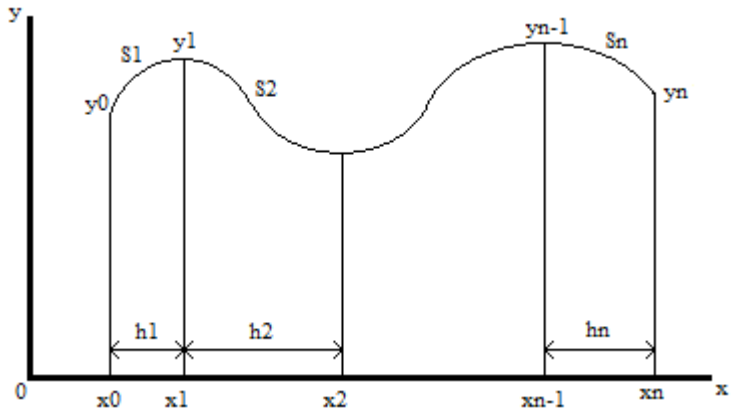
$$l_i(x) = \frac{(x - x_0)(x - x_1) \dots (x - x_{i-1})(\downarrow)(x - x_{i+1}) \dots (x - x_n)}{(x_i - x_0)(x_i - x_1) \dots (x_i - x_{i-1})(\uparrow)(x_i - x_{i+1}) \dots (x_i - x_n)}$$

$(\downarrow)$ и $(\uparrow)$ - отсутствует x_i.

От себя: пример приводить не буду – все делали дз по Лагранжу.

8.Сплайн-интерполяция.

Сплайн – отдельный участок кривой между точками. Их задача – обеспечить максимальную гладкость кривой.



h_i - зона ответственности сплайна S_i.

$h_i = x_i - x_{i-1}$

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

Где:

x – свободная переменная (любое значение);

x_{i-1} - левая граница сплайна;

a, b, c, d – неизвестные данной задачи.

Требования интерполяции:

1)$S_i(x_{i-1}) = y_{i-1} => a_i = y_{i-1}$ - n уравнений

2)$S_i(x_i) = y_i => a_i + b_i h_i + c_i {h_i}^2 + d_i {h_i}^3 = y_i$ - n уравнений

3)$S'_i(x_i) = S'_{i+1}(x_i) \rightarrow$ получаем гладкое сцепление сплайнов =>

$S'_i(x) = b_i * c_i(x + x_{i-1}) + 3d_i(x + x_{i-1})^2 => b_i + 2c_i + 3d_i {h_i}^2 = b_{i+1}$ - (n-1) уравнение

4) $S''_i(x_i) = S''_{i+1}(x_i) => S''_i(x_i) = 2c_i + 6d_i(x - x_{i-1}) => c_i + 3d_i h_i = c_{i+1}$ - (n-1) уравнение

5)Суть – за границами сплайны идут прямо.

$$S''_1(x_0) = S''_n(x_n) = 0 \Rightarrow\ c_1 = 0, c_n + 3d_n h_n = 0$$

Упрощение:
Из (1) a_i -> (2)
Из (4) $d_i = \frac{c_{i+1} - c_i}{3h_i}$ -> в (2) и (3)
Из (2) b_i -> в (3)
Результат: базовое правило составления системы.

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

$$\begin{array}{lccccc} i = 2 & \ddots & \ddots & 0 & c_2 & \beta_2 \\ \vdots & \ddots & \ddots & \ddots & \vdots & = \ \vdots \\ i = n & 0 & \ddots & \ddots & c_n & \beta_n \end{array}$$

Cond(a)->1

$$d_i = \frac{c_{i-1} - c_i}{3h_i}, \qquad d_n = \frac{-c_n}{3h_n}$$

$$b_i = (y_i - a_i - c_i h_i^{\,2} - d_i h_i^{\,3})/h_i$$

Ответ формируется в виде таблицы: Номер сплайна на неизвестные a_i, b_i, c_i, d_i.

Берем первые 3 точки. По ним строим функцию. Берем следующие 3. Опять строим. И так далее.

9. Аппроксимация по методу МНК

Аппроксимация по методу наименьших квадратов

Алгебраическая аппроксимация (приближение) функции многочленом *k*-го порядка ***(k < (n + 1)***, где ***n + 1*** - количество точек таблицы) имеет общий вид:

$$\phi_k(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_k x^k,$$

где коэффициенты многочлена a_i, ***(i=0, 1,..., k)*** находятся из условия минимизации функционала квадрата ошибки:

$$S(x, a_0, a_1, \ldots a_k) = \sum_{i=0}^{n} (\varphi_k(x_i) - f(x_i))^2 = \sum_{i=0}^{n} (a_0 + a_1 x + a_2 x^2 + \ldots + a_k x^k - f(x_i))^2$$

Требование равенства *0* частных производных функционала $S(x, a_0, a_1, \ldots, a_k)$

$$\begin{cases} \dfrac{dS}{da_0} = 0 \\ \ldots\ldots\ldots \\ \dfrac{dS}{da_k} = 0 \end{cases}$$

дает систему линейных уравнений порядка ***(k+1)***:

$$\begin{cases} a_0(n+1) + a_1\sum_{i=0}^{n} x_i + a_2\sum_{i=0}^{n} x_i^{\ 2} + \ldots + a_k\sum_{i=0}^{n} x_i^{\ k} = \sum_{i=0}^{n} f(x_i) \\ a_0\sum_{i=0}^{n} x_i + a_1\sum_{i=0}^{n} x_i^{\ 2} + a_2\sum_{i=0}^{n} x_i^3 + \ldots + a_k\sum_{i=0}^{n} x_i^{k+1} = \sum_{i=0}^{n} x_i * f(x_i) \\ a_0\sum_{i=0}^{n} x_i^{\ 2} + a_1\sum_{i=0}^{n} x_i^{\ 3} + a_2\sum_{i=0}^{n} x_i^4 + \ldots + a_k\sum_{i=0}^{n} x_i^{k+2} = \sum_{i=0}^{n} x_i^{\ 2} * f(x_i) \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_0\sum_{i=0}^{n} x_i^{\ k} + a_1\sum_{i=0}^{n} x_i^{\ k+1} + a_2\sum_{i=0}^{n} x_i^{k+2} + \ldots + a_k\sum_{i=0}^{n} x_i^{2k} = \sum_{i=0}^{n} x_i^{\ k} * f(x_i) \end{cases}$$

решение которой, например методом исключения Гаусса, даст искомые коэффициенты аппроксимации по методу наименьших квадратов. Иными словами, найденный аппроксимирующий многочлен $\phi_k(x)$ будет из всех многочленов *k*-го порядка проходить наиболее близко к заданным значениям функции (табличным точкам). *Отметим, что если* ***k=n****, то аппроксимирующий многочлен автоматически превращается в его частный случай - интерполирующий многочлен.*

10. Аппроксимация в базисе взаимно ортогональных функций. Аппроксимация Фурье, Чебышева, Лежандра, Лаггера

Аппроксимация в базисе ортогональных функций

Ортогональными называются функции, интеграл которых, на отрезке от а до б равен 0.

Главная проблема полиномиальной аппроксимации по методу наименьших квадратов заключается в том, что при высоких порядках аппроксимации (***k>10***) главная матрица системы линейных уравнений ***A*** становится плохо обусловленной, т.е. $\mathbf{Cond}(A) \to \infty$ и решение ее содержит большие погрешности.

Альтернативный вариант - использование в качестве базиса аппроксимации *ортогональных функций*, таких, что

$$\int_a^b f_m(x) * f_i(x)dx = 0, если\ , m \neq j$$

или в дискретной форме по всем (***n+1***) заданным точкам (число точек - четное)

$$\sum \left[f_m(x_i) * f_j(x_i) \ \right] = 0,,,,, еccл\ , m \neq j$$

Свойством ортогональности обладают многие функции: тригонометрические ряды, полиномы Чебышева, Бесселя, Лежандра, Лаггера и др. Если выбрать в качестве базиса аппроксимации набор ортогональных функций $f_m(x),\ m = 0, 1, 2, \ldots, k-1$

$$\phi_k(x) = \sum_{m=0}^{k} a^m * f_m(x),$$

то система линейных уравнений, составленная по методу наименьших квадратов, с учетом ортогональности превращается в систему несвязанных уравнений с диагональной матрицей ($\mathbf{Cond}(A) \to 1$)

$$a_m * \sum_{i=0}^{n} f^2{}_m(x_i) = \sum_{i=0}^{n} (y_i * f_m(x_i)),, m = 01,2,..., k-1$$

Которая решается относительно неизвестных коэффициентов аппроксимации a_m достаточно просто. Если в качестве базисных функций использовались ортогональные полиномы (многочлены), то через приведение слагаемых при одинаковых степенях результат аппроксимации можно представить в виде обычной степенной функции:

$\phi_k(x) = c_0 + c_1 \cdot x + c_2 \cdot x^2 + ... + c_k \cdot x^k$.

Ортогональность – на плоскости это перпендикуляр.

1) Аппроксимация Фурье (ДПФ) – дискретное преобразование Фурье

Их плюс в том, что, например если точек Н, но в МНК будет матрица Н на Н. И будут проблемы с точностью. При 10 на 10 и больше, матрица будет Вандер Морта, то есть коэффиценты будут слишком крупными, так как Х в степени 10 (или 20). Благодаря ортогональности получается матрица, где по главной идут числа, а остальные элементы 0.

2) Полиномы Чебышева

$$T_0(x) = 1, T_1(x) = x, T_{k+1}(x) = 2xT_k(x) - 1$$

3) Полиномы Лежандра

$$P_0 = 1, P_1 = x, P_{k+1} = \frac{(2k+1)x * P_k - P_{k-1}}{k+1}$$

4) Полином Лаггера

$$L_0, L_1 = 1 - x, L_{k+1} = \frac{(2k+1-x)L_k - kL_{k-1}}{k+1}$$

11. Аппроксимация Безье. Полиномы Бернштейна. Итерационный алгоритм.

Используется для аппроксимации кривых. Принцип параметрических функций.

Аппроксимация Безье

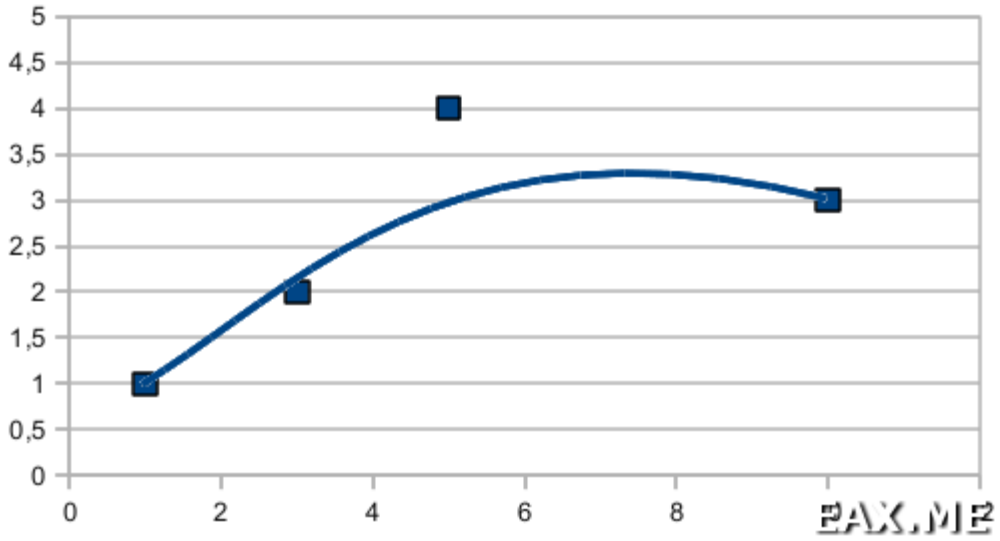
Представьте себе такую задачу. Есть N точек на плоскости. Требуется провести в непосредственной близости от них гладкую кривую. Точки эти могут представлять собой экспериментальные данные на графике или координаты, по которым пользователь кликнул мышкой в графическом редакторе. Так вот, одно из решений этой задачи заключается в использовании кривых Безье.

Кривая Безье для N точек задается полином степени (N-1). Вот как выглядит этот полином для для N = 4:

$B(t) = (1 — t)^3 P_0 + 3\,t\,(1 — t)^2 P_1 + 3\,t^2(1 — t)\,P_2 + t^3 P_3,\ t \in [0;1]$

Здесь P_i — это наши точки, t — параметр функции, меняя значение которого в диапазоне от 0 до 1, можно получить координаты всех точек кривой Безье. Общий вид функции для произвольного числа точек и очень красивую иллюстрацию к ней можно посмотреть в Википедии.

Приведенную формулу несложно проверить в Excel или OpenOffice:



Хорошо, с четырьмя точками понятно. А что делать, если точек 10 или 10 000? Кривую Безье можно построить для любого числа точек, но вычислять полиномы десятитысячной степени — это мягко говоря грустно. Делают очень просто — разбивают точки на группы по 4 штуки, строят для каждой из них кривую Безье и соединяют полученные сегменты в одну кривую.

Единственная проблема — полученная кривая будет «не очень гладкой» на границах сегментов. Спрашивается — что делать? И снова все оказывается очень просто (хотя ответ в сети хрен найдешь). Допустим, у нас есть шесть точек. Пусть $(x_3; y_3)$ и $(x_4; y_4)$ — координаты третьей и четвертой точки соответственно. Вставляем между ними дополнительную точку с координатами $x' = (x_3 + x_4)/2$ и $y' = (y_3 + y_4)/2$, после чего проводим одну кривую через точки 1-4, а вторую — через точки 4-7.

В результате получим одну гладкую кривую для шести точек. Если учесть поведение кривой Безье и то, что точки $(x_3; y_3)$, $(x'; y')$ и $(x_4; y_4)$ лежат на одной прямой, это становится вполне очевидно. Собственно полученная кривая и есть сплайн. Если точек больше шести, их нужно разбить по схеме 3-2-2…2-2-3 и связать полученные группы с помощью дополнительных точек, как описано выше. Последнюю точку можно повторить несколько раз, если множество точек не делится на целое число групп.

$x^2 + y^2 = R^2 - окружность \rightarrow y = \pm\sqrt{R^2 - x^2}$

$t = 0 : 0.01 : 2\Pi$

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

Параметрическое задание функции ^

t – внутренняя или рабочая переменная

$$\begin{cases} \beta_1(t) = \sum_{i=0}^{n} x_i * \phi_{i,n}(t) \\ \beta_1(t)_y = \sum_{i=0}^{n} y_i * \phi_{i,n}(t) \end{cases}$$
$$\phi_{i,n}(t) = c_n^i * t^i * (1-t)^{n-i}$$

$\phi_{i,n}(t)$ **- Полином бернштайна**

Итерационный алгоритм

Стационарный итерационный метод — это метод, который может быть представлен в следующей простой форме:

$x^k = Bx^{k-1} + c,$

где *B* и *c* не зависят от номера итерации *k*.

Стационарные итерационные методы

- метод Якоби
- метод Гаусса-Зейделя

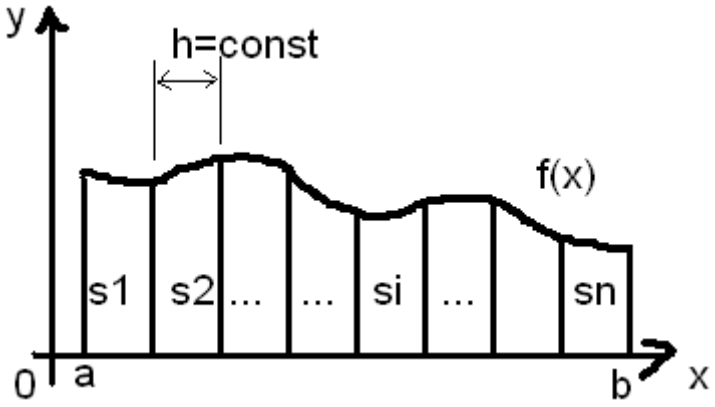
12. Численное дифференцирование. Конечные разности.

Наиболее известны численные методы, называемые разностными:

$$\frac{dy}{dx} \approx \frac{y_k - y_{k-1}}{h} - (левая\_разность)$$
$$\frac{dy}{dx} \approx \frac{y_{k+1} - y_k}{h} - (правая\_разность)$$
$$\frac{dy}{dx} \approx \frac{y_{k+1} - y_{k-1}}{2*h} - (центральная\_разность)$$

Разностные схемы отличаются порядком точности, т.е. многочлен какого порядка будет данным методом продифференцирован без ошибки. Для левой и правой разности - первый порядок. Для центральной разности - второй, она точнее. Однако, левая разность не вычисляет производную в первой точке ***(k=0)***, правая разность - в последней ***(k=n)***, а центральная - в обеих точках.
Они находят дифференциал. Ищется разница между двумя точками. Это дельта X. Строим как бы другой график на дельтах, это и есть производная.

13. Методы Ньютона-Котеса для численного интегрирования.



Вычисление интеграла функции f(x) сводится к вычислению суммы площадей элементарных кусков (располагающихся под функцией f(x)).

$$I = \int_a^b f(x)dx \approx \sum_{i=1}^{n} s_i$$

Что так же можно записать с помощью рекурсии:

$F(x_0) = 0$

$F(x_{i+1}) = F(x_i) + s_i$

Квадратурные формулы Ньютона-Котеса:

0-ой порядок: $s_i = h \cdot y_{i-1}$

1-ый порядок: $s_i = h \cdot \frac{(y_{i-1} + y_i)}{2}$

2-ой порядок: $s_i = h \cdot \frac{(y_{i-1} + 4y_i + y_{i+1})}{6}$

3-ий порядок: $s_i = \frac{(y_{i-1} + 3y_i + 3y_{i+1} + y_{i+2})}{8} \cdot h$

Примечание: Граковский на лекции записывал следующую формулу:

$s_i = \frac{3}{8} \cdot h \cdot (y_{i-1} + 3y_i + 3y_{i+1} + y_{i+2})$, но в этом случае длина s_i составляет 3h.

14. Методы решения нелинейных уравнений. Методы бисекции и «золотого сечения», секущих(хорд) и парабол.

Общие принципы работы:

1) 1 реализация -> 1 решение

2) поиск интервала с 1 решением при условии существования этого решения $f(a) \cdot f(b) \le 0$

3) Методы – итерационный, работают до заданной точности: $|x_{k+1} - x_k| \le \varepsilon$

4) Скорость метода: $|x_{k+1} - x_k| < |x_k - x_{k-1}|$

$|x_{k+1} - x_k| = c \cdot |x_k - x_{k-1}|^p$, где 0<c<1, p – порядок сходимости

p=1 – линейная сходимость

p=2 – квадратичная сходимость

5) 2 группы: интервальные и точечные.

Метод бисекции:

1. Выбор интервала с корнем: $[a,b] \rightarrow f(a) \cdot f(b) \le 0$
2. $x = \frac{a+b}{2}$, $f(x)$

3. Если $f(a)\cdot f(x)\le 0$, то сдвигаем правую границу b=x,
   иначе сдвигаем левую границу a=x
4. Повторяем до заданной точности:
   Если $(b-a)\le\varepsilon$, то останавливаемся – заданная точность достигнута,
   иначе продолжаем, переходя к пункту (2).
   Ищемь корень – X. Для этого делим пополам, отбрасываем часть и т.д

Метод «золотого сечения»:

Золотое сечение: $\frac{m}{n}=\frac{n}{m+n}$

Отсюда получаем пропорции: примерно 0.382 и 0.618

1. Выбор интервала с корнем: $[a,b]\to f(a)\cdot f(b)\le 0$
2. Если $|f(a)|\succ|f(b)|$, то x=a+0.618*(b-a)
   иначе x=a+0.382*(b-a)
3. Если $f(a)\cdot f(x)\le 0$, то сдвигаем правую границу b=x,
   иначе сдвигаем левую границу a=x
4. Повторяем до заданной точности:
   Если $(b-a)\le\varepsilon$, то останавливаемся – заданная точность достигнута,
   иначе продолжаем, переходя к пункту (2).
   Ищем x, для этого делим не поровну, а по пропорции.

Метод хорд(секущих):

1. Выбор интервала с корнем: $[a,b]\to f(a)\cdot f(b)\le 0$
2. $x=b-\frac{b-a}{f(b)-f(a)}f(b)$
3. Если $f(a)\cdot f(x)\le 0$, то сдвигаем правую границу b=x,
   иначе сдвигаем левую границу a=x
4. Повторяем до заданной точности:
   Если $\Delta(b-a)\le\varepsilon$, то останавливаемся – заданная точность достигнута,
   иначе продолжаем, переходя к пункту (2).
   $\Delta(b-a)=\left((b-a)_{k+1}-(b-a)_k\right)$

   Строится хорда, прямая через а и б. Там где она пересекает ось x будет следующая точка. Потом считаем, что отбрасываем а или б и новая точка примет букву отброшенной.

Метод парабол:

1. Выбор интервала с корнем: $[a,b]\to f(a)\cdot f(b)\le 0$
2. $x_0=\frac{a+b}{2}$, $f(x_0)$
3. Получаем таблицу:

x	A	x	b
y	f(a)	f(x)	f(b)

   Используем интерполяцию Лагранжа, получаем $L_2(x)=a_0+a_1x+a_2x^2=0$
4. Ищем корни $x_{1,2}=\frac{-a_1\pm\sqrt{a_1^2-4a_0a_2}}{2a_2}$ и выбираем тот x_1 или x_2, который находится в интервале от *a* до *b*. Это будет x_{k+1}
5. Если $x_{k+1}\in[a,x_k]$, то сдвигаем правую границу b= x_k ,

иначе сдвигаем левую границу a= x_k

6. Можем переходить к следующей итерации:
   $x_k = x_{k+1}$
7. Повторяем до заданной точности:
   Если $|x_{k+1} - x_k| \le \varepsilon$, то останавливаемся – заданная точность достигнута,
   иначе продолжаем, переходя к пункту (3).

Берется x0, считается значение в этой точки и через 3 точки строится парабола. Смотрится где она пересекает ось. Далее меняется граница а или б. Ищем точку, где парабола пересекает ось x. Смотрится где находится этот x и отбрасывается та часть, где его нет.

15. Метод Ньютона, метод простых итераций для решения нелинейных уравнений и их систем.

Метод Ньютона с коррекцией:

1. Выбор интервала с корнем: $[a,b] \to f(a) \cdot f(b) \le 0$
2. $x_0 = \frac{a+b}{2}, \quad f(x_0)$
3. Alfa=1
4. $x_{k+1} = x_k - Alfa \cdot \frac{f(x_k)}{f'(x_k)}$
   Если $|f(x_{k+1})| \ge |f(x_k)|$, то Alfa=Alfa/2 и снова вычисляем x_{k+1}
5. Повторяем до заданной точности:
   Если $|x_{k+1} - x_k| \le \varepsilon$, то останавливаемся – заданная точность достигнута,
   иначе продолжаем, переходя к пункту (3).
   Ищем середину, потом значение в этой точке, потом производную. Проводим касательную к этой точке. Находим точку пересечения касательной и оси x. Там будет новая точка, рассматриваем ее.

Метод Якоби (простых итераций):

Изначально задана функция f(x)=0.
Преобразовываем её к неявному виду $x = \varphi(x)$, например, прибавлением с каждой стороны +x.

1. Выбор интервала с корнем: $[a,b] \to f(a) \cdot f(b) \le 0$
2. $x_0 = \frac{a+b}{2}, \quad f(x_0)$
3. $x_{k+1} = \varphi(x_k)$
4. Повторяем до заданной точности:
   Если $|x_{k+1} - x_k| \le \varepsilon$, то останавливаемся – заданная точность достигнута,
   иначе продолжаем, переходя к пункту (3).
   Условие сходимости $\varphi'(x_k) < 1$.
   При необходимости коррекции, можно домножить f(x) на функцию, не имеющую нулей на заданном интервале (или на константу), так, чтобы впоследствии это условие сходимости соблюдалось:
   $\varphi(x) = x + k \cdot f(x)$
   $\varphi'(x) = 1 + k \cdot f'(x)$
   Ищем середину. Прибавляем к функции с обеих сторон x. Получается x=ф(x). Обозначаем полученную фун ф(x)+x=фи(x). x-нов. = фи(x стар.) при условии что | фи(x стар)|<1, pri neobxodimosti vvoditsja koeficent k, s pomowju kotorogo i podgonaetsja po uslovie.

Метод Ньютона для решения систем уравнений слабой нелинейности

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

$$W(X) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \dots & \dots & \dots \\ \frac{\partial f_n}{\partial x_1} & \dots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} - \textit{матрица Якоби}$$

Если det(W(X)) = 0, то нужно решать другим методом.

Если $\max |\Delta X_{(k+1)}| < \varepsilon$, то заданная точность достигнута, конец алгоритма.
Начальный вектор $X_{(0)}$ можно взять случайным образом.
Toze samoe, 4to i prostoj Njuton, no rewaetsja ne odno uravnenie, a sistema.. a tak odno i toze

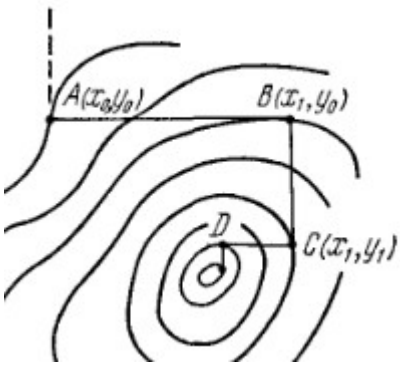
Метод Якоби для решения систем уравнений
Изначальную систему уравнений F(X)=0 приводим к неявному виду $X = \varphi(X)$.
Например, прибавив с каждой стороны вектор X, получим X=F(X)+X.
Задаём начальный вектор $X_{(0)}$ некоторыми значениями (можно случайными).
На каждой итерации вычисляем $X_{(k+1)} = \varphi(X_{(k)})$
Условие сходимости: $\|W(X)\| < 1$, где W(X) – матрица Якоби (см.метод Ньютона).
Можно так же ввести вектор коррекции $K = \begin{bmatrix} k_1 \\ \dots \\ k_n \end{bmatrix}$.

Тогда из приведённого примера:
$$X = X + K \cdot F(X)$$
$$\|1 + K \cdot W(X)\| < 1$$

Analogi4no, tolko rewaetsja ne uravnenija, a sistema uravnenij.

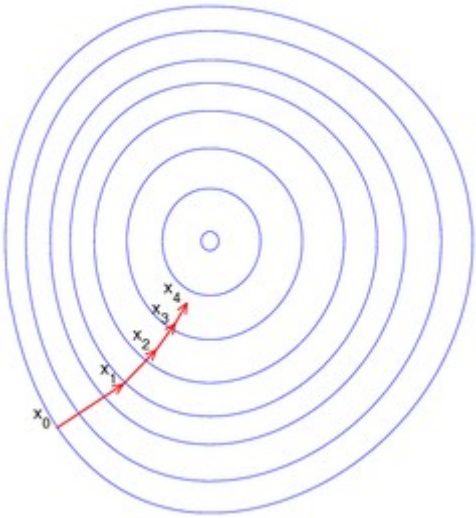
16. Задачи оптимизации. Методы спуска. Оптимизация на основе генетических алгоритмов.

1. Метод случайного поиска (Монте-Карло)
Генерируются случайным образом элементы для X. Далее X подставляется в заданную функцию F(X) и вычисляется значение этой функции. После чего сравнивается с лучшим из полученных ранее решений (если лучше, то теперь он становится лучшим, иначе лучшим остаётся предыдущий).
2. Метод покоординатного поиска.
Выбирается одна из осей, по вдоль которой создаётся цикл и вычисляются значения функции на этой оси и ищется минимум. После чего меняется ось на другую и аналогичным способом ищется новый минимум аналогичным путём.
Если найден минимум по всем осям, значит найден локальный минимум (который может быть, а может и не быть глобальным). Алгоритм обычно запускается несколько раз из разных точек, что обеспечивает большую вероятность нахождения глобального минимума.



3. Метод градиентного поиска

В точке вычисляется градиент функции. Исходя из него определяется направление, в котором будет взята следующая точка. Аналогично предыдущему методу, минимум может быть локальным. Желательно повторить алгоритм несколько раз.



4. Метод наискорейшего поиска

Вначале вычисляется градиент и по нему определяется ось, по которой начинается вычисление по методу координатного поиска.

Генетические алгоритмы

1. Создание начальной популяции случайным образом (рекомендуется N>500)
2. «Кроссовер» (скрещивание)
   Берём пару особей и скрещиваем между собой, при необходимости исключаем повторы.
   Получаем в 2 раза больше особей.
3. Селекция (отбор).
   Отбор можно проводить по следующим критериям:
   - элитный (только по f(x))
   - рулетка (случайный выбор, но с учётом f(x))
   - турнирный (вся популяция делится на пары и из каждой пары находится лучший по f(x)
4. «Мутация»
   С вероятностью p (например, 0.001) у особи случайно меняются местами 2 «гена».
5. Если критерии выполнены, то решение найдено и конец алгоритма, иначе возврат к пункту (2).
   В роли критериев могут выступать:
   - количество итераций алгоритма
   - n итераций популяция не меняется

- более скольки-то процентов особей являются клонами друг друга (например, 85%)

17. Классический метод решения линейных дифференциальных уравнений с постоянными коэффициентами.

Для линейного неоднородного дифференциального уравнения n-го порядка

y(n) + a1(x) y(n-1) + ... + an-1 (x) y' + an(x) y = f(x),

где y = y(x) — неизвестная функция, a1(x), a2(x), ..., an-1(x), an(x), f(x) — известные, непрерывные, справедливо:
1) если y1(x) и y2(x) — два решения неоднородного уравнения, то функция
y(x) = y1(x) - y2(x) — решение соответствующего однородного уравнения;
2) если y1(x) решение неоднородного уравнения, а y2(x) — решение соответствующего однородного уравнения, то функция
y(x) = y1(x) + y2(x) — решение неоднородного уравнения;
3) если y1(x), y2(x), ..., yn(x) — n линейно независимых решений однородного уравнения, а yч(x) — произвольное решение неоднородного уравнения,
то для любых начальных значений
x0, y0, y0,1, ..., y0,n-1
существуют такие значения
c*1, c*n, ..., c*n, что решение
y*(x)=c*1 y1(x) + c*2 y2(x) + ... + c*n yn (x) + yч(x)
удовлетворяет при x = x0 начальным условиям
y*(x0)=y0, (y*)'(x0)=y0,1 , ...,(y*)(n-1)(x0)=y0,n-1.

Выражение
y(x)= c1 y1(x) + c2 y2(x) + ... + cn yn(x) + yч(x)
называется общим решением линейного неоднородного дифференциального уравнения n-го порядка.

Для отыскания частных решений неоднородных дифференциальных уравнений с постоянными коэффициентами с правыми частями вида:
Pk(x)exp(ax)cos(bx) + Qm(x)exp(ax)sin(bx),
где Pk(x), Qm(x) — многочлены степени k и m соответственно, существует простой алгоритм построения частного решения, называемый **методом подбора.**

Метод подбора, или метод неопределенных коэффициентов, состоит в следующем.
Искомое решение уравнения записывается в виде:
(Pr(x)exp(ax)cos(bx) + Qr(x)exp(ax)sin(bx))xs,
где Pr(x), Qr(x) — многочлены степени r = max(k, m) с неизвестными коэффициентами pr , pr-1, ..., p1, p0, qr, qr-1, ..., q1, q0.
Сомножитель xs называют резонансным сомножителем. Резонанс имеет место в случаях, когда среди корней характеристического уравнения есть корень
l =a ± ib кратности s.
Т.е. если среди корней характеристического уравнения соответствующего однородного уравнения есть такой, что его действительная часть совпадает с коэффициентом в показателе степени экспоненты, а мнимая — с коэффициентом в аргументе тригонометрической функции в правой части уравнения, и кратность этого корня s, то в искомом частном решении присутствует резонансный сомножитель xs. Если же такого совпадения нет (s=0), то резонансный сомножитель отсутствует.

Подставив выражение для частного решения в левую часть уравнения, получим обобщенный многочлен того же вида, что и многочлен в правой части уравнения, коэффициенты которого неизвестны.

Два обобщенных многочлена равны тогда и только тогда, когда равны коэффициенты при сомножителях вида xtexp(ax)sin(bx), xtexp(ax)cos(bx) с одинаковыми степенями t.

Приравняв коэффициенты при таких сомножителях, получим систему 2(r+1) линейных алгебраических уравнений относительно 2(r+1) неизвестных. Можно показать, что такая система совместна и имеет единственное решение.

Iwetsja diksriminant i v zavisimosti ot znaka (><=0) v dalnejwem eto vlijaet na vid otveta.

Классический метод

Классический метод основан на утверждении, что полное решение неоднородного дифференциального уравнения (с правой частью):

$$y^{(n)} + c_1 \cdot y^{(n-1)} + c_2 \cdot y^{(n-2)} + ... + c_{n-2} \cdot y'' + c_{n-1} \cdot y' + c_n \cdot y = f(x)$$

всегда есть сумма общего решения однородного уравнения (правая часть равна **0** $f(x) = 0$) - свободной составляющей $y_0(x)$ и частного решения неоднородного уравнения - принужденной составляющей $y(\infty)$:

$$y(x) = y_0(x) + y(\infty),$$

где свободная составляющая всегда есть сумма экспонент:

$$y_0(x) = A_1 \cdot e^{p_1 x} + A_2 \cdot e^{p_2 x} + ... + A_n \cdot e^{p_n x}.$$

Здесь $p_1, p_2, ..., p_n$ - корни характеристического уравнения для исходного однородного уравнения:

$$p^n + c_1 \cdot p^{n-1} + c_2 \cdot p^{n-2} + ... + c_{n-1} \cdot p + c_n = 0,$$

а неизвестные коэффициенты $A_1, A_2, ..., A_n$ - неизвестные постоянные интегрирования, которые определяются из начальных условий. Вид общего решения (свободной составляющей) зависит от характера корней характеристического уравнения. Например, для дифференциального уравнения второго порядка

$$y'' + c_1 \cdot y' + c_2 \cdot y = f(x)$$

характеристическое уравнение принимает вид:

$$p^2 + c_1 \cdot p + c_2 = 0,$$

и общее решение, в зависимости от характера корней, принимает форму:

$y_0(x) = A_1 \cdot e^{p_1 x} + A_2 \cdot e^{p_2 x}$ - корни вещественные $p_1 \neq p_2, \quad p_{1,2} < 0,$

$y_0(x) = (A_1 + A_2 \cdot p_1) \cdot e^{p_1 x}$ - вещественные, кратные $p_1 = p_2, \quad p_{1,2} < 0,$

$y_0(x) = A_0 \cdot e^{-b \cdot x} Sin(\omega \cdot x + \varphi)$ - корни комплексно- сопряженные.

$$p_{1,2} = -b \pm j\omega, b > 0$$

Во всех трех выражениях A_1, A_2, A_0, φ - постоянные интегрирования, определяемые из начальных условий.

18. Операторный метод решения линейных дифференциальных уравнений с постоянными коэффициентами. Преобразования Лапласа и его свойства.

Операторный метод

Основой операторного метода является преобразование Лапласа:

$$L\{y(x)\} = Y(p) = \int_0^{\infty} e^{px} \cdot |y(x)|\, dx, \quad p = -s + j\omega .$$

Его основные свойства таковы:

1. Умножение на константу оригинала приводит к умножению на нее же образа:

$$L\{k \cdot y(x)\} = k \cdot Y(p).$$

2. Преобразование Лапласа от суммы функций есть сумма их образов:

$$L\{y_1(x) + y_2(x)\} = Y_1(p) + Y_2(p).$$

3. Изображение производной включает в себя начальные условия:

$$L\{y^{(n)}(x)\} = p^n \cdot Y(p) - p^{n-1} \cdot y(0) - p^{n-2} \cdot y'(0) - \ldots - p \cdot y^{(n-2)}(0) - y^{(p-1)}(0).$$

4. Изображение интеграла от функции есть образ самой подынтегральной функции, деленный на p.

$$L\{\int_0^x y(x)dx\} = \frac{Y(p)}{p}.$$

В справочной литературе приводятся изображения по Лапласу всех элементарных функций.

Алгоритм решения дифференциального уравнения *n-го* порядка:

$$y^{(n)} + c_1 \cdot y^{(n-1)} + c_2 \cdot y^{(n-2)} + \ldots + c_{n-2} \cdot y'' + c_{n-1} \cdot y' + c_n \cdot y = f(x)$$

операторным методом заключается в переходе в область изображений:

$$y^{(n)}(x) \Rightarrow p^n Y(p) - p^{n-1} y(0) - p^{n-2} y'(0) - \ldots - p \cdot y^{(n-2)}(0) - y^{(n-1)}(0),$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$y''(x) \Rightarrow p^2 Y(p) - p \cdot y(0) - y'(0),$$

$$y'(x) \Rightarrow pY(p) - y(0),$$

$$f(x) \Rightarrow F(p).$$

Т.к. в области изображения уравнение становится алгебраическим, его решение выражается в виде дробно-рациональной функции от p.

$$Y(p) = \frac{a_0 + a_1 p + a_2 p^2 + \ldots + a_m p^m}{b_0 + b_1 p + b_2 p^2 + \ldots + b_n p^n}, \quad m \le n.$$

Для полученного изображения оригинал (решение дифференциального уравнения) находится при возвращении в область времени с помощью теоремы разложения. Она

записывается в двух формах для различных видов решений в зависимости от наличия или отсутствия нулевого корня в знаменателе изображения:

I форма: $Y(p)=\dfrac{F_1(p)}{F_2(p)}$.

а) $y(x)=\sum_{k=1}^{n}\dfrac{F_1(p_k)}{F_2'(p_k)}\cdot e^{p_k x}$, если корни простые вещественные,

б) $y(x)=\sum_{k=1}^{n-\alpha}\dfrac{F_1(p_k)}{F_2'(p_k)}\cdot e^{p_k x}+\dfrac{1}{(\alpha-1)!}\cdot\dfrac{d^{(\alpha-1)}}{dp^{(\alpha-1)}}\left[\dfrac{F_1(p)}{F_3(p)}\cdot e^{px}\right]_{p=p_{n-\alpha+1}}$,

$F_2(p)=F_3(p)(p-p_{n-\alpha+1})^{\alpha}$,

если среди них есть кратные (α - кратность),

в) $y(x)=2\cdot Real\left\{\dfrac{F_1(p_1)}{F_2'(p_1)}\cdot e^{p_1 x}\right\}$, для каждой пары комплексно-сопряженных корней $p_{1,2}=-b\pm j\omega$.

II форма: $Y(p)=\dfrac{F_1(p)}{pF_2(p)}$.

а) $y(x)=\dfrac{F_1(0)}{F_2(0)}+\sum_{k=1}^{n}\dfrac{F_1(p_k)}{p_k\cdot F_2'(p_k)}\cdot e^{p_k x}$, если корни простые вещественные,

б) $y(x)=\dfrac{F_1(0)}{F_2(0)}+\sum_{k=1}^{n-\alpha}\dfrac{F_1(p_k)}{p_k\cdot F_2'(p_k)}\cdot e^{p_k x}+\dfrac{1}{(\alpha-1)!}\cdot\dfrac{d^{(\alpha-1)}}{dp^{(\alpha-1)}}\left[\dfrac{F_1(p)}{p\cdot F_3(p)}\cdot e^{px}\right]_{p=p_{n-\alpha+1}}$,

$F_2(p)=F_3(p)(p-p_{n-\alpha+1})^{\alpha}$,

если среди них есть кратные (α - кратность),

в) $y(x)=\dfrac{F_1(0)}{F_2(0)}+2\cdot Real\left\{\dfrac{F_1(p_1)}{p_1\cdot F_2'(p_1)}\cdot e^{p_1 x}\right\}$,

для каждой пары комплексно-сопряженных корней $p_{1,2}=-b\pm j\omega$.

Во всех формулах p_k, $k=1,2,...,n$ - корни характеристического уравнения

$$F_2(p)=p^n+c_1\cdot p^{n-1}+c_2\cdot p^{n-2}+...+c_{n-1}\cdot p+c_n=0.$$

19. Полуаналитические методы решения линейных дифференциальных уравнений. Метод последовательного дифференцирования для решения задачи Коши.

Полуаналитические подходы особенно эффективны в зонах так называемого краевого эффекта, который возникает в результате сосредоточенных воздействий на краях конструкции и/или в промежуточных зонах, ибо при этом часть составляющих решения представляет собой быстроменяющиеся функции, скорость изменения которых не всегда может быть адекватно учтена при использовании традиционных численных методов.

Кроме того, при численном решении сложных задач строительной механики предварительное аналитическое изучение отдельных локальных свойств проблемы может принести значительную пользу. Сравнение с аналитическими решениями сложной задачи в более простых и частных случаях позволяет дать оценку принятой расчетной схемы конструкции, используемого метода, алгоритма и полученного решения, в частности его точности.

Полуаналитические методы:

А). Метод **«Пикара»** (только ДУ 1-го порядка)

$y' = f(t, y), y(0) = y_0$

$$\int_0^t y' dt = \int_0^t f(t, y) dt$$

$$y(t) = y(0) + \int_0^t f(t, y) dt$$

Пример:

$y' = -2y + t^2, y(0) = 1$

Итерация1: $y(t) = 1 + \int_0^t (-2 + t^2) dt = 1 - 2t + \frac{t^3}{3}$

Итерация2: $y(t) = 1 + \int_0^t (-2(1 - 2t + \frac{t^3}{t}) + t^2) dt = 1 - 2t + 2t^2 + \frac{t^3}{3} - \frac{t^4}{6}$

Недостаток: только уравнение 1-го порядка.

Б). **Метод последовательного дифференцирования.**

$$y(t) = y(0) + y'(0) * t + \frac{y''(0)}{2!} t^2 + \frac{y'''(0)}{3!} t^3 + \ldots$$

$y' = -2y + t^2, y(0) = 1$

$y'(0) = -2 * 1 + 0^2 = -2$

$y'' = -2y' + 2t \rightarrow y''(0) = -2y'(0) + 2 * 0 = 4$

$y''' = -2y'' + 2 \rightarrow y'''(0) = -2 * y''(0) + 2 = -6$

$y^{IV} = -2y''' - y^{IV}(0) = -2 * y'''(0) = 12$

Ответ: $y(t) = 1 - 2t + 2t^2 - t^3 + \frac{t^4}{2};$

20. Численные методы решения дифференциальных уравнений и их систем. Явные и неявные схемы. Методы Рунге-Кутты для одного дифференциального уравнения и для системы ОДУ.

Универсальная формула для численных методов:

$$y_{r+1} = y_k + \int_{t_k}^{t_{k+1}} f(t, y) dt$$

Общие сведения

Только малая часть дифференциальных уравнений может быть решена в квадратурах, т.е. аналитическими методами может быть получено выражение для неизвестной функции *y(x),* входящей в дифференциальное уравнение *n-го* порядка:

$$y^{(n)} = f(x, y, y', y'', y''', ..., y^{(n-1)}),$$

заданного в форме задачи Коши, т.е. с начальными условиями:

$$y(0) = y_0, \quad y'(0) = y_0', \quad y''(0) = y_0'', \quad \ldots, \quad y^{(n-1)}(0) = y^{(n-1)}{}_0.$$

В основном же, такие задачи решаются численными методами. Любое диффуравнение *n-го* порядка всегда можно элементарной заменой переменных свести к системе п диффуравнений 1-го порядка:

$$\begin{cases} y' = z_1(x), \\ z_1' = z_2(x), \\ \quad \ldots\ldots\ldots, \\ z_{n-1}' = f(x, y, z_1, z_2, \ldots, z_{n-1}). \end{cases}$$

Поэтому все численные методы рассматриваются в применении к решению уравнений 1-го порядка вида $y' = f(x,y)$ и их систем.

Сущность численных методов состоит в следующем. На интервале наблюдения $[x_0, x_n]$ выбирается некоторое множество точек, называемое сеткой: $x_0 < x_1 < x_2 < \ldots < x_n$ с начальным условием $y(0) = y_0$. На узлах сетки $y_1, y_2, y_3, \ldots, y_n$ определяются значения искомой функции $y(x)$. Тогда проблема численного решения дифференциального уравнения заключается в построении рекуррентной процедуры определения последующего значения функции y_{k+1} в точке x_{k+1} по предыдущему y_k в точке x_k. Такая процедура называется разностной схемой. Разность $h = \Delta x = x_{k+1} - x_k$ называют шагом сетки или шагом интегрирования и в большинстве случаев полагают постоянной.

Явные методы

Явными называют такие методы, в которых последующее значение функции явно выражается через предыдущее. Простейший пример - метод Эйлера:

$$y_{k+1} = y_k + h \cdot f(x_k, y_k).$$

Явные методы более устойчивы, но обладают свойством накапливать ошибку с объемом вычислений.

Явные методы

Явными называют такие методы, в которых последующее значение функции явно выражается через предыдущее. Простейший пример - метод Эйлера:

$$y_{k+1} = y_k + h \cdot f(x_k, y_k).$$

Явные методы более устойчивы, но обладают свойством накапливать ошибку с объемом вычислений.

Неявные методы

В таких методах неизвестное значение функции в новом узле входит как в левую, так и в правую часть разностной схемы. Например, метод Эйлера-Коши:

$$y_{k+1} = y_k + \frac{h}{2}(f(x_k, y_k) + f(x_{k+1}, y_{k+1})).$$

Неявный метод требует предварительного определения y_k, которое можно вычислить с использованием явного метода, например метода Эйлера (см. выше). Такие методы неустойчивы, но обладают большей точностью.

Методы прогноза и коррекции

Это двухступенчатые комбинированные методы, объединяющие достоинства явных и неявных схем. На шаге прогноза определяется приближенное значение y_{k+1} с помощью явного метода, а затем, на шаге коррекции, строится итерационная процедура, в которой найденное решение уточняется по неявной схеме. Процесс коррекции ограничивается либо заданной точностью, либо заданным числом итераций. Пара методов Эйлера и Эйлера-Коши или многошаговая схема Адамса-Башфорта может быть рассмотрена как пример метода прогноза и коррекции.

Методы Рунге-Кутта

Различные методы данной группы отличаются друг от друга объемом производимых вычислений и получаемой при этом точностью. Они включают в себя комбинации явных и неявных методов, в том числе и метод Эйлера (метод Рунге-Кутта 0-го порядка), метод Эйлера-Коши (метод Рунге-Кутта 1-го порядка). В качестве примера приведем метод Рунге-Кутта 4-го порядка:

$$y_{k+1} = y_k + \frac{h}{6} \cdot [f_1 + 2f_2 + 2f_3 + f_4],$$

где

$$f_1 = f(x_k, y_k), \qquad f_2 = f(x_{k+\frac{1}{2}}, y_k + \frac{h}{2} \cdot f_1),$$

$$f_3 = f(x_{k+\frac{1}{2}}, y_k + \frac{h}{2} \cdot f_2), \qquad f_4 = f(x_{k+1}, y_k + h \cdot f_3).$$

Здесь точность повышается за счет использования фиктивных промежуточных точек на половине шага $x_{k+\frac{1}{2}}, y_{k+\frac{1}{2}}$.

Метод Рунге-Кутта 4-го порядка

$$f_1 = f(t_k, y_k),$$

$$f_2 = f(t_k + \frac{h}{2}, y_k + \frac{h}{2} * f_1),$$

$$f_3 = f(t_k + \frac{h}{2}, y_k + \frac{h}{2} * f_2),$$
$$f_4 = f(t_k + h, y_k + h * f_3),$$
$$y_{k+1} = y_k + \frac{h}{6}[f_1 + 2f_2 + 2f_3 + f_4]$$

!!! Все численные методы предназначены для решений дифференциальных уравнений только первого порядка.

21. Многошаговые схемы. Прогноз и коррекция. Метод Адамса-Башфорта.

Ме́тод А́дамса — разностный метод численного интегрирования обыкновенных дифференциальных уравнений, позволяющий вычислять таблицу приближённых значений решения в начальных точках.

Методы Адамса обладают лучшей по сравнению с методами Рунге — Кутта устойчивостью.

Метод Адамса – явный многошаговый метод. $L_3(x) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$ Сначала вычисляем y_1, y_2, y_3 и далее y_{k+1}

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

Метод Башфорта – неявный многошаговый

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

y_{k+1} начальный можно найти методом Адамса

Т.к. в **методах Адамса** аппроксимация $u'(x)$ проводится только по двум точкам, то это означает, что коэффициенты левой части принимают значения $a_0 = -a_1 = 1, \quad a_m = 0, \quad m = 2,3,\ldots,k$. Следовательно, методы Адамса можно записать следующим образом

$$\frac{y_i - y_{i-1}}{h} = \sum_{m=0}^{k} b_m f_{i-m} \quad .. \quad (1)$$

Если $b_0 = 0$ получаем явные (экстраполяционные) **методы Адамса**, если $b_0 \neq 0$, методы будут называться неявными (интерполяционными) **методами Адамса**. В зависимости от количества k предыдущих узлов, методы называются k –шаговыми. Например трех, четырехшаговые методы Адамса.

Чтобы получить порядок аппроксимации конкретной многошаговой схемы, в том числе схемы Адамса, необходимо исследовать невязку (погрешность аппроксимации) этой схемы.

$$\psi = -\sum_{m=0}^{k} \frac{a_m}{h} u_{i-m} + \sum_{m=0}^{k} b_m f(x_{i-m}, u_{i-m}) \qquad i = k, k+1, \ldots \tag{2}$$

Эта функция ψ получается как результат подстановки точного решения $u(x)$ задачи Коши в разностное уравнение вида (3) из предыдущего пункта. Чтобы метод имел требуемый порядок аппроксимации, необходимо чтобы выполнялись условия согласованности многошагового метода, т.е условия, накладываемые на коэффициенты схемы a_m, b_m. (см.Сам.Ч.М. стр.233).

Для **методов Адамса** доказывается, что наивысший порядок аппроксимации *k*-шагового неявного метода Адамса равен *k+1,* а наивысший порядок аппроксимации *k*-шагового явного ($b_0 = 0$) метода Адамса равен *k.*

Приведем примеры конкретных схем Адамса.

Явные методы, т.е. $b_0 = 0$. При k=1 получаем метод Эйлера, $b_1 = 1$

$$\frac{y_i - y_{i-1}}{h} = f_{i-1}$$

При *k=2* получим схему второго порядка аппроксимации

$$\frac{y_i - y_{i-1}}{h} = \frac{2}{3} f_{i-1} - \frac{1}{2} f_{i-2}$$

При *k=3* получим схему третьего порядка аппроксимации

$$\frac{y_i - y_{i-1}}{h} = \frac{1}{12}(23 f_{i-1} - 16 f_{i-2} + 5 f_{i-3})$$

При *k=4* получим схему четвертого порядка аппроксимации

$$\frac{y_i - y_{i-1}}{h} = \frac{1}{24}(55 f_{i-1} - 59 f_{i-2} + 37 f_{i-3} - 9 f_{i-4})$$

При *k=5* получим схему пятого порядка аппроксимации

$$\frac{y_i - y_{i-1}}{h} = \frac{1}{720}(190 f_{i-1} - 277 f_{i-2} + 2616 f_{i-3} - 1274 f_{i-4} + 251 f_{i-5})$$

Для неявных методов Адамса. При *k*=1 получаем схему 2-го порядка точности

$$\frac{y_i - y_{i-1}}{h} = \frac{1}{2}(f_i + f_{i-1})$$

При *k*=2 получаем схему 3-го порядка точности

$$\frac{y_i - y_{i-1}}{h} = \frac{1}{12}(5 f_i + 8 f_{i-1} - f_{i-2})$$

При *k*=3 получаем схему 4-го порядка точности

$$\frac{y_i - y_{i-1}}{h} = \frac{1}{24}(9 f_i + 19 f_{i-1} - 5 f_{i-2} + f_{i-3})$$

При *k*=4 получаем схему 5-го порядка точности

$$\frac{y_i - y_{i-1}}{h} = \frac{1}{720}(251 f_i + 646 f_{i-1} - 264 f_{i-2} + 106 f_{i-3} - 19 f_{i-4})$$

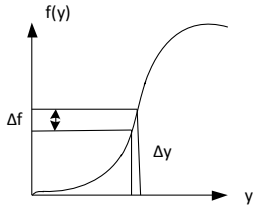
В практических расчетах чаще всего используется вариант явного метода Адамса, имеющий 4 порядок точности и использующий на каждом шаге значения сеточных функций в четырех предыдущих узлах.

На основе **методов Адамса** разработан целый ряд весьма сложных алгоритмов и программ для математических пакетов. В этих программах **методы Адамса** реализованы таким образом, что имеется возможность изменять не только величину шага сетки, но и порядок самого метода. В настоящее время существуют программы, максимальный порядок точности которых достигает 13.

Если требуется достичь ***ЛЮБОЙ*** точности на шаге, то следует использовать методы прогноза и коррекции. Этот подход состоит в том, что расчет траектории, задаваемой уравнением, на каждом шаге происходит многократно. А именно, сначала происходит расчет приближенного значения функции на конце шага какой-либо простой формулой (например, методом Эйлера), далее в этой точке вычисляется производная, и расчет происходит снова из начальной точки на шаге, но с уточненным значением производной. Последняя операция — уточнения производной и значения функции на конце шага — происходит ***МНОГОКРАТНО НА КАЖДОМ ШАГЕ***, то есть до тех пор, пока вычисленные значения (функции и производной в конце шага) не перестанут меняться или будут меняться уже незначительно, меньше чем задаваемая заранее величина ε. Только тогда можно сказать, что точность ε достигнута.

22. Методы решения «жестких» ОДУ. Метод Гира.

«Жесткие» ОДУ – функция f(t,y) обладает свойством жесткой нелинейности.



Метод Гира. Не явный метод с элементом цифровой фильтрации (сглаживания).

Метод Гира II порядка:

$$y_{k+1} = \frac{4}{3}y_k - \frac{1}{3}y_{k-1} + \frac{2}{3}hf(t_{k+1}, y_{k+1})$$

Метод Эйлера: $y_{k+1} = y_k + h \cdot f(t_k, y_k)$

Для реализации требуется прогноз III порядка:

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

23. Методы решения краевых задач. Метод стрельбы и метод конечных разностей.

Граничная задача – задача для диффур, только 2-го порядка.

$y'' = f(t, y, y'), y(0), y(T)$ краевая $y(0), y'(0)$ – Коши

1. *Линейное ОДУ:* $p(t) \cdot y'' + q(t) \cdot y' + m(t) \cdot y = f(t)$

Метод конечных разностей: $y(t) \to y_k, y' \to \frac{y_{k+1} - y_{k-1}}{2h}$

$$y'' \to \frac{y_{k+1} - 2y_k + y_{k-1}}{h^2}$$

$$p(t_k) = \frac{y_{k+1} - 2yk + y_{k-1}}{h^2} + q(tk)\frac{y_{k+1} - y_k - 1}{2h} + m(tk)y_k = f(tk)$$

$$y_{k-1}\left(\frac{p(tk)}{h^2} - \frac{q(tk)}{2h}\right) + y_k\left(m(tk) - \frac{2p(tk)}{h^2}\right) + y_{k+1}\left(\frac{p(tk)}{h^2} + \frac{q(tk)}{2h}\right) = f(tk)$$

tk= k*h, k=1,2,3,..,n-1

Метод стрельбы.

Происходит подмена задач. Все сводится к задаче Коши.

$y'' = f(t, y, y'), y(0), y(T)$

$y(0), y'(0) = tg\ \alpha^*$

1. Выбираем произвольный угол прицеливания $\alpha_1 - 90° < \alpha_1 < 90°$

Tg α_1=y'(0,α_1)→y(t)→$y_{(T)}$ α_1

2. $\alpha_2 = \alpha_1 + \Delta\alpha(\text{коррекция}) \to tg\ \alpha_2 = y'(0, \alpha_2) \to y(t) \to y_{(T)}(\alpha_2)$

чусвствит. $\frac{\partial y^T}{\partial \alpha} \approx \frac{y_T(\alpha_2) - y_T(\alpha_1)}{\Delta\alpha} \to \begin{cases} y' = z \\ z' = f(t, y, z) \end{cases} \to \Delta\alpha^* = \frac{y_{(T)} - y_T(\alpha_1)}{\partial y_T / \partial \alpha}$

3. $\alpha_3 = \alpha_1 + \Delta\alpha^* \to tg\ \alpha_3 = y'(0, \alpha_3) \to y(t) \to y(\alpha_3) \approx y(T)$

Метод конечных разностей.

Метод конечных разностей — широко известный и простейший метод интерполяции. Его суть заключается в замене дифференциальных коэффициентов уравнения на разностные коэффициенты, что позволяет свести решение дифференциального уравнения к решению его разностного аналога, то есть построить его конечно-разностную схему.

Так, заменив производную в обыкновенном дифференциальном уравнении

$$u'(x) = 3u(x) + 2$$

на конечную разность

$$\frac{u(x+h) - u(x)}{h} \approx u'(x),$$

получаем аппроксимированную форму (конечно-разностную схему)

$$u(x+h) = u(x) + h \cdot (3u(x) + 2).$$

Последнее выражение носит название конечно-разностного уравнения, а его решение соответствует приближённому решению первоначального дифференциального уравнения.

24. Метод конечных разностей для решения дифференциальных уравнений в частных производных. Синтез сетки решения.

Одним из наиболее распространенных подходов численного решения дифференциальных уравнений является *метод конечных разностей* (*метод сеток*) [5,11]. Следуя этому подходу, область решения D представляется в виде дискретного (как правило, равномерного) набора (*сетки*) точек (*узлов*). Так, например, прямоугольная сетка в области D может быть задана в виде (рис. 6.1)

$$\begin{cases} D_h = \{(x_i, y_j) : x_i = ih,\ y_i = jh,\ 0 \le i, j \le N+1, \\ h = 1/(N+1), \end{cases}$$

где величина N задает количество узлов по каждой из координат области D.

Обозначим оцениваемую при подобном дискретном представлении аппроксимацию функции $u(x,y)$ в точках (x_i, y_j) через u_{ij}. Тогда, используя *пятиточечный шаблон* (см. рис. 6.1) для вычисления значений производных, уравнение Пуассона может быть представлено в *конечно-разностной форме*

$$\frac{u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} - 4u_{ij}}{h^2} = f_{ij}.$$

Данное уравнение может быть разрешено относительно u_{ij}

$$u_{ij} = 0.25\,(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} - h^2 f_{ij}).$$

Разностное уравнение, записанное в подобной форме, позволяет определять значение u_{ij} по известным значениям функции $u(x,y)$ в соседних узлах используемого шаблона. Данный результат служит основой для построения различных *итерационных схем* решения задачи Дирихле, в которых в начале вычислений формируется некоторое приближение для значений u_{ij}, а затем эти значения последовательно уточняются в соответствии с приведенным соотношением. Так, например, *метод Гаусса-Зейделя* для проведения итераций уточнения использует правило

$$u_{ij}^k = 0.25(u_{i-1,j}^k + u_{i+1,j}^{k-1} + u_{i,j-1}^k + u_{i,j+1}^{k-1} - h^2 f_{ij}),$$

по которому очередное *k*-ое приближение значения u_{ij} вычисляется по последнему *k*-ому приближению значений $u_{i-1,j}$ и $u_{i,j-1}$ и предпоследнему (*k-1*)-ому приближению значений $u_{i+1,j}$ и $u_{i,j+1}$. Выполнение итераций обычно продолжается до тех пор, пока получаемые в результате итераций изменения значений u_{ij} не станут меньше некоторой

заданной величины (*требуемой точности вычислений*). Сходимость описанной процедуры (получение решения с любой желаемой точностью) является предметом всестороннего математического анализа (см., например, [5,11]), здесь же отметим, что последовательность решений, получаемых методом сеток, равномерно сходится к решению задачи Дирихле, а погрешность решения имеет порядок h^2.

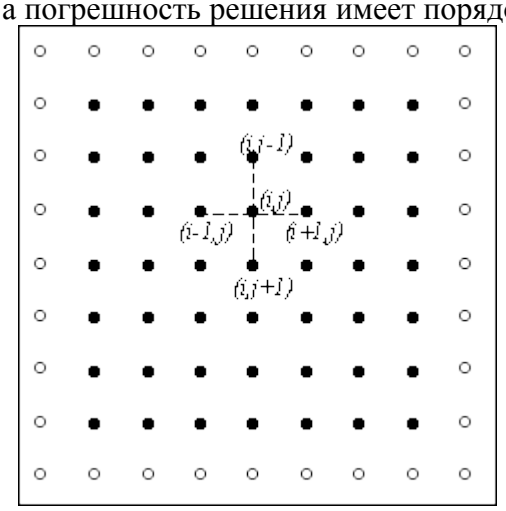


Рис. 6.1. Прямоугольная сетка в области *D* (темные точки представляют внутренние узлы сетки, нумерация узлов в строках слева направо, а в столбцах - сверху вниз).

Рассмотренный алгоритм (метод Гаусса-Зейделя) на псевдокоде, приближенного к алгоритмическому языку C++, может быть представлен в виде:

```
// Алгоритм 6.1
 do {
   dmax = 0; // максимальное изменение значений u
   for ( i=1; i<N+1; i++ )
    for ( j=1; j<N+1; j++ ) {
      temp = u[i][j];
      u[i][j] = 0.25*(u[i-1][j]+u[i+1][j]+
              u[i][j-1]+u[i][j+1]–h*h*f[i][j]);
      dm = fabs(temp-u[i][j]);
      if ( dmax < dm ) dmax = dm;
    }
 } while ( dmax > eps );
```

(напомним, что значения u_{ij} при индексах $i,j=0,N+1$ являются граничными, задаются при постановке задачи и не изменяются в ходе вычислений).

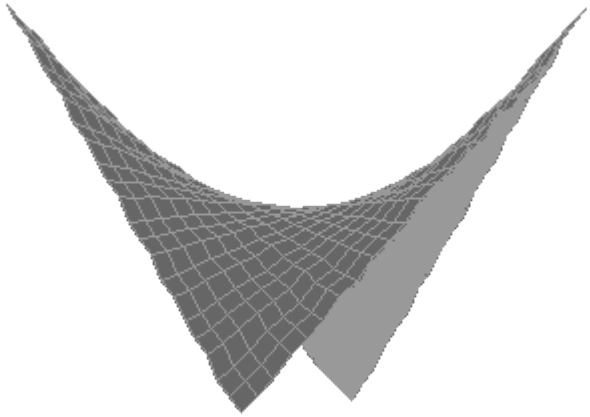


Рис. 6.2. Вид функции $u(x,y)$ в примере для задачи Дирихле

Для примера на рис. 6.2 приведен вид функции $u(x,y)$, полученной для задачи Дирихле при следующих граничный условиях:

$$\begin{cases} f(x,y) = 0, & (x,y) \in D, \\ 100 - 200x, & y = 0, \\ 100 - 200y, & x = 0, \\ -100 + 200x, & y = 1, \\ -100 + 200y, & x = 1, \end{cases}$$

Общее количество итераций метода Гаусса-Зейделя составило 210 при точности решения $eps = 0.1$ и $N = 100$ (в качестве начального приближения величин u_{ij} использовались значения, сгенерированные датчиком случайных чисел из диапазона [-100,100]).

25.Методы анализа символов и графических изображений. Морфологические алгоритмы.

Морфологические алгоритмы используется для извлечения некоторых свойств изображения, полезных для его представления и описания. Например, контуров, остовов, выпуклых оболочек. Например, морфологическая фильтрация, утолщение или утоньшение.

В первую очередь математическая морфология используется для извлечения некоторых свойств изображения, полезных для его представления и описания. Например, контуров, остовов, выпуклых оболочек. Также интерес представляют морфологические методы, применяемые на этапах предварительной и итоговой обработки изображений. Например, морфологическая фильтрация, утолщение или утоньшение.

Входными данными для аппарата математической морфологии являются два изображения: обрабатываемое и специальное, зависящее от вида операции и решаемой задачи. Такое специальное изображения принято называть примитивом или структурным элементом.

//Помните он показывал слайды с буквой «А» её утолщали, потом делали белее тонкой — для
того что бы потом её сравнить с образцом.

Основными операциями математической морфологии являются наращивание, эрозия, замыкание и размыкание. В этих названиях отражена суть операций: наращивание увеличивает область изображения, а эрозия делает её меньше, операция замыкания позволяет замкнуть внутренние отверстия области и устранить заливы вдоль границы области, операция размыкания помогает избавиться от маленьких фрагментов, выступающих наружу области вблизи её границы. Далее будут представлены математические определения морфологических операций.

////////примеры////////
Следующие операции мы рассмотрим на конкретном примере. Пусть у нас есть следующее бинарное изображение и структурный элемент:

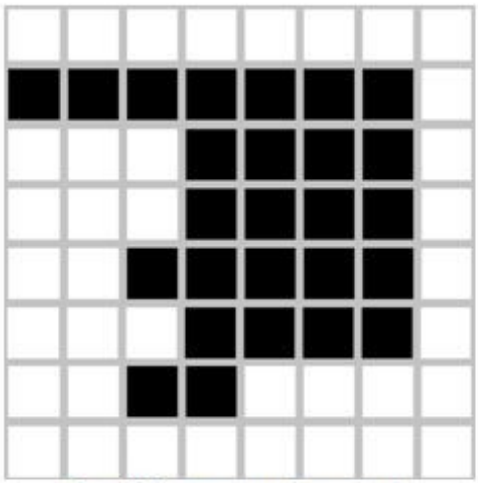


Рис. 10 Бинарное изображение B

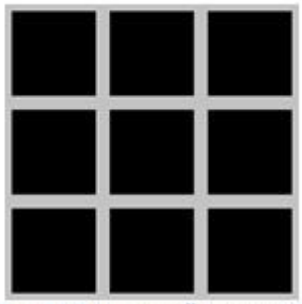


Рис. 11 Структурный элемент S

Наращивание

Структурный элемент S применяется ко всем пикселам бинарного изображения. Каждый раз, когда начало координат структурного элемента совмещается с единичным бинарным пикселом, ко всему структурному элементу применяется перенос и последующее логическое сложение с соответствующими пикселами бинарного изображения. Результаты логического сложения записываются в выходное бинарное изображение, которое изначально инициализируется нулевыми значениями.

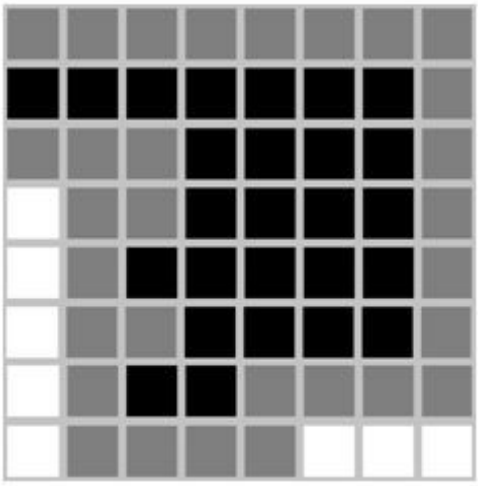


Рис. 12 Наращивание изображения B структурным элементом S^1

Эрозия

При выполнении операции эрозии структурный элемент тоже проходит по всем пикселам изображения. Если в некоторой позиции каждый единичный пиксел структурного элемента совпадет с единичным пикселом бинарного изображения, то выполняется логическое сложение центрального пиксела структурного элемента с соответствующим пикселом выходного изображения.

Рис. 13 Эрозия изображения B структурным элементом S^2

В результате применения операции эрозии все объекты, меньшие чем структурный элемент, стираются, объекты, соединённые тонкими линиями становятся разъединёнными и размеры всех объектов уменьшаются.

Размыкание

Операция эрозии полезна для удаления малых объектов и различных шумов, но у этой операции есть недостаток – все остающиеся объекты уменьшаются в размере. Этого эффекта можно избежать, если после операции эрозии применить операцию наращивания с тем же структурным элементом.
Размыкание отсеивает все объекты, меньшие чем структурный элемент, но при этом помогает избежать сильного уменьшения размера объектов. Также размыкание идеально подходит для удаления линий, толщина которых меньше, чем диаметр структурного элемента. Также важно помнить, что после этой операции контуры объектов становятся более гладкими.

Рис. 14 Размыкание изображения B структурным элементом

Замыкание

Если к изображению применить сначала операцию наращивания, то мы сможем избавиться от малых дыр и щелей, но при этом произойдёт увеличение контура объекта. Избежать этого увеличения позволяет операция эрозия, выполненная сразу после наращивания с тем же структурным элементом.

Рис. 15 Замыкание изображения B структурным элементом S

Условное наращивание

Одним из типичных применений бинарной морфологии является выделение на бинарном изображении компонент, у которых форма и размеры удовлетворяют заданным ограничениям. Во многих подобных задачах возможно построение структурного элемента, который после применения к бинарному изображению удаляет не удовлетворяющие ограничениям компоненты и оставляет несколько единичных пикселов, соответствующих удовлетворяющим ограничениям компонентам. Но для последующей обработки могут потребоваться компоненты целиком, а не только их фрагменты, оставшиеся после эрозии. Для решения этой проблемы была введена операция условного наращивания. Множество полученное в результате эрозии циклически наращивается структурным элементом S, и на каждом шаге результат уменьшается до подмножества пикселов, которые имеют единичные значения на исходном изображении B. Операция условного наращивания пояснена на рисунке ниже. На этом рисунке бинарное изображение B было подвергнуто эрозии элементом V для выделения компонент, содержащих вертикальные фрагменты высотой 3 пиксела. На полученном изображении C есть две таких компоненты. Чтобы выделить эти компоненты целиком, изображение C условно наращивается элементом D относительно исходного изображения B.

Рис. 16 Структурный элемент V

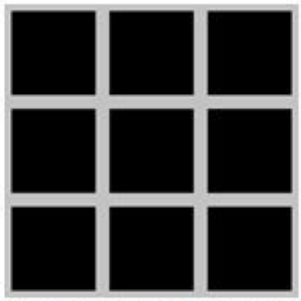


Рис. 17 Структурный элемент D

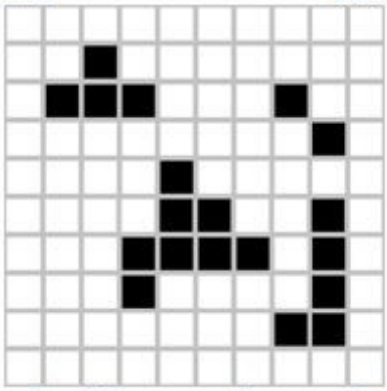


Рис. 18 Бинарное изображение B

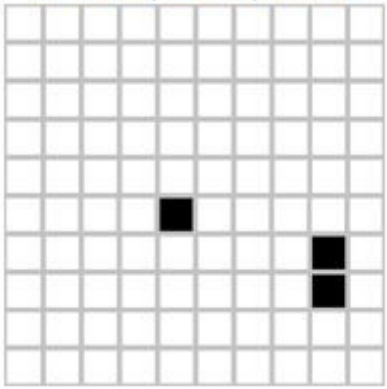


Рис. 19 Эрозия изображения B структурным элементом V

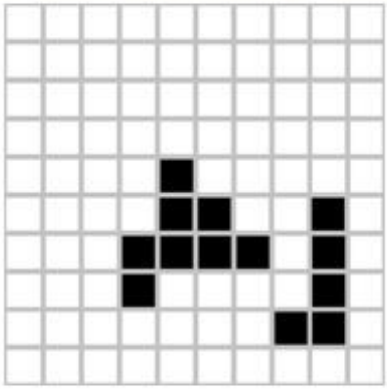


Рис. 20 Условное наращивание изображения структурным элементом D

Выделение границ

Морфологические операции можно также использовать для выделения границ бинарного объекта. Это операция очень важна, потому что граница является полным, и в то же время весьма компактным описанием объекта.

Легко заметить, что граничные точки имеют как минимум один фоновый пиксел в своей окрестности. Таким образом, применив оператор эрозии с структурным элементом, содержащим все возможные соседние элементы, мы удалим все граничные точки… Тогда граница получится с помощью операции разности множеств между исходным изображением и изображением, полученным в результате эрозии.

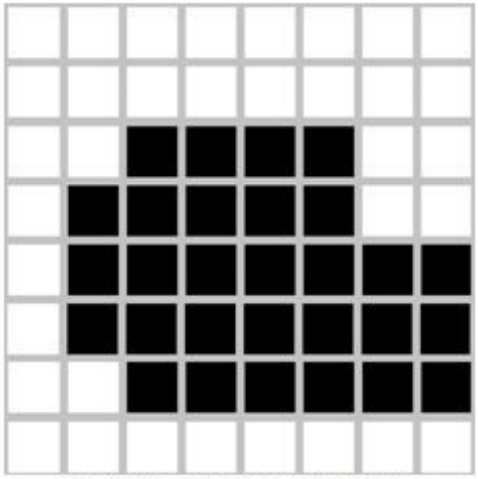


Рисунок 21 Исходное изображение

Рисунок 22 Структурный элемент для 4-компонентной окрестности

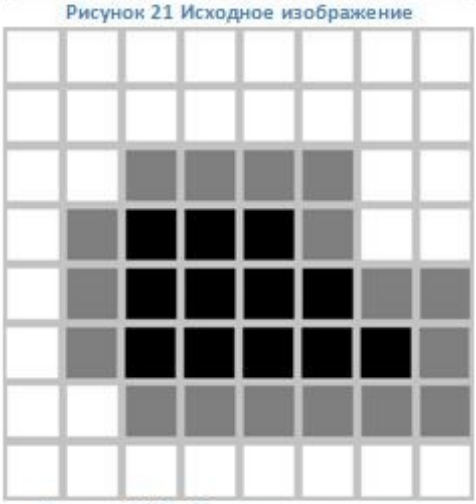


Рисунок 23 Изображение, после эрозии

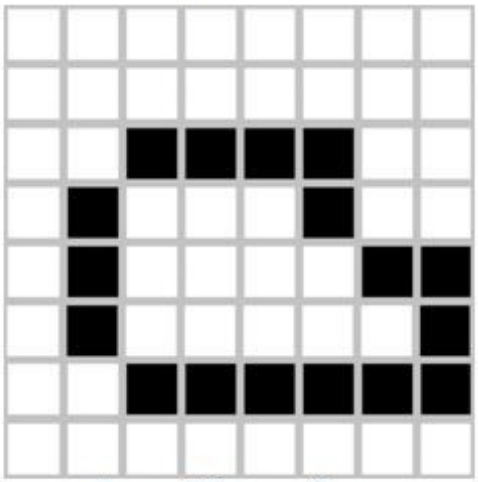


Рисунок 24 Границы объекта

26. Клеточные автоматы