

## Численные методы

### Лекция №1

#### Введение

#### Раздел №1. Численные методы алгебры

##### Тема 1.1 Основы теории погрешности

Содержание : Введение в курс. Понятие и свойства погрешностей. Неустраняемая и вычислительная погрешности. Абсолютная, относительная погрешности. Оценка погрешности. Прямая и обратная задачи теории погрешностей. Методы решения прямой задачи. Метод приближений. Методы решения обратной задачи (методы равных вкладов, равных погрешностей, метод оптимизации). Примеры приближенной оценки погрешности.

#### Введение в курс «Численные методы»

Целью освоения дисциплины «Численные методы» является:

- формирование систематических знаний о современных методах прикладной информатики, её месте и роли в системе наук;
- расширение и углубление понятий математики, информатики, численных методов;
- развитие абстрактного мышления, вычислительной, алгоритмической культур и общей математической и информационной культуры.

Дисциплина «Численные методы» относится к вариативной части профессионального цикла. Она изучается после дисциплин «Дискретная математика», «Математическая логика», «Программирование». Для ее освоения студенты также используют знания, умения, навыки, сформированные в ходе изучения основных математических курсов: «Математический анализ», «Алгебра», «Геометрия». Всего планируется проведение 7 лекций, 7 практических занятий, 12 лабораторных занятий, промежуточная аттестация — зачет.

#### Изучаемые темы и разделы:

Тема	Название
<b>Раздел 1. Численные методы алгебры</b>	
1_1	Основы теории погрешности
1_2	Численные методы решения уравнений
<b>Раздел 2. Численные методы матанализа</b>	
2_1	Интерполяция и Аппроксимация
2_2	Численное дифференцирование и интегрирование

Исторические истоки численных методов :

- Вычислительная математика как база математики
- Инженерная математика как особенная сфера применения математики

- Проблема автоматизации и упрощения расчетов
- Появление ЭВМ
- Появление численных методов решения математических задач с помощью ЭВМ

### Учебная литература

1. Корнеев, П.К. Численные методы : [16+] / П.К. Корнеев, Е.О. Тарасенко, А.В. Гладков ; Министерство образования и науки Российской Федерации, ФГАОУ ВО «Северо-Кавказский федеральный университет». – Ставрополь : СКФУ, 2017. – Часть 1. – 145 с. : ил. – Режим доступа: – URL: <http://biblioclub.ru/index.php?page=book&id=563066> – Текст : электронный.
2. Гильмутдинов, Р.Ф. Численные методы / Р.Ф. Гильмутдинов, К.Р. Хабибуллина ; Министерство образования и науки России, ФГБОУ ВО «Казанский национальный исследовательский технологический университет». – Казань : Издательство КНИТУ, 2018. – 92 с. : ил. – Режим доступа: – URL: <http://biblioclub.ru/index.php?page=book&id=500887> – Текст : электронный.
3. Численные методы / П.К. Корнеев, Е.О. Тарасенко, А.В. Гладков, М.А. Дерябин ; Министерство науки и высшего образования РФ, ФГАОУ ВО «Северо-Кавказский федеральный университет». – Ставрополь : СКФУ, 2018. – Ч. Часть 2. – 107 с. : ил. – Режим доступа: – URL: <http://biblioclub.ru/index.php?page=book&id=562830> – Текст : электронный.
4. Основы вычислительной математики, математического и информационного моделирования / авт.-сост. А.Н. Макоха, М.А. Дерябин ; Министерство образования и науки Российской Федерации, Северо-Кавказский федеральный университет. – Ставрополь : СКФУ, 2018. – 195 с. – Режим доступа: – URL: <http://biblioclub.ru/index.php?page=book&id=494783> . – Текст : электронный.
5. Бахвалов, Н.С. Численные методы в задачах и упражнениях [Электронный ресурс] : учеб. пособие / Н.С. Бахвалов, А.В. Лапин, Е.В. Чижонков. — Электрон. дан. — Москва : Издательство "Лаборатория знаний", 2015. — 243 с. — Режим доступа: <https://e.lanbook.com/book/70743>.
6. Горлач, Б.А. Математическое моделирование. Построение моделей и численная реализация [Электронный ресурс] : учеб. пособие / Б.А. Горлач, В.Г. Шахов. — Электрон. дан. — Санкт-Петербург : Лань, 2016. — 292 с. — Режим доступа: <https://e.lanbook.com/book/74673>.
7. Бахвалов, Н.С. Численные методы [Электронный ресурс] : учеб. пособие / Н.С. Бахвалов, Н.П. Жидков, Г.М. Кобельков. — Электрон. дан. — Москва : Издательство "Лаборатория знаний", 2015. — 639 с. — Режим доступа: <https://e.lanbook.com/book/70767>

### Рейтинговая система оценки текущей успеваемости студентов

Распределение рейтинговых баллов по видам оцениваемых работ представлено в следующей таблице.

№	Наименование разделов	Виды оцениваемых работ	Максимальное кол-во баллов
1	<b>Основы теории погрешности</b> <b>Численные методы решения уравнений</b>	Домашняя практическая работа	4
		Письменная проверочная работа	8
		Активная работа на занятиях	2
		Защита лабораторных работ	20
2	<b>Интерполяция и Аппроксимация</b> <b>Численное дифференцирование и интегрирование</b>	Домашняя практическая работа	2
		Письменная проверочная работа	6
		Активная работа на занятиях	2
		Защита лабораторных работ	16
3	Текущая аттестация по всем разделам	Компьютерное тестирование	40
ВСЕГО			100

### Понятие и свойства погрешностей

**Погрешность – это разница между точным значением величины и известным значением. Известное значение называют приближенным.** Погрешность возникает как правило из-за того, что точное значение величины не известно (например при измерении), а известно только приближенное значение величины. По определению понятно, что погрешность может быть положительной или отрицательной величиной, поэтому на практике используют величину взятую по модулю от погрешности (абсолютную). Для обозначения отклонения приближенного значения от точного часто используют знак +/- (или  $\pm$ ), который не нужно путать с обычными знаками математических операций. Особенность изучения или вычисления погрешностей связано с проблемами :

- Нет возможности однозначно вычислить погрешность (даже ее знак).
- Трудность определения изменения погрешности при вычислении функций, формул, решения уравнений и т. д.
- Для погрешности важно не только значение абсолютной ее величины, но и ее соотношение (отношение) с значением самой приближенной величины.

## Неустраняемая и вычислительная погрешности

Появление погрешности принято разделять на 2 процесса и 2 составляющих в погрешности — неустраняемую и вычислительную. Неустраняемая связана с измерением величины или исходных данных (имеющих погрешности, которые от нас не зависят). Вычислительная связана появлением дополнительных погрешностей при вычислениях (особенно компьютерных, где обязательно использовать округление до заданного числа разрядов). В принципе вычислительная погрешность во многом зависит от нашего алгоритма вычисления, используемых типов данных. Существуют специальные методы уменьшающие вычислительную погрешность. Итоговая погрешность объединяет оба варианта погрешности (не обязательно как сумма).

### Абсолютная, относительная погрешности

**Абсолютной погрешностью  $d$**  приближённой величины  $A$  называется модуль разности  $A_m$  – точного и приближённого значения  $A$ .

$$d = |A - A_m|$$

**Относительной погрешностью  $\gamma$**  называется отношение абсолютной погрешности к модулю приближённой величины.

$$\gamma = d(A) / |A|$$

**Абсолютную погрешность** обычно невозможно определить, так как неизвестно точное значение, поэтому пользуются оценками этой величины.

### Оценка погрешности

**Оценка погрешности  $d$**  – это положительная величина  $d'$  заведомо превышающая исходную  $d' \geq d$ . По данному определению понятно, что оценок бесконечно много. Среди всех возможных при данной информации оценок погрешностей есть наименьшая. Такая оценка называется **предельной погрешностью**. В некоторых случаях предельная погрешность равна абсолютной, а в некоторых – больше. **Целью оценки погрешности всегда является максимальное приближение к предельной погрешности.**

Таким образом, проблема построения оптимальных оценок для погрешностей фактически оказалась центральной задачей теории погрешностей. Ее принято делить на прямую и обратную задачи теории погрешностей.

**Замечание.** Если у нас известна абсолютная погрешность или какая-то её оценка  $d(A)$ , то мы можем утверждать, что точное значение точно находится на отрезке  $[A-d(A), A+d(A)]$ . Данный отрезок принято называть интервалом неопределённости величины  $A$ . Для удобства интервал неопределённости записывают в виде  $A \pm d(A)$ . Существует так же специальное правило записи погрешности:

- Все цифры, которые по разряду больше погрешности, называются верными, остальные – приближёнными.
- При записи чисел записывают только одну приближённую цифру, при записи погрешности используют одну или две значащие цифры.

## Прямая и обратная задачи теории погрешностей

**Прямая задача:** заданы значения приближённых величин  $A$ ,  $B$ ,  $C$  и их погрешности  $d(A)$ ,  $d(B)$ ,  $d(C)$  и функции этих величин  $F(A,B,C)$ . Требуется найти погрешность этой функции  $d(F)$ .

**Обратная задача:** заданы значения приближённых величин  $A$ ,  $B$ ,  $C$ , функция этих величин  $F(A,B,C)$  и погрешность этой функции  $d(F)$ . Необходимо найти подходящие погрешности  $d(A)$ ,  $d(B)$ ,  $d(C)$ . Эта задача имеет множество решений и является более сложной чем прямая задача.

### Методы решения прямой задачи

#### Метод приближений

Наиболее распространенным методом решения прямой задачи является метод последовательных приближений, когда погрешность оценивается разложением в ряд Тейлора:

$$d(F(A)) = |F(A) - F(A \pm d)| = |F(A) - F(A) \pm F'(A) \cdot d/1! \pm F''(A) \cdot d^2/2! \pm F'''(A) \cdot d^3/3! \pm \dots| =$$
$$|\pm F'(A) \cdot d/1! \pm F''(A) \cdot d^2/2! \pm F'''(A) \cdot d^3/3! \pm \dots| \Rightarrow d(F) \leq |F'(A)| \cdot d/1! + |F''(A)| \cdot d^2/2! + |F'''(A)| \cdot d^3/3! + \dots$$

Здесь  $d$  — абсолютная погрешность величины  $A$ . Для последовательных приближений учитывается конечное число членов такого ряда. В первом приближении  $d(F) \leq |F'(A)| \cdot d/1! = d_1(F)$ , во втором -  $d(F) \leq |F'(A)| \cdot d/1! + |F''(A)| \cdot d^2/2! = d_1(F) + |F''(A)| \cdot d^2/2! = d_2(F)$ , в третьем -  $d(F) \leq |F'(A)| \cdot d/1! + |F''(A)| \cdot d^2/2! + |F'''(A)| \cdot d^3/3! = d_2(F) + |F'''(A)| \cdot d^3/3! = d_3(F)$ . Формула для  $n$ -го приближения:  $d_n(F) = d_{n-1}(F) + |F^{(n)}(A)| \cdot d^n/n!$

Здесь  $F^{(n)}(A)$  обозначает  $n$ -ю производную от функции  $F(A)$ .

Для функций двух аргументов  $F(A,B)$  разложение будет гораздо сложнее

*например*  $d(F) \leq |\partial F/\partial A| \cdot d(A) + |\partial F/\partial B| \cdot d(B) + |\partial^2 F/\partial A \partial B| \cdot d(A) \cdot d(B) +$

$$+ |\partial^2 F/\partial^2 A| \cdot d^2(A)/2! + |\partial^2 F/\partial^2 B| \cdot d^2(B)/2! + \dots$$

На практике (для таких функций) ограничиваются первым приближением :

$$d(F) \leq d_1(F) = |\partial F/\partial A| \cdot d(A) + |\partial F/\partial B| \cdot d(B)$$

### Метод преобразования формулы определения

Решить прямую задачу можно и традиционным способом — используя определения абсолютной или относительной погрешности и оценки для основных математических операций (+, -, \*, /). Для этого необходимо раскрыть модуль в определении погрешности и выразить его верхнюю положительную оценку. Рассмотрим этот метод на примерах.

Примеры приближенной оценки погрешности методом преобразования :

С помощью определения абсолютной и относительной погрешности можно построить формулы для оценки погрешности математических операций :

для суммы:

$$d(A+B) = |A+B - (A \pm d(A) + B \pm d(B))| = |\pm d(A) \pm d(B)| \leq d(A) + d(B)$$

Получили удобную для практики оценку: абсолютная погрешность суммы равна сумме погрешностей.

Для разности формула аналогична:  $d(A-B) = |\pm d(A) \pm d(B)| \leq d(A) + d(B)$

Для произведения получим следующую формулу:

$$d(A \cdot B) = |A \cdot B - (A \pm d(A)) \cdot (B \pm d(B))| = |\pm B d(A) \pm A d(B) \pm d(A) \cdot d(B)| \leq |B| \cdot d(A) + |A| \cdot d(B) + d(A) \cdot d(B)$$

$$\gamma(A \cdot B) = d(A \cdot B) / |A \cdot B| = \gamma(A) + \gamma(B) + \gamma(A) \cdot \gamma(B)$$

Формула для оценки погрешности отношения (без вывода):

$$d(A/B) \leq (|B| \cdot d(A) + |A| \cdot d(B)) / (B^2 - B \cdot d(B)) = (|B| \cdot d(A) + |A| \cdot d(B)) / (B^2 (1 - \gamma(B)))$$
$$\gamma(B) = d(B) / |B|$$

### **Методы решения обратной задачи (методы равных вкладов, равных погрешностей, метод оптимизации)**

Существуют несколько методов решения обратной задачи:

- Равных погрешностей.
- Равных вкладов.
- Оптимизации (линейной или нелинейной).

#### **Метод равных погрешностей**

Суть метода – все погрешности величин считаются равными между собой.

Введем обозначение:  $d = d(A) = d(B) = d(C) = \dots$

Теперь выражение для первого приближения погрешности будет таким:

$$d(F) \leq [|\partial F / \partial A| + |\partial F / \partial B| + |\partial F / \partial C| + \dots] \cdot d$$

Окончательно формула оценки для погрешности

$$d(F) / [|\partial F / \partial A| + |\partial F / \partial B| + |\partial F / \partial C| + \dots] \leq d$$

#### **Метод равных вкладов**

Суть метода – все вклады погрешностей величин считаются равными между собой. Введем обозначение:

$$dX = |\partial F / \partial A| \cdot d(A) = |\partial F / \partial B| \cdot d(B) = |\partial F / \partial C| \cdot d(C)$$

Теперь для оценки погрешности получим  $d(F) \leq N \cdot dX$

Здесь N – количество равных вкладов, окончательно будем иметь оценку

$$d(F) / N \leq dX \Rightarrow d(F) / [N \cdot |\partial F / \partial A|] \leq d(A) \Rightarrow d(F) / [N \cdot |\partial F / \partial B|] \leq d(B) \Rightarrow \dots$$

#### **Метод оптимизации**

- Суть метода – все погрешности величин считаются разными и их вклады тоже считаются разными. Тогда существует бесконечное множество решений задачи с разным выбором величин погрешностей. Для того чтобы выбрать 1 решение, мы должны ввести дополнительное условие – критерий оптимизации. Это условие должно определять (математически как max или min) наилучший вариант выбора величин погрешностей. Такую задачу называют задачей оптимизации. Если критерий будет линейным, то это линейная оптимизация. В противном случае мы имеем дело с нелинейной оптимизацией. Разработаны специальные методы численного решения таких задач, которые изучаются в курсе «Исследование операций».

## Лекция №2

### Раздел №1. Численные методы алгебры

#### Тема 1.2 Численные методы решения уравнений

Содержание : Решение нелинейных уравнений. Определение существования корня на отрезке. Локализация (отделение корней). Уточнение корней. Конечные методы решения нелинейного уравнения. Метод дихотомии. Метод хорд. Сравнительная характеристика методов. Итерационные методы. Сходимость итерационного метода, принцип сжимающихся отображений. Метод простых итераций. Метод Ньютона. Итерационный метод хорд. Метод Чебышева. Модифицированный метод Ньютона. Сравнительная характеристика методов. Численное решение системы нелинейных уравнений. Векторно-матричная форма записи систем нелинейных уравнений. Метод простых итераций. Метод Ньютона. Метод градиентного поиска. Сравнительная характеристика методов.

#### Решение нелинейных уравнений

Во многих научных и инженерных задачах возникает необходимость решения нелинейных уравнений вида:  $f(x)=0$ . Решение такого уравнения принято называть корнем. Решить уравнение с заданной погрешностью  $\varepsilon$  фактически означает указать значение корня  $X$  такое, что в интервал неопределенности корня  $X \pm \varepsilon$  попадает и точка пересечения графика функции  $f(x)$  с осью  $x$ . Эта точка и есть точное значение корня.

Численное решение нелинейного уравнения обычно проводят в два этапа. На первом этапе необходимо отделить корни нелинейного уравнения, т.е. найти такие интервалы изменения переменной  $x$ , где расположен только один корень. По сути дела, на этом этапе находят приближённое значение корней с погрешностью равной половине длины интервала (интервал неопределенности корня). Приближённое значение корня при этом выбирается в центре данного интервала. Нередко отделение корней удаётся провести, не обращаясь к математическим методам и алгоритмам, на основании физического смысла задачи или из анализа её упрощённой модели. На втором этапе проводят уточнение отделённых корней, т.е. находят корни с заданной точностью, для этого известен богатый набор алгоритмов, например, метод дихотомии рассмотренный ниже.

#### **Вывод:**

Процесс решения нелинейных уравнений разбивается на два этапа:

1. Отделение корней
2. Уточнение корней

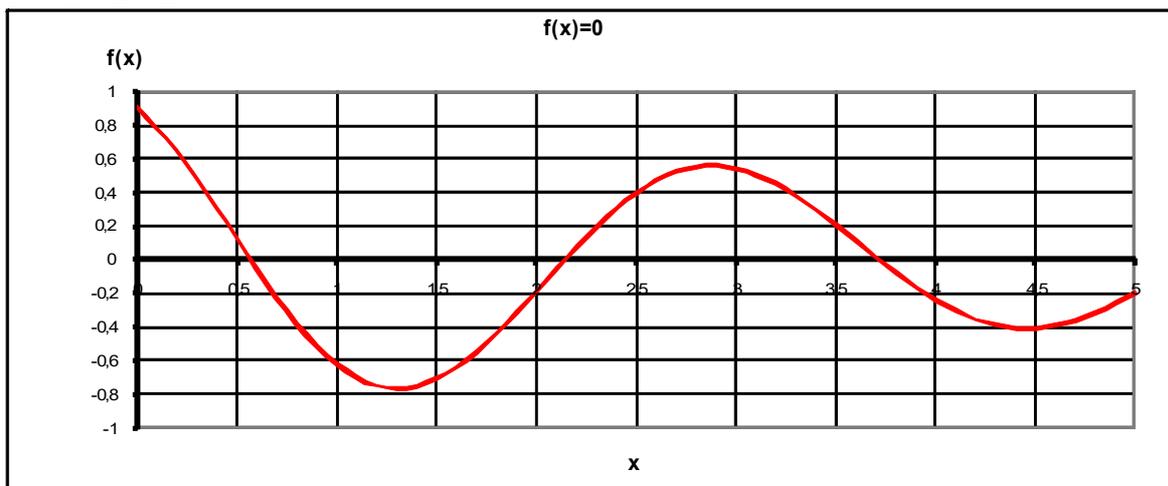
Определение существования корня на отрезке - локализация (отделение корней)

Отделить корни – задать такие отрезки, на которых корень существует и он единственный. Основная проблема задачи отделения – возможность наличия нескольких (в принципе бесконечного количества) корней.

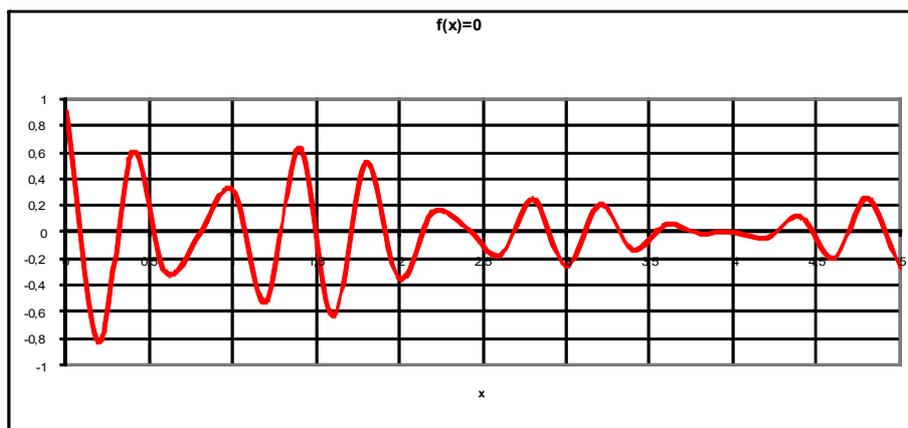
Рассмотрим два наиболее употребительных на практике метода

отделения корней:

### А) Графический метод.

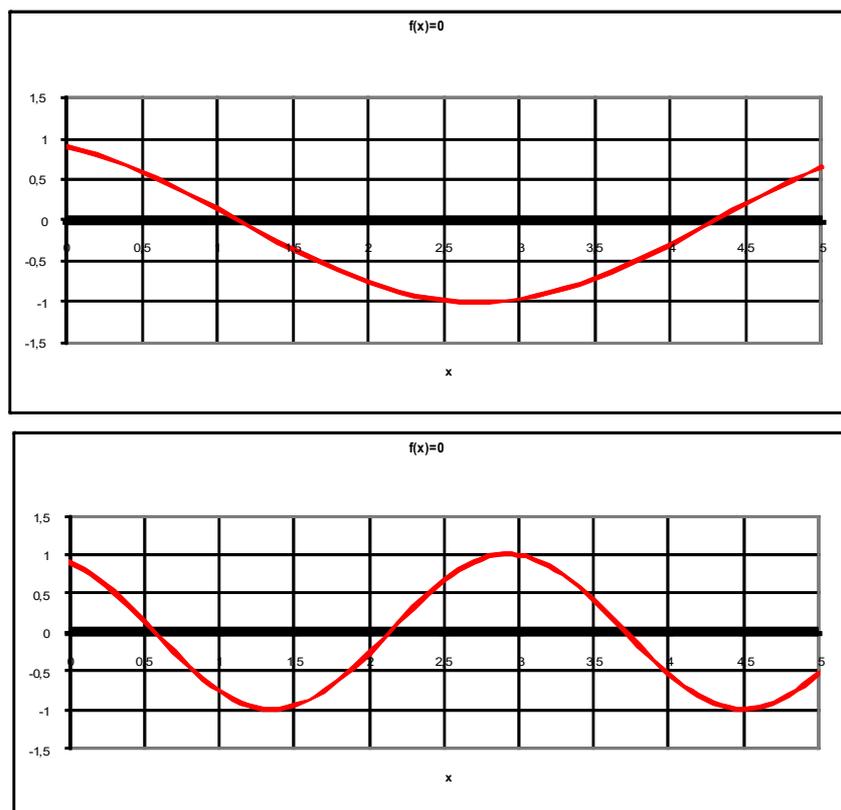


Суть метода заключается в построении графика функции  $f(x)$  для нелинейного уравнения вида  $f(x)=0$ . На графике выделяются отрезки, на которых есть один единственный корень (точка пересечения функцией оси X). При необходимости выяснить поведение функции строится график части отрезка в увеличенном масштабе.



### Б) Приближенный аналитический метод.

Аналитический метод основан на теореме математического анализа – если непрерывная функция меняет знак на концах отрезка  $[a,b]$  (т.е.  $f(a)*f(b) < 0$ ), то на этом отрезке обязательно существует корень уравнения  $f(x)=0$ . Используя теорему можно построить простой метод отделения корней на конечном отрезке – нужно разделить весь отрезок на большое число маленьких отрезков и проанализировав произведение функции на концах отрезков  $f(a_i)*f(b_i)$  выделить отрезки, где это произведение отрицательно или равно нулю (корень в конце отрезка). К сожалению, существуют возможности поведения функции  $f(x)$ , которые приведут в этом алгоритме к потере корней:



В приведенных примерах:

- В первом случае функция не меняет знак, но корни есть – их четное количество 2,4,6 ....
- Во втором случае функция меняет знак, но корней 3 а не один. Возможны так же варианты 5, 7, 9 и другого нечетного количества корней.

В принципе математика не дает общего метода отделения корней для любого нелинейного уравнения, хотя существует частные методы отделения корней некоторых классов функций  $f(x)$  – тригонометрические, полиномы. Однако для практического использования можно сформулировать не абсолютно точный, но достаточно эффективный метод, основанный на простейшем алгоритме.

Приведем такой примерный алгоритм отделения корней аналитически:

1. задается отрезок  $[a,b]$ , на котором необходимо отделить корни функции  $f(x)$ .
2. задается начальное значение  $n$  и строится сетка  $x_i = a+ih$ ,  $h=(b-a)/n$ ,  $i=0,1,2,\dots,n$ . Сетка делит отрезок на  $n$  частей с помощью  $n+1$  точки.
3. вычисляется значение функции  $f(x_i)=f_i$  и вычисляется произведение значений функции на концах отрезка  $f_i \cdot f_{i+1}$ , где  $i = 0,1,2,\dots,n-1$ .
4. подсчитываем количество отрицательных произведений и запоминаем отрезки, где произведение отрицательно или равно 0. Число таких отрезков  $k$ . Каждый из таких отрезков должен содержать хотя бы 1 корень.
5. увеличиваем  $n$  в два раза и повторяем процедуру. Получаем новое количество отрезков (корней)  $k_1$ .

$$6. \text{ Если } \begin{cases} k < k_1 \Rightarrow k = k_1 \text{ и повторяем п.5} \\ k = k_1 \Rightarrow \text{выход} \end{cases}$$

Данный алгоритм теоретически может пропустить корень, но практика показала его эффективность и достаточную точность.

### Уточнение корней

Методы уточнения корней делятся на конечные и итерационные. Конечные методы основаны на идее уменьшения интервала неопределенности на каждом шаге с помощью какого-либо процесса.

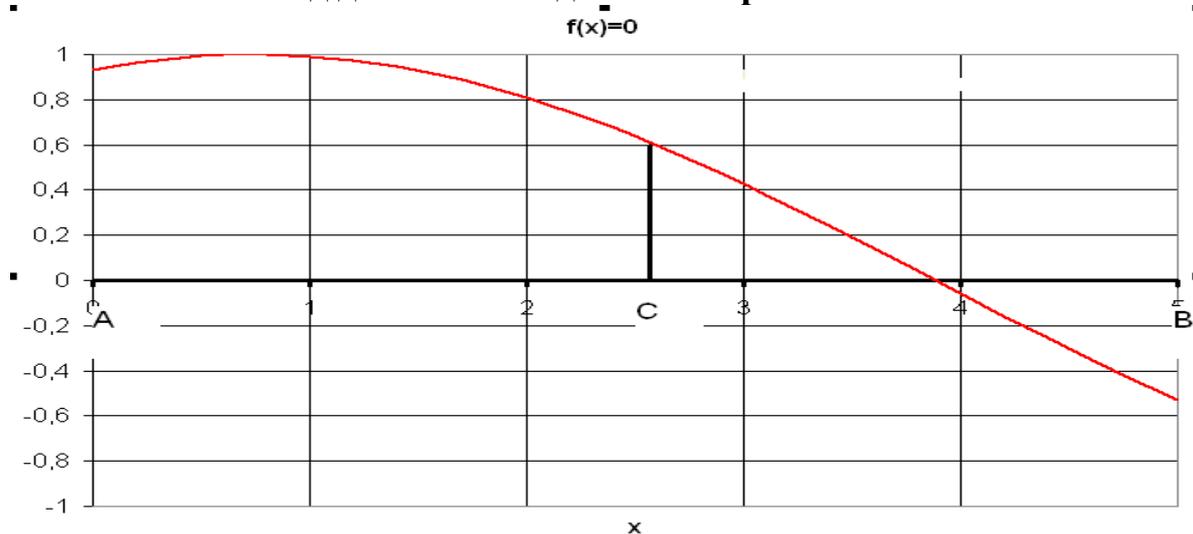
Суть итерационного метода в построении итерационной формулы и вычисления итерационной последовательности такой, чтобы эта последовательность сходилась к заданному корню.

Конечный метод построен таким образом, что можно заранее оценить количество итераций, которое потребуется для уточнения корня с заданной погрешностью. Для итерационного метода это не возможно. К конечным методам можно отнести метод «дихотомии» (деления отрезка пополам) и конечный вариант метода хорд.

### Конечные методы решения нелинейного уравнения

#### Метод дихотомии

#### Метод дихотомии – деления отрезка пополам



**Суть метода:** Пусть задан отрезок  $[a, b]$  на котором отделен корень уравнения  $f(x)=0$ . Если разделить этот отрезок пополам точкой  $C$ , то можно проверить знак произведения  $F(a)*F(c)$  и выбрать ту часть отрезка, где это произведение отрицательно и находится корень. Таким образом, отрезок неопределенности корня уменьшится в 2 раза. Это 1 итерация метода. За  $n$  итераций отрезок уменьшится в  $2^n$  раз. Если необходимо получить результат с абсолютной погрешностью  $\epsilon$ , то можно оценить количество итераций  $n$

$$(b-a)/2^n < \epsilon \Rightarrow (b-a)/\epsilon < 2^n \Rightarrow n > \ln((b-a)/\epsilon) / \ln(2)$$

**Алгоритм метода дихотомии:**

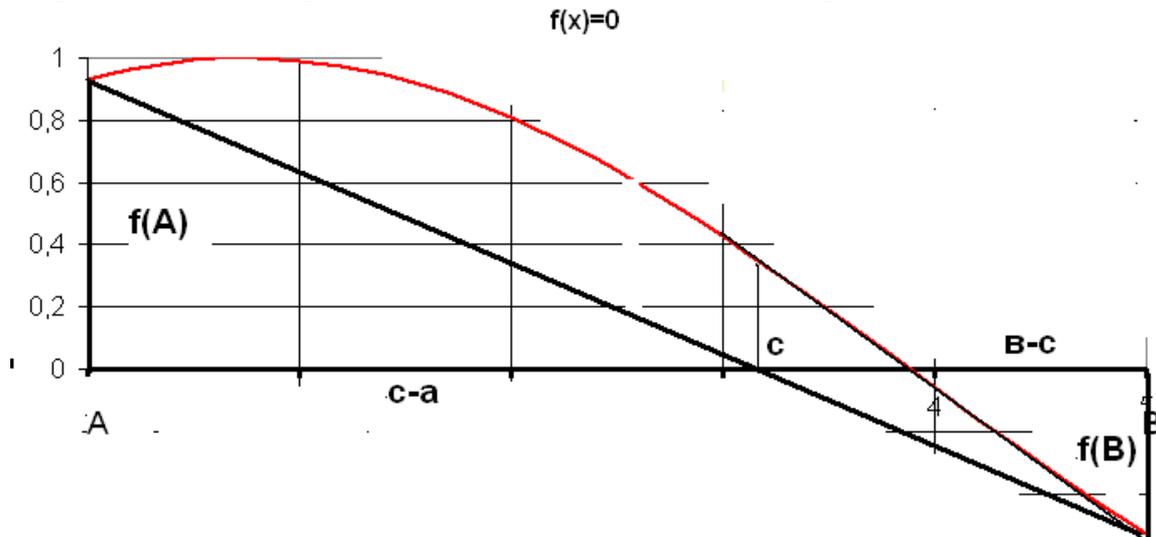
1. Задаем уравнение  $F(x)=0$ , отрезок для корня  $[a,b]$ , погрешность  $\varepsilon$ .
2. Ищем  $c=(a+b)/2$  и значение произведения  $F(a)*F(c)$ .
3. Если  $F(a)*F(c)<0 \Rightarrow a=c$ ;  $b=c$ , при  $F(a)*F(c)>0 \Rightarrow a=c$ ;  $b=b$ , при  $F(c)=0$   $a=b=c$ .
4. Находим оценку погрешности  $d=|b-a|/2$ . Если  $d>\varepsilon$  то возвращаемся в п.2, в противном случае найден корень  $X_k = c$  при заданной погрешности  $d$ .

### Метод хорд

**Суть метода:** Если в методе дихотомии выбор точки  $C$  изменить на точку пересечения хорды с осью  $x$ , то мы получим новый более быстрый конечный метод. Найдем значение точки  $C$  – пересечения хорды с осью  $x$ . Из правила подобия треугольников получаем 2 формулы вычисления:

$$C = A - f(A) * (b-a) / [f(b) - f(a)] \text{ или } C = B - f(B) * (b-a) / [f(b) - f(a)]$$

Обе эти формулы должны давать один и тот же результат, но с точки зрения реализации итераций метода они отличаются кардинально.



Проблемой такого метода является тот факт, что как правило работает только одна из формул. Это значит, что один из концов отрезка всегда остается неподвижным. Можно легко определить какой именно – например на вышележащем рисунке хорда отделяет корень на отрезке  $[C, B]$ , следовательно новая хорда вновь будет отсекает точку  $C$  слева от корня, хотя и гораздо ближе к нему. Таким образом, если мы будем оценивать отрезок неопределенности как  $[C, B]$ , то это будет явно неправильно – длина этого отрезка не будет стремиться к нулю, хотя как видно из рисунка точка  $C$  к корню быстро приближается. Это приводит к 2 характерным особенностям конечного метода хорд:

- На первой итерации необходимо выбрать неподвижный конец отрезка и соответствующую формулу. Далее вычисления ведутся только по этой формуле, а проверять условие  $F(a)*F(c)<0$  нет необходимости.
- Так как один конец отрезка не меняется, то точка  $C$  образует быстро сходящуюся последовательность значений  $C_1, C_2, C_3, C_4, \dots$ . Для оценки по-

грешности здесь используют косвенную оценку (как правило используемую в итерационных методах)  $d \approx |C_{k+1} - C_k|$ .

### Сравнительная характеристика методов

Конечные методы можно сравнивать по нескольким важным характеристикам:

1. Количеству итераций.
2. Сложности алгоритма.
3. Удобства реализации.

С точки зрения 1-го критерия метод дихотомии уступает методу хорд, который сходится быстрее. По другим характеристикам метод дихотомии существенно лучше. Его алгоритм прост, надежен и легко реализуем. Первый критерий становится важен только в задачах критичных по времени вычисления корня. В некоторых задачах корни вычисляются многократно (тысячи или десятки тысяч раз), тогда метод хорд имеет некоторые конкурентные преимущества.

### Итерационные методы. Сходимость итерационного метода, принцип сжимающихся отображений

Итерационные вычисления производятся по итерационным формулам, которые бывают двух видов:

- явная формула  $x=f(x)$
- неявная формула  $F(x)=0$ .

Неявные формулы могут быть преобразованы к явному виду:  $x=C*F(x)+x$ , где  $C$ - константа, которая выбирается из условия сходимости метода. Явная формула может быть представлена в виде рекуррентной формулы, когда элемент  $x_k$  последовательности выражается в виде  $x_k =F(x_{k-1}, x_{k-2} \dots)$ . Результатом вычисления по такой формуле является итерационная последовательность  $x_k$ . Пределом последовательности является решение нашего уравнения.

Алгоритм вычисления по итерационной формуле:

1. Определение начального приближения из условия сходимости.
2. Организация вычисления последовательности по правилу:  $x_1 =f(x_0) \Rightarrow x_2 =f(x_1) \Rightarrow \dots x_n =f(x_{n-1})$
3. Прерывание вычислений производится при достижении **косвенной оценки** заданной погрешности  $\varepsilon$ , **формула косвенной оценки** :  $|x_{k+1} - x_k| < \varepsilon$

В данном методе формула выражается в явном виде  $x=\varphi(x)$ . Условие сходимости для этого метода определяется с помощью теоремы о сжимающем отображении (если отображение  $\varphi(x)$  удовлетворяет условию сжимаемости, то итерационная последовательность будет сходящейся).

### Метод простых итераций

В одномерном случае будет простая формула условия сходимости для проверки начального приближения  $x_0$ :  $|\varphi'(x_0)| < 1$ . В простейшем случае  $\varphi(x)= C*F(x)+x$ , где  $C$  – некая константа. Тогда  $\varphi'(x)= C*F'(x)+1$ . Если  $F'(x)>0$ , то  $-2/F'(x)<C<0$ , если  $F'(x)<0$ , то  $-2/F'(x)>C>0$ . Выбирая  $C$  можно практически всегда

удовлетворить условие сходимости.

**Замечание:** Следует учитывать, что итерационная последовательность может сходиться к искомому корню, другому корню или расходиться (уход значений на бесконечность). Нам необходим только первый вариант. Для этого нужно, чтобы вокруг искомого корня была область, в которой все точки удовлетворяют условию сходимости. Это несколько более сложное условие, чем  $|\varphi'(x_0)| < 1$ , которое определяется только для 1 точки  $x_0$ .

Рассмотрим это замечание на примере использования метода простой итерации для вычисления корня из  $x=2$  по 2-м итерац. формулам  $y_{i+1}=y_i/x$  и  $y_{i+1}=(x+y_i/x)/2$ :

<b>Пример:</b>	$y=x*x \Rightarrow$	$x=y/x$	$\Rightarrow$	$x=(x+y/x)/2$
<b>X=1,73205081</b>		<b>y=</b>	<b>3</b>	
	<b>x0=</b>	<b>1</b>	<b>3</b>	<b>1</b>
		<b>3</b>	<b>1</b>	<b>2</b>
		<b>1</b>	<b>3</b>	<b>1,75</b>
		<b>3</b>	<b>1</b>	<b>1,73</b>
		<b>1</b>	<b>3</b>	<b>1,73</b>
		<b>3</b>	<b>1</b>	<b>1,73</b>
		<b>1</b>	<b>3</b>	<b>1,73</b>
		<b>3</b>	<b>1</b>	<b>1,73</b>
		<b>1</b>	<b>3</b>	<b>1,73</b>
		<b>3</b>	<b>1</b>	<b>1,73</b>
		<b>1</b>	<b>3</b>	<b>1,73</b>
		<b>3</b>	<b>1</b>	<b>1,73</b>
		<b>1</b>	<b>3</b>	<b>1,73</b>
		<b>3</b>	<b>1</b>	<b>1,73</b>
		<b>1</b>	<b>3</b>	<b>1,73</b>

Таким образом, производная первой итерационной формулы  $x=y/x$  вблизи корня близка к единице и метод не сходится. Если формулу преобразовать к виду  $x=(x+y/x)/2$ , то производная будет уже близка к нулю и метод будет не только сходиться, но и сходиться максимально быстро. Чем ближе производная к нулю, тем быстрее сходимость. Это свидетельствует о необходимости правильного выбора итерационной формулы и некоторого начального исследования области вокруг предполагаемого корня нелинейного уравнения. Теперь можно сформулировать итоговый алгоритм метода простой итерации:

1. Задается отрезок отделения корня и нелинейное уравнение.
2. Строится итерационная формула.
3. Исследуется сходимость построенной итерационной формулы на заданном отрезке. При необходимости формула корректируется. Для упрощения часто проверяется условие сходимости на концах отрезка, если там все в порядке, то считается, что на всем отрезке условие сходимости тоже выполняется.
4. Для выбранной формулы выбирается подходящее начальное приближение  $x_0$ . Для упрощения часто выбирается середина отрезка.
5. Производится вычисление итерационной последовательности до тех пор пока не будет выполнено условие выхода, которое определяется как  $d < \epsilon$ , где  $d = |x_{k+1} - x_k|$  (формула косвенной оценки). Тогда  $x_{k+1}$  - искомый корень нелинейного уравнения.

Данный алгоритм применим к любому итерационному методу, отличаются только итерационные формулы и условие сходимости. Более того, можно доказать что рассматриваемые далее итерационные методы – частные случаи метода простой итерации со специально выбранными итерационными формулами.

### Метод Ньютона

Метод простой итерации — наиболее простой, но не самый быстро сходящийся итерационный метод. Более быстрый итерационный метод уточнения корней называется методом касательных или методом Ньютона. Его итерационная формула :  $x_{k+1} = x_k - F(x_k)/F'(x_k)$ , здесь  $F(x_k)$  — определяется из стандартного вида нелинейного уравнения  $F(x)=0$ .

Условия сходимости для метода Ньютона можно получить, если считать его частным случаем метода простой итерации. Получаем формулу:

$$\text{При } \varphi(x) = x - F/F' \Rightarrow |F'' \cdot F / (F')^2| \leq 1$$

Метод Ньютона так же имеет особое название — метод касательных, что определяется геометрическим смыслом его итерационных формул. Если метод хорд геометрически описывается хордой к графику функции  $F(x)$ , то производную  $F'$  можно графически интерпретировать как касательную к этому графику. В этом случае метод Ньютона рассматривается как предельный случай метода хорд и фактически как оптимальный (по скорости сходимости) вариант итерационного метода хорд.

### Итерационный метод хорд

Из формулы метода хорд  $C = B - f(B) \cdot (b-a) / [f(b)-f(a)]$  можно построить итерационную формулу  $X_{k+1} = X_k - f(X_k) \cdot (X_k - X_{k-1}) / [f(X_k) - f(X_{k-1})]$

Это частный случай метода простой итерации, поэтому условие сходимости можно определить как в этом методе, но лучше использовать условие сходимости метода Ньютона, поскольку метод хорд (в пределе) сходится к методу Ньютона. Главная особенность метода хорд – необходимость 2-х начальных условий для расчета по итерационной формуле, что явно не удобно для расчета на ЭВМ.

### Метод Ньютона – Чебышева

Русский математик Чебышев предложил обобщить итерационные методы решения нелинейного уравнения путем вывода наилучших последовательных приближений. Пусть задано нелинейное уравнение:  $f(x)=0$  и существует корень этого уравнения, который обозначим как  $x_k$ . Разложение в ряд Тейлора в точке корня имеет вид:  $f(x_k)=0 = f(x) + f'(x) \cdot (x_k - x) / 1! + f''(x) \cdot (x_k - x)^2 / 2! + \dots$

В первом приближении разложения в ряд получается:

$$0 = f(x) + f'(x) \cdot (x_k - x) / 1! \Rightarrow x_k = x - f(x) / f'(x) - \text{это метод Ньютона. Для}$$

более точных оценок Чебышев предложил раскладывать в ряд не функцию  $f$ , а обратную к ней функцию  $x=C(f)$ . Тогда такое разложение имеет вид:

$$x_k = x + dC/df \cdot (f(x_k) - f(x)) / 1! + d^2 C/df^2 \cdot (f(x_k) - f(x))^2 / 2! + \dots$$

Использование данной формулы, дает принципиальную возможность вычислить любое, сколь угодно точное приближение к корню.

Рассмотрим пример вычисления первого приближения:

$$x_k = x + dC/df * (f(x_k) - f(x)) / 1! = x_k = x + dC/df * (0 - f(x)) \Rightarrow dC/df = 1/f' \\ (\text{формула производной обратной функции}) \Rightarrow x_k = x - f(x) / f'(x).$$

В терминах рекурсивных формул имеем:  $x_k = x_{k-1} - f(x_{k-1})/f'(x_{k-1})$ . Таким образом, в первом приближении метод Чебышева так же дает формулу метода Ньютона. Рассмотрим пример вычисления второго приближения:

$$x_k = x - \frac{f(x)}{f'(x)} + \frac{C''}{2!} f^2(x)$$

$$C'' = \frac{d}{df}(C') = \frac{d}{df}\left(\frac{1}{f'}\right) = -\frac{1}{(f')^2} \cdot \frac{df'}{df} = -\frac{1}{(f')^2} \cdot \frac{df'}{dx} \Rightarrow C'' = -\frac{f''}{(f')^3}$$

$$\frac{dx}{df} = -\frac{1}{(f')^2} f'' \cdot C' = -\frac{1}{(f')^3} \cdot f''$$

$$x_k = x - f(x) / f'(x) - f''(x) * (f(x))^2 / (2 * (f'(x))^3) \Rightarrow$$

$$\Rightarrow x_k = x_{k-1} - f(x_{k-1}) / f'(x_{k-1}) - f''(x_{k-1}) * (f(x_{k-1}))^2 / (2 * (f'(x_{k-1}))^3)$$

Второе же приближение можно уже назвать формулой Чебышева.

Рассмотрим теперь условия сходимости для этих приближений.

Так как метод Ньютона – это 1-е приближение формулы Чебышева, то формула сходимости останется справедливой и для последующих, более точных приближений (обратное неверно – если метод Ньютона не сходится, то следующее приближение может в этой точке сходиться). Таким образом, на практике можно сходимость метода Чебышева оценивать по формуле для метода Ньютона.

Замечание: Можно считать метод Чебышева частным случаем метода простой итерации и вывести формулу точного условия сходимости, но эта формула сложна и для практических целей не удобна.

### Комбинированный метод

Для реализации итерационных методов в ЭВМ некоторые из представленных итерационных методов модернизировали. Так, например, существует метод Ньютона с постоянной производной. В формуле Ньютона самое сложное для реализации на ЭВМ – вычислить производную. Ее приходится считать на каждой итерации. С другой стороны вблизи корня (при малых изменениях  $x$ ) производная меняется слабо. Это дало основание упростить алгоритм – производная вычисляется в начальной точке, а дальше считается неизменной. Алгоритм сходится медленнее, но вычисления на ЭВМ в результате часто даже ускоряются. Формула для этого метода  $x_{k+1} = x_k - F(x_k)/F'(x_0)$

Другая модификация метода Ньютона заключается в установленном математиками факте отделения корня результатами вычисления по формулам Ньютона и хорд. Это значит, что если начальное приближение в обеих формулах одинаковое, то результат вычисления дает отрезок (между  $x_k$  для Ньютона и хорд) на котором точно есть корень. Середина этого отрезка уточняет корень лучше чем оба используемых методов, кроме того метод получает дополнительную сходимость. Описанная процедура уточнения корня с помощью двух методов сразу получила название комбинированного метода. Он отличается исключительной скоростью уточнения.

## Сравнительная характеристика итерационных методов

Итерационные методы можно сравнивать по нескольким важным характеристикам:

4. Количеству итераций.
5. Сложности алгоритма.
6. Удобства реализации.

С точки зрения 1-го критерия можно построить некий рейтинг методов в порядке убывания их скорости вычисления – метод Чебышева, комбинированный метод, метод Ньютона, метод хорд, метод простой итерации. Конечно это некая усредненная характеристика, так как большинство из этих методов можно считать частным случаем одного метода – метода простой итерации. Можно например построить формулу в методе простой итерации, которая для частного случая конкретного уравнения будет сходиться быстрее формулы Чебышева. Однако практически это не выгодно, лучше иметь одну программу для разных уравнений. Таким образом, 2 и 3 критерий при реализации в ЭВМ часто более важны.

На практике компромисс между критериями чаще выигрывает метод Ньютона и его производный комбинированный метод. Метод Чебышева остается лучшим вариантом для ручных вычислений, когда не желательно иметь более 2 итераций. В простейших задачах используется и метод простой итерации.

## Численное решение системы нелинейных уравнений

### Векторно-матричная форма записи систем нелинейных уравнений

В случае решения систем нелинейных уравнений мы имеем несколько нелинейных уравнений, связанных между собой:

$$F_1(x_1, x_2, x_3, \dots) = 0, F_2(x_1, x_2, x_3, \dots) = 0, F_3(x_1, x_2, x_3, \dots) = 0, \dots$$

Многоточие обозначает, что число переменных  $m$  и число уравнений  $n$  не фиксировано и для разных систем нелинейных уравнений (СНУ) может быть различным (в общем случае  $m \neq n$ ). В сокращенной форме это можно записать как:  $F_i(x_j) = 0$ , где  $i = 1, 2, \dots, n$ ;  $j = 1, 2, \dots, m$ .

В алгебре используются так же понятия вектор и матрица. Совокупность переменных  $x_j$  образует вектор неизвестных  $X$ , а совокупность функций  $F_i(X)$  – вектор функций  $F$ . Тогда  $m, n$  — это размерности векторов. В такой векторной форме СНУ запишется еще проще  $F(X) = 0$ .

Теперь мы видим, что в векторной форме СНУ аналогично 1 нелинейному уравнению, поэтому логично попытаться распространить на СНУ методы решения 1-го уравнения. Конечные варианты здесь обычно не используются, но аналогии методов простой итерации и Ньютона активно используются. Задача отделения корней СНУ сложна и имеет решение только в частных случаях. Обычно область где отделен 1 корень называют областью унимодальности.

### Методы решения СНУ

Будет считать, что корни отделены, тогда существует 2 варианта решения СНУ:

- 1) Использование итерационных методов.

2) Сведение к задаче нелинейной оптимизации.

### Метод простых итераций

При использовании метода простой итерации необходимо (по аналогии с одномерным вариантом) :

- Представить СНУ из обычного вида  $F_i(x_j)=0$  в итерационный  $\Phi_i(x_j)=x_i$ .
- Проверить условие сходимости (здесь оно бывает гораздо сложнее, но основывается на той же теореме о сжимающих отображениях) и выбрать вектор начальных приближений  $X_0$ .
- Найти последовательность  $X_k$  векторов по итерационной формуле (начая с  $X_0$ )  $X_k = \Phi(X_{k-1})$ .
- Остановить расчет последовательности  $X_k$  векторов при достижении заданной погрешности по условию прерывания  $\| X_k - X_{k-1} \| \leq \varepsilon$  (здесь вместо модуля используется обозначение нормы, о нормах векторов поговорим позже).

### Метод Ньютона

Метод Ньютона будет отличаться от простой итерации итерационной формулой :  $X_k = X_{k-1} - F(X_{k-1}) * (F')^{-1}(X_{k-1})$ . Здесь  $F'$  - вектор функции производных, с учетом вектора неизвестных получаем матрицу, которая называется **матрицей Якоби**, а ее определитель **якобианом**. Степень -1 означает, что умножаем мы на обратную к матрице Якоби матрицу (это аналог деления для матриц). В алгебре определена операция умножения матриц, но нет операции деления матриц. С другой стороны, для существования обратной матрицы нужно, чтобы ее определить (якобиан) был не равен нулю. Фактически это условие и будет условием сходимости метода Ньютона (оно более жесткое, чем условие при сжимаемости отображений).

### Метод градиентного поиска

Сведение к задаче нелинейной оптимизации заключается в том, что СНУ вида

$$F_i(x_j)=0, \text{ где } i=1,2\dots n; j=1,2\dots m$$

можно заменить  $\sum F^2_i(x_j)=0$ . Тогда это аналог задачи поиска минимума векторной многомерной функции  $F^2(X)$ . То есть решение оптимизационной задачи и СНУ совпадают в области унимодальности. Для решения же задачи нелинейной оптимизации есть свои методы — градиентные, релаксационные, случайного поиска. Наиболее быстрым считается метод градиентного поиска. Градиент функции — вектор, направление которого совпадает с направлением наискорейшего роста многомерной функции (локального в точке вычисления). Если все время идти по градиенту, то дойдешь до максимума, если против — до минимума. Разница между разными градиентными методами различается только методами выбора размера шага по направлению (или против направления) градиента. По смыслу градиентный метод — аналог метода Ньютона. Релаксационный метод заключается в фиксации всех переменных кроме одной, тогда задача сводится к одномерной, где есть например аналог метода деления отрезка пополам. Затем фиксируется другой набор переменных и т. д. Таким образом, здесь мы ходим вдоль координатных осей. При случайном поиске генерируются множество случайно разбросанных в области

унимодальности точек. Вычисляются значения функции в них. Выбирается точка, где функция минимальна. С точки зрения теории вероятностей при увеличении числа точек, результат будет стремиться к минимуму.

### Сравнительная характеристика методов

Методы решения СЧУ можно анализировать так же как и для одномерного случая. Метод Ньютона более быстрый, но и более сложный. На практике метод простой итерации часто требует слишком много операций (итераций) и метод Ньютона более распространен. Оптимизационные методы в целом аналогичны СЧУ и используются потому, что задача нелинейной безусловной оптимизации более изучена и разработано множество алгоритмов и программ. Фактически можно выбирать по скорости и простоте реализации.

## Лекция №3

### Раздел №1. Численные методы алгебры

#### Тема 1.2 Численные методы решения уравнений

Содержание: Численное решение систем линейных уравнений (СЛАУ). Векторно-матричная форма записи СЛАУ. Существование и единственность решения СЛАУ. Обусловленность СЛАУ. Конечные методы решения СЛАУ. Метод Гаусса. Прямой и обратный ход. Выбор главного элемента. Метод полного исключения Жордана. Вычисление определителя и обратной матрицы. Метод Халецкого. Сравнительная характеристика методов. Итерационные методы решения СЛАУ. Сходимость итерационного метода. Метод простых итераций. Метод Зейделя. Метод невязки (координатной релаксации). Сравнительная характеристика методов.

### Численное решение систем линейных уравнений (СЛАУ)

Системой Линейных Алгебраических Уравнений (СЛАУ) будем называть систему уравнений следующего вида:

$$A_{11} \cdot X_1 + A_{12} \cdot X_2 + A_{13} \cdot X_3 + A_{14} \cdot X_4 + \dots + A_{1n} \cdot X_n = B_1$$

$$A_{21} \cdot X_1 + A_{22} \cdot X_2 + A_{23} \cdot X_3 + A_{24} \cdot X_4 + \dots + A_{2n} \cdot X_n = B_2$$

$$A_{31} \cdot X_1 + A_{32} \cdot X_2 + A_{33} \cdot X_3 + A_{34} \cdot X_4 + \dots + A_{3n} \cdot X_n = B_3$$

.....

$$A_{n1} \cdot X_1 + A_{n2} \cdot X_2 + A_{n3} \cdot X_3 + A_{n4} \cdot X_4 + \dots + A_{nn} \cdot X_n = B_n$$

### Векторно-матричная форма записи СЛАУ

Такую систему уравнений можно записывать в сокращенном виде, используя как минимум 2 формы:

1. С использованием знака суммы  $\sum_{j=1}^n a_{ij} x_j = b_i$
2. С использованием матриц:  $A \cdot X = B$ , здесь  $A$  – матрица системы,  $X$  – вектор неизвестных,  $B$  – вектор правой части. Иногда используют расширенную матрицу системы  $A'$  – которая образуется дописыванием вектора  $B$

справа к матрице  $A$ .

Матрица системы:

$$\begin{array}{cccccc} A_{11} & A_{12} & A_{13} & A_{14} & \dots & A_{1n} \\ A_{21} & A_{22} & A_{23} & A_{24} & \dots & A_{2n} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ A_{n1} & A_{n2} & A_{n3} & A_{n4} & \dots & A_{nn} \end{array}$$

Расширенная матрица системы:

$$\begin{array}{ccccccc} A_{11} & A_{12} & A_{13} & A_{14} & \dots & A_{1n} & B_1 \\ A_{21} & A_{22} & A_{23} & A_{24} & \dots & A_{2n} & B_2 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ A_{n1} & A_{n2} & A_{n3} & A_{n4} & \dots & A_{nn} & B_n \end{array}$$

### Существование и единственность решения СЛАУ. Обусловленность СЛАУ

В алгебре для линейных уравнений вводится понятие линейной зависимости.

**О. Линейной комбинацией** уравнений  $T$  и  $M$  называют уравнение полученное по формуле  $C_1 * T + C_2 * M$ , где  $C_1$  и  $C_2$  постоянные коэффициенты.

Если одно из уравнений системы может быть представлено линейной комбинацией остальных, то такая система называется линейно-зависимой. Линейная зависимость системы определяется вырождением матрицы системы.

**О. Матрица  $A$  будет вырожденной**, если определитель этой матрицы равен нулю  $\Rightarrow \det(A)=0$ .

Линейная зависимость СЛАУ означает, что некоторые из уравнений системы «лишние» и могут быть без ущерба для решения удалены из системы. В дальнейшем будем считать, что все зависимые уравнения уже удалены из системы, хотя проверку неравенства нулю определителя матрицы системы будем планировать перед решением СЛАУ. В алгебре доказывается, что для линейно независимой СЛАУ, в которой  $N$  уравнений и  $M$  неизвестных:

- При  $N > M$  (число уравнений больше числа неизвестных) – решения СЛАУ не существует.
- При  $N < M$  (число уравнений меньше числа неизвестных) – существует бесконечно много решений СЛАУ.
- При  $N = M$  (число уравнений равно числу неизвестных) – существует единственное решение СЛАУ.

Нас интересует только последний вариант, поэтому будем считать, что число уравнений равно числу неизвестных. Таким образом, рассмотрим методы численного решения СЛАУ с квадратной матрицей системы, которая является линейно-независимой. Помимо вырождения на решение системы влияет так же

обусловленность матрицы системы. Если система линейно-зависима, то обусловленность матрицы системы равна нулю. Чем больше модуль определителя матрицы системы, тем лучше (или больше) обусловленность. Есть разные коэффициенты определяющие обусловленность. Например отношение определителя к норме матрицы. Величина обусловленности влияет на точность решения (так как любое численное решение имеет погрешность). Поэтому для плохо обусловленной системы определитель не равен нулю и решение как бы существует, но при численном решении получаются слишком большие погрешности. Для итерационных методов плохая обусловленность системы ведет к увеличению числа итераций. И тот и другой случай мешает решению, поэтому для плохо обусловленных систем нужно в начале произвести их преобразование так, чтобы значение определителя увеличилось (норма при этом как правило не изменяется).

### Конечные методы решения СЛАУ

Для численного решения СЛАУ (аналогично нелинейных уравнениям) используют конечные и итерационные методы.

В конечном методе можно заранее точно указать количество операций, которое потребуется для решения СЛАУ.

### Метод Гаусса

Метод исключения Гаусса делится на два этапа: прямой и обратный ход. **Прямой** ход – с помощью эквивалентных преобразований (не изменяющих решения системы) исключаем элементы (делаем равными 0), лежащие ниже или выше главной диагонали. В общем случае необходимо получить треугольную матрицу. Тогда возможны 2 варианта: приведение матрицы системы к L или U виду.

**О. Главной диагональю** квадратной матрицы называют множество элементов с одинаковыми индексами  $A_{ii}$ . Эти элементы располагаются на диагонали матрицы от верхнего левого угла до правого нижнего.

**О. Матрица называется треугольной**, если все элементы, лежащие ниже главной диагонали равны нулю (**L –матрица**, от слова Low-нижний) или выше главной диагонали равны нулю (**U –матрица**, от слова Upper-верхний)

**Обратный** ход используется для определения неизвестных  $x_i$  в треугольной матрице системы.

В качестве преобразований, не изменяющихся решений системы, являются следующие преобразования **расширенной матрицы** системы:

3. Деление или умножение строки на любое постоянное число, кроме 0.
4. Сложение или вычитание любых двух строк.
5. Построение линейных комбинаций строк.
6. Перестановка строк.
7. Перестановка столбцов с одновременным изменением индексов переменных.

### Прямой и обратный ход

Рассмотрим процесс исключения прямого хода на примере получения L

матрицы:

$$\begin{array}{ccccccc}
 a_{11} & a_{12} & a_{13} & a_{14} & \dots & a_{1n} & b_1 \\
 a_{21} & a_{22} & a_{23} & a_{24} & \dots & a_{2n} & b_2 \\
 \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
 a_{n1} & a_{n2} & a_{n3} & a_{n4} & \dots & a_{nn} & b_n
 \end{array}$$

В данном случае необходимо исключить элементы, лежащие ниже главной диагонали. При исключении элемент нужно будет обратить в ноль. Исключения производятся по столбцам, при этом базовым или **главным** является диагональный элемент и его строка. Для L матрицы нужно начинать с 1-го столбца и исключать элементы ниже главной диагонали. Таким образом, номер столбца будет совпадать с номером главной строки, которая содержит главный диагональный элемент. Легко убедиться, что вариант исключения, приводящий к получению L матрицы из произвольной матрицы можно реализовать с помощью линейных преобразований следующего вида:

$$a_{2j} = a_{2j} - a_{1j} \cdot (a_{21}/a_{11}) \Rightarrow a_{kj} = a_{kj} - a_{1j} \cdot (a_{k1}/a_{11}) \Rightarrow a_{kj} = a_{kj} - a_{mj} \cdot (a_{km}/a_{mm})$$

и  $b_k = b_k - b_m \cdot (a_{km}/a_{mm})$ , где  $m = 1, 2, 3, \dots, n-1$ ;  $k = m+1, m+2, \dots, n$ ;  $j = m, m+1, \dots, n$

Для получения U – матрицы необходимо идти в обратную сторону:  
 $m = n, n-1, \dots, 3, 2$ ;  $k = m-1, m-2, \dots, 1$ ;  $j = m, m-1, \dots, 1$

Точность вычисления по методу Гаусса определяется процессами округления при этом, чем больше абсолютное значение используемого диагонального элемента, тем точнее будет вычисление. Если диагональный элемент = 0, то вычислять нельзя. Если он мал (плохая обусловленность), то результат будет неточным.

Рассмотрим обратный ход метода Гаусса. Пусть в результате исключения мы получим L матрицу.

$$\begin{array}{cccc}
 a_{11} & a_{12} & a_{13} & b_1 \\
 0 & a_{22} & a_{23} & b_2 \\
 0 & 0 & a_{33} & b_3 \\
 0 & 0 & 0 & b_4
 \end{array} \Rightarrow x_4 = b_4/a_{44} \Rightarrow x_3 = (b'_3 - a'_{34} x_4) / a'_{33} \Rightarrow$$

$$x_k = \frac{1}{a'_{kk}} \left( b'_k - \sum_{j=k+1}^n a'_{kj} x_j \right)$$

получили формулу обратного хода для L матрицы при  $k=n, n-1, n-2, \dots, 1$

Для U – матрицы похожая формула:  $x_k = \frac{1}{a'_{kk}} \left( b'_k - \sum_{j=1}^{k-1} a'_{kj} x_j \right)$  при  $k=1, 2, \dots, n-1, n$

### Выбор главного элемента

Существует модификация метода Гаусса за счет использования алгоритма выбора главного элемента (главный элемент – диагональный элемент, который используется при вычислениях). В процессе вычисления используются все элементы главной диагонали, как главные элементы. Поэтому, на главной

диагонали матричной системы желательно иметь max по модулю элементы в своих строках (столбцах). Выбор такого главного элемента может производиться либо перестановкой строк расширенной матрицы, либо перестановкой столбцов (во втором случае необходимо будет переименовывать переменные, поэтому чаще всего используется первый вариант – перестановка строк). Таким образом, метод Гаусса существует в двух видах: просто метод Гаусса и метод Гаусса с выбором главного элемента.

Пример. Решить методом Гаусса СЛАУ:

$$4x+3y-2z=20 \quad (a)$$

$$5x-1y+3z=-1 \quad (b)$$

$$7x+4y+6z=4 \quad (c)$$

Выпишем расширенную матрицу системы :

$$4 \quad 3 \quad -2 \quad 20$$

$$5 \quad -1 \quad 3 \quad -1$$

$$7 \quad 4 \quad 6 \quad 4$$

Шаг 1. Исключаем элементы 1-го столбца, лежащие ниже диагонали. Это 2-я и 3-я строка. Главный элемент — 4 (в 1-й строке и 1-м столбце) не равен нулю, решение возможно.

В соответствии с формулой Гаусса для исключения 5 во второй строке нужно заменить (b) на  $(b_1)=(b)-(a)*5/4$ . Для удобства умножим эту формулу на 4. Получим  $(b_1)=4*(b)-(a)*5$ . Получим  $(b_1) \quad 0 \quad -19 \quad 22 \quad -104$

В соответствии с формулой Гаусса для исключения 7 в 3 строке нужно заменить (c) на  $(c_1)=(c)-(a)*7/4$ . Для удобства умножим эту формулу на 4. Получим  $(c_1)=4*(c)-(a)*7$ . Получим  $(c_1) \quad 0 \quad -5 \quad 38 \quad -124$

$$4 \quad 3 \quad -2 \quad 20 \quad (a)$$

$$0 \quad -19 \quad 22 \quad -104 \quad (b_1)$$

$$0 \quad -5 \quad 38 \quad -124 \quad (c_1)$$

Шаг 2. Исключаем элементы 2-го столбца, лежащие ниже диагонали. Это 3-я строка. Главный элемент — -19 (в 2-й строке и 2-м столбце) не равен нулю, решение возможно.

В соответствии с формулой Гаусса для исключения -5 в 3 строке нужно заменить (c1) на  $(c_2)=(c_1)-(b_1)*(-5)/(-19)$ . Для удобства умножим эту формулу на 19. Получим  $(c_2)=19*(c_1)-(b_1)*5$ . Получим  $(c_2) \quad 0 \quad 0 \quad -612 \quad 1836$

$$4 \quad 3 \quad -2 \quad 20$$

$$0 \quad -19 \quad 22 \quad -104$$

$$0 \quad 0 \quad -612 \quad 1836$$

Получили треугольную матрицу, прямой ход выполнен. Такая матрица соответствует системе:

$$4x+3y - 2z = 20 \quad (a)$$

$$-19y + 22z = -104 \quad (b1)$$

$$-612z = 1836 \quad (c2)$$

Обратный ход. Из последнего уравнения (c2) выразим  $z = 1836/-612 = -3$ . Подставим это значение в (b1)  $-19y + 22*(-3) = -104 \Rightarrow -19y = -104 - 22*(-3) = -104 + 66 = -38 \Rightarrow y = 2$ .

Подставим  $z = -3, y = 2$  в (a)  $\Rightarrow 4x + 3*2 - 2*(-3) = 20 \Rightarrow 4x + 12 = 20 \Rightarrow 4x = 8 \Rightarrow x = 2$

Ответ: решение СЛАУ  $z = -3, y = 2, x = 2$ .

### Метод полного исключения Жордана

Жордан предложил исключительные элементы не только ниже главной диагонали, но и выше, в результате такого преобразования матрица становится полностью диагональной, т. е. все элементы, кроме диагональных равны 0.

Для такого исключения можно воспользоваться той же формулой Гаусса, но индексы меняются иначе:

$$m = 1, 2, \dots, n. \text{ при условии, что } k \neq m \text{ и } j = 1, \dots, n$$

Обратный ход для метода Жордана чрезвычайно прост:  $x_i = b'_i/a_{ii}$ , где  $i = 1, 2, \dots, n$ . Штрих означает, что это элементы расширенной матрицы, преобразованные после процедуры исключения.

### Вычисление определителя и обратной матрицы

Для решения некоторых матричных задач часто используют методы Гаусса и Жордана.

Вычисление определителя матрицы.

Так как исключения в методе Гаусса не меняют значение определителя матрицы, то с помощью прямого хода можно привести матрицу к треугольному виду L или U. В тоже время определитель треугольной матрицы равен произведению диагональных элементов. Следовательно, получаем алгоритм вычисления определителя для произвольной матрицы:

По данной формуле производим исключение элементов выше или ниже главной диагонали и перемножаем элементы главной диагонали

$$\det A = \prod_{i=1}^n a'_{ii}$$

Эта формула определяет значение определителя исходной матрицы.

### Вычисление элементов обратной матрицы:

По определению обратной матрицы  $AA^{-1} = E$ . Это произведение эквивалентно множеству систем уравнений для столбцов обратной матрицы. Таким образом, определение обратной матрицы сводится к решению систем линейных уравнений, где вектор неизвестных – один из столбцов обратной матрицы, а в правой части один из столбцов единичной матрицы. Матрица системы во всех уравнениях одна и та же. Для решения этих систем лучше использовать метод Жордана.

## Метод Халецкого

Суть метода Халецкого в разложении матричной системы в произведение матрицы  $U*L = A$ . При этом диагональные элементы одной из треугольных матриц равны 1. Для решения методом Халецкого необходимо получить матрицы разложения и воспользоваться обратным ходом:

$$A=U*L \Rightarrow A*X=B \Rightarrow U*(L*X)=B \Rightarrow Y=L*X \Rightarrow \begin{cases} UY = B \\ LX = Y \end{cases}$$

В методе Халецкого вместо 1 задачи решения СЛАУ получают 2 задачи решения СЛАУ, но обе эти задачи имеют треугольные матрицы систем и решаются только с помощью 2-х обратных ходов. Поскольку обратный ход имеет только 2 цикла вместо 3 циклов прямого хода, то такой вариант сулит значительные выгоды. Все определяется только затратами на построение L и U. Далее требуются следующие формулы обратного хода для треугольных матриц U и L:

$$\text{Для } U \text{ – матрицы: } y_k = \left( b_k - \sum_{j=1}^{k-1} U_{kj}x_j \right), \text{ для } L \text{ матрицы } x_k = \frac{1}{L_{kk}} \left( y_k - \sum_{j=k+1}^n L_{kj}x_j \right)$$

Очень интересен алгоритм вычисления элементов матриц разложения L и U. Диагональные элементы матрицы  $U_{kk} = 1$ . Воспользуемся формулой произведения матриц.  $A=U*L = A_{ij} = \left( \sum_{k=1}^n U_{ik}L_{kj} \right)$  - здесь строка U умножается на столбец L. 1-я строка U и 1-й столбец L содержат только по одному элементу  $\Rightarrow A_{11}=L_{11}$ , далее можно найти элементы 2-й строки U и 2 столбца L  $\Rightarrow$  3-я строка и 3-й столбец и т.д.. Таким образом, вычисление треугольных матриц максимально оптимизировано. Кроме того, элементы этих матриц можно последовательно располагать на месте элементов матрицы A. Когда размер оперативной памяти – критичный параметр, такой алгоритм будет очень полезным.

### Сравнительная характеристика конечных методов решения СЛАУ

- При решении сравнительно небольших СЛАУ метод Гаусса выгоднее других методов компромиссом между простотой и скоростью. Метод Жордана проще чем метод Гаусса, но требует гораздо больше вычислений (так как использует двойной прямой ход). Метод Халецкого быстрее метода Гаусса, но при этом и гораздо сложнее для реализации.
- Сравнение вариантов метода Гаусса с выбором главного элемента и без него показывает, что выбор главного элемента всегда лучше.
- При увеличении размера СЛАУ метод Гаусса постепенно теряет свое преимущество перед методом Халецкого. Этот метод разработан специально для ЭВМ и имеет оптимальные варианты расчета.
- Метод Жордана особенно удобен при решении задач, когда есть много СЛАУ у которых одна матрица (например при вычислении обратной матрицы). В такой задаче он существенно лучше метода Гаусса. Правда в этой задаче ему конкурентом является метод Халецкого, где так же можно

оптимизировать подобные вычисления.

### Итерационные методы решения СЛАУ

В алгебре для векторов и матриц вводят численную характеристику – норму. В некоторых случаях она имеет аналогию с длиной. Норма должна удовлетворять условиям:

$$a) \|\lambda x\| = |\lambda| \cdot \|x\|, \quad b) \|x+y\| \leq \|x\| + \|y\|, \quad c) \|x\| = 0 \Rightarrow x=0,$$

Обычно определяют 3 разных нормы:

$$\begin{aligned} & \|x\|_1 = \sum_{i=1}^n |x_i| & \|x\|_\infty &= \max(|x_i|) \\ a) \text{ первая} & \|A\| = \max\left(\sum_{i=1}^n |a_{ij}|\right) & \text{в) бесконечная} & \|A\|_\infty = \max\left(\sum_{j=1}^n |a_{ij}|\right) \\ c) \text{ Евклидова} & \|x\|_e = \sqrt{\sum_{i=1}^n x_i^2} & \text{и} & \|A\|_e = \frac{\sqrt{\sum_j \sum_j a_{ij}^2}}{N} \end{aligned}$$

### Сходимость итерационного метода

Итерационные методы СЛАУ основаны на получении итерационных векторов последовательных приближений. Находится каждое  $K$ -е приближение через  $K-1$ :  $x^{(k)} = F(x^{(k-1)})$

Начиная преобразования с начального вектора  $(x^{(0)} \rightarrow x^{(1)} \rightarrow x^{(2)} \rightarrow \dots)$ , получим последовательные приближения векторов. Говорят, что эта последовательность векторов сходится к вектору  $x$  если  $\lim_{k \rightarrow \infty} x^{(k)} = x, \lim_{k \rightarrow \infty} \|x - x^{(k)}\| = 0$  Так как,

оценить сходимость такой последовательности сложно, то пользуются косвенной оценкой вида  $\|X^{(k)} - X^{(k-1)}\| \leq d$

### Метод простой итерации

Пусть задана система уравнений  $AX = B$ , которая записывается в виде

$$\sum_{j=1}^n a_{ij} x_j = b_i.$$

Выразим из этой системы в каждом уравнении по одной неизвестной

$$x_i + \sum a_{ij} / a_{ii} x_j = b_i / a_{ii} \text{ (при } i \neq j) \Rightarrow x_i = (b_i - \sum a_{ij} x_j) / a_{ii} = b_i' - \sum a_{ij}' x_j,$$

где  $b_i' = b_i / a_{ii}$ ,  $a_{ij}' = \{ a_{ij} / a_{ii}, i \neq j \text{ и } 0 \text{ при } i=j \}$ . Сумма считается по  $j$  от 1 до  $n$ .

Теперь представим эту систему в итерационном виде выразив  $k$  приближение через  $k-1$  :

$$X = B' - A'X \Rightarrow X^{(k)} = B' - A'X^{(k-1)} \Rightarrow x_i = b_i' - \sum a_{ij}' x_j$$

В качестве нулевого приближения выбираем  $x^0 = B'$ . Сходимость данного метода доказывается теоремой (следствие теоремы о сжимающих отображениях).

**Теорема:** Если матрица  $A$  не вырождена и норма преобразованной матрицы  $A'$ ,  $\|A'_{ij}\| < 1$  то метод простой итерации будет сходящимся.

**Замечание:** Таким образом, сходимость метода простой итерации не зависит от выбора начального приближения  $x^0$ , а зависит только от матричной системы. Условием сходимости будет следующая формула  $\|A'_{ij}\| < 1$ . В качестве

нормы может быть выбрана любая из трех норм. По норме а) норма матрицы равна сумме модулей элементов столбца, который имеет max сумму. По норме в) норма = сумме модулей элементов строки, среди которой выбирается max.

Таким образом, либо 
$$\begin{cases} \max_j \sum_j |a_{ij}'| < 1 \\ \max_j \sum_j |a_{ij}| < 1 \end{cases}$$
 Рассмотрим формулу для  $a_{ij}$ ,

$\max_j \sum_{j \neq i} \frac{|a_{ij}|}{|a_{ii}|} < 1$  Для всех строк  $i$ :  $\sum_{j \neq i} |a_{ij}| < |a_{ii}|$  – условие диагонального

преобладания. Модуль элемента главной диагонали больше суммы модулей остальных элементов строки. Такое условие должно быть справедливо для всех

строк. Из второй нормы получаем другое условие  $\sum_{i \neq j} |a_{ij}| < |a_{jj}|$ . В данном

случае диагональный элемент должен быть больше по модулю суммы модулей остальных элементов столбца. Это свойство справедливо для всех столбцов.

Алгоритм метода простой итерации

1.Выбираем  $x^0 = B'$  и преобразуем матричную систему к итерационному виду  $X^{(k)} = B' - A' X^{(k-1)}$ .

2.Используя итерационную формулу находим следующее приближение.

3.Сравним по формуле  $\|X^{(k)} - X^{(k-1)}\| < d$ . норму с заданной погрешностью  $d$ . Если условие выполняется, то прекращаем вычисление, в противном случае, возвращаемся к п 2.

### Метод Зейделя

Зейдель предложил усовершенствовать итерационную формулу, путем использования уже полученных элементов вектора  $X$ . Тогда формула выглядит следующим образом (используем найденные значения  $x_2^{(k)}$   $k$ -го приближения):

$$x_i^{(k)} = b' - (a_{ii}' x_i^{(k-1)} + a_{ii+1}' x_{i+1}^{(k-1)} + \dots + a_{in}' x_n^{(k-1)}) - (a_{i1}' x_1^{(k)} + a_{i2}' x_2^{(k)} + \dots + a_{ii-1}' x_{i-1}^{(k)})$$

Это формула Зейделя, она точнее формулы простой итерации и сходится быстрее. У Зейделя каждое приближение сходится быстрее, чем в методе простой итерации. Поэтому весь метод сходится гораздо быстрее. Сходимость метода Зейделя можно определять по формулам метода простой итерации (так как этот метод просто улучшение метода простой итерации => если сходится простой метод, то должен сходиться и улучшенный метод).

### Метод координатной релаксации

Суть метода – последовательное уменьшение (релаксации) вектора невязки.

Пусть задана система уравнений  $A * X = B$ , если у нас есть некое приближение к точному решению  $X^{(0)}$ , то после подстановки этого приближения в систему уравнений равенства не будет:  $A * X^{(0)} \neq B$  => можно определить вектор разности правой и левой части  $N = B - A * X^{(0)}$  – это вектор невязки. Чем точнее приближение, тем меньше вектор невязки (в общем случае норма вектора невязки стремится к нулю при приближении к точному решению). Таким образом, если есть метод, который будет последовательно уменьшать вектор невязки, то этот метод является методом решения СЛАУ. Такой метод назвали

методом релаксации. Построим алгоритм подобного метода путем изменения на 1 шаге только 1 координаты, когда это изменение приводит к обнулению максимального по модулю элемента вектора невязки (координатная релаксация). Примерный алгоритм:

1. Задаем систему уравнений (все исходные матрицы  $A$  и  $B$ ).
2. Выбираем начальное приближение  $X^{(0)}=B$  аналогично методу простой итерации (в методе релаксации сходимость тоже не зависит от выбора начального приближения).
3. Вычислим вектор невязки по формуле  $N_i^{(k)} = b_i - \sum_{j=1}^n A_{ij}x_j^{(k-1)}$
4. Найдем максимальный по модулю элемент вектора невязки  $N_k = \max_i(N_i)$  здесь  $k$  – номер максимального элемента.
5. Изменим  $x_k$  на такую величину  $\Delta x_k$ , чтобы  $N_k=0$ .  $\Rightarrow$   

$$N_i^{(t)} = b_i - \sum_{j=1}^n A_{ij}x_j^{(t-1)} - A_{ik} * \Delta x_k = 0 \Rightarrow N_k^{(t)} = N_k^{(t-1)} - A_{kk} * \Delta x_k = 0 \Rightarrow \Delta x_k = N_k^{(t-1)} / A_{kk}.$$
6. Теперь вычислим следующее приближение  $X_p^{(t)} = X_p^{(t-1)}$  при  $X_k^{(t)} = X_k^{(t-1)} + \Delta x_k$  и  $p \neq k$ , тогда следующее приближение вектора невязки  $N_p^{(t)} = N_p^{(t-1)} - A_{pk} * \Delta x_k$ . Эти формулы являются итерационными формулами для данного метода.
7. Повторяем п.4-6 до тех пор пока норма вектора невязки не станет меньше заданной погрешности  $\|N\| < \varepsilon$  - это условие прерывания итераций в данном итерационном методе.

### Сравнительная характеристика итерационных методов решения СЛАУ

- Наиболее быстрый метод Зейделя, но он и самый сложный по формуле вычисления. Однако для большинства задач это усложнение вычисления не очень важно.
- Метод релаксации наименее сходящийся и количество итераций в нем всегда гораздо больше чем в других методах, но зато и операций на 1 итерацию здесь в  $N$  (число уравнений) раз меньше. Поэтому эффективность метода возрастает при увеличении числа уравнений в СЛАУ.
- Конечные методы предпочтительнее итерационных всегда, кроме 2 случаев – когда система уравнений чрезвычайно велика (слишком большие массивы требуются) и когда в матрице системы много нулей (разреженные матрицы).

## Лекция №4

### Раздел №2. Численные методы математического анализа

#### Тема 2.1 Интерполяция и Аппроксимация

Содержание: Интерполяция и экстраполяция. Интерполяционные многочлены. Конечноразностные интерполяционные формулы. Полиномы Лагранжа и

Ньютона. Погрешность интерполяции. Интерполяционные сплайны и тригонометрическая интерполяция. Дискретное и быстрое преобразование Фурье. Сравнительная характеристика методов.

### Интерполяция и Аппроксимация

Аппроксимация – замена одной функции на другую, удовлетворяющую условию близости. Условие близости – строгое математическое ограничение, которое позволяет в идеале выбрать единственную функцию. Функцию, которую заменяют при аппроксимации принято называть аппроксимируемой. Функцию, которой заменяют аппроксимируемую принято называть аппроксимирующей. Смысл аппроксимации в том, чтобы более сложную или неявную аппроксимируемую функцию заменить более простой аппроксимирующей. Поэтому аппроксимирующие функции, как правило, элементарные – полиномы, тригонометрические функции и т.д.

Интерполяция – вид аппроксимации, для которого условие близости – совпадение значений функций в отдельных точках (узлах интерполяции). Пусть задан отрезок  $[a, b]$  на котором строится интерполирующая функция  $L(x)$  для интерполируемой  $F(x)$ . Этот отрезок разбиваем на  $n$  частей точками  $x_i = a + i \cdot h$  – узлами интерполяции, где  $h = (b - a) / n$ . Это вариант равноотстоящих узлов. В общем случае узлы расположены произвольно, но при этом одна из точек совпадает с  $a$ , а другая с  $b$ . Пронумеруем точки так, чтобы они распределились по возрастанию:  $a = x_0 < x_1 < x_2 < \dots < x_n = b$ . При этом необходимо, чтобы точки не совпадали между собой. В узлах интерполяции зададим значения интерполируемой функции  $y_i = F(x_i)$ . Значения индекса меняются  $i = 0, 1, 2, \dots, n - 1$ . Таким образом, задаются 2 набора значений  $x_i, y_i$ , которые должны быть связаны условием интерполяции (условием близости)  $L(x_i) = y_i$ . Практически это совпадение функций  $L(x)$  и  $F(x)$  в узлах интерполяции. Этот вариант осуществляется, например, при построении графика функции по точкам. Принято приближение функции  $F(x)$  интерполирующей  $L(x)$  на отрезке  $[a, b]$  называть собственно интерполяцией, а вне данного отрезка экстраполяцией.

### Интерполяционные многочлены

Существуют различные варианты интерполяции (полиномиальная, тригонометрическая, кусочно-сплайновая и др.), но наиболее употребительна интерполяция полиномами. Рассмотрим некоторые варианты интерполяции полиномами.

#### Канонический интерполяционный полином

Если представить полином как функцию  $L_n(x) = \sum a_k x^k$ , то условие интерполяции  $L_n(x_i) = f_i$  превращается в систему линейных уравнений для коэффициентов полинома  $a_k$ . Можно решить эту систему и найти эти коэффициенты. Такой полином называется каноническим интерполяционным полиномом.

Полиномом называют функцию вида  $P_n(x) = \sum_{k=0}^n a_k x^k$ . Здесь наивысшая степень  $n$  называется порядком полинома. Полином легко вычисляется (например, по

схеме вычисления Горнера), дифференцируется и интегрируется. Ищем интерполирующую функцию в виде полинома  $L_n(x)$ . Запишем условие близости интерполяции:

$L_n(x_i) = f_i = \sum_{k=0}^n a_k x_i^k$  - система уравнений первой степени для величин  $a_k$ , где

полином выражен формулой  $L_n(x) = \sum_{k=0}^n a_k x^k$  и  $a_k$  - неизвестные коэффициенты, которые необходимо определить, а  $x_i^k$  - степени узлов интерполяции (узлы заданы по условию).

Для  $a_k$  получаем систему линейных уравнений  $\sum_{k=0}^n c_{ik} a_k = f_i$ ,  $n$ -степень полинома,  $c_{ik} = x_i^k$ , где  $a_0, a_1, \dots, a_n$  -  $n+1$  неизвестный коэффициент.

Канонический полином строится путем решения данной системы линейных уравнений. Вычисление полинома производится в два этапа:

- 1) вычисление  $a_k$
- 2) вычисление  $L_n(x)$

Вычисления на втором этапе можно упростить, если воспользоваться схемой Горнера. Для этого нужно преобразовать сумму в итерационный вид:

$$\sum_{k=0}^n a_k x^k = a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n = (((a_n x + a_{n-1})x + a_{n-2})x + a_{n-3})x + \dots)x + a_0$$

$$S_0 = a_n; \quad S_k = S_{k-1}x + a_{n-k}; \quad L_n(x) = \sum_{k=0}^n a_k x^k$$

Таким образом полином можно вычислять только с помощью одной итерационной формулы с использованием схемы Горнера.

### Конечноразностные интерполяционные формулы

#### Разделенные разности

$f_i = f_i(x_i)$  - 0-й порядок;  $f_{ij} = (f_j - f_i)/(x_j - x_i)$  1-ый порядок;  $f_{ijk} = (f_{jk} - f_{ij})/(x_k - x_i)$  - 2-ой порядок; аналогично  $n$ -й порядок можно выразить через  $n-1$  порядок разностей. Для эффективного вычисления строится таблица разделенных разностей:

$f_0$			
$f_1$	$f_{01}$		
...	...	$f_{012}$	
$f_n$	$f_{0n}$	$f_{01n}$	$f_{0123}$

Пусть например заданы 5 точек  $x_0, x_1, x_2, x_3, x_4$ , тогда таблица разделенных разностей выглядит следующим образом:

$x_0$	$f_0$				
$x_1$	$f_1$	$f_{01}$			
$x_2$	$f_2$	$f_{02}$	$f_{012}$		
$x_3$	$f_3$	$f_{03}$	$f_{013}$	$f_{0123}$	
$x_4$	$f_4$	$f_{04}$	$f_{014}$	$f_{0124}$	$f_{01234}$

Вычисление производится последовательно по столбцам, при этом каждый

новый столбец содержит на 1 элемент меньше. Такое вычисление завершится, когда в столбце остается 1 элемент. Главное достоинство такой таблицы – возможность простой достройки еще 1-й строки. Добавим новый узел и последовательно вычислим элементы новой строки. Элементы таблицы, которые расположены на главной диагонали, используются для вычисления полинома:  $L_4(x) = f_0 + f_{01} * (x - x_0) + f_{012} * (x - x_0) * (x - x_1) + f_{0123} * (x - x_0) * (x - x_1) * (x - x_2) + f_{01234} * (x - x_0) * (x - x_1) * (x - x_2) * (x - x_3)$

### Полиномы Лагранжа и Ньютона Полином Лагранжа

Лагранж предложил строить интерполяционный полином в следующем виде:

$$L_n(x) = \sum_{k=0}^n f_k \prod_{j \neq k} \frac{(x - x_j)}{(x_k - x_j)},$$

где  $x_i$  – узлы интерполяции,  $x$  – исследуемая точка,  $f_i$  – значение интерполируемой функции в узле  $i$ . Условие интерполяции для полинома Лагранжа  $L_n(x_j) = f_j$ . Легко проверить, что полином Лагранжа удовлетворяет условию интерполяции.

Полином Лагранжа по сравнению с каноническим полиномом имеет важное преимущество – он вычисляется прямым способом, без решения системы уравнений. Это значительно облегчает вычисления. Кроме того, существует специальная вычислительная схема Эйткена, которая позволяет вычислять значения полинома Лагранжа быстрым способом (похожим на схему Горнера):

$$L_{012\dots n}(x) = [(x - x_0) * L_{12\dots n}(x) - (x - x_n) * L_{012\dots n-1}(x)] / [x_n - x_0]$$

Здесь  $L_{012\dots n}$  – полином  $n$  порядка по всем узлам,  $L_{12\dots n}$  – полином  $n-1$  порядка по узлам от 1 до  $n$ -го,  $L_{012\dots n-1}$  – полином  $n-1$  порядка по узлам от 0 до  $n-1$ . Таким образом, схема расчета ведется по увеличению степени полинома. Минимальная степень  $n=0$ , тогда полином просто равен значению функции.

**Замечание.** Следует учитывать, что интерполяционный полином  $N$ -го порядка, заданный на  $N-1$  узле интерполяции единственный. Поэтому вычисления канонического полинома, полинома Лагранжа или полинома Ньютона дают одинаковый результат. Но форма полиномов и расчет их отличается, поэтому отличается погрешность и скорость вычисления. Вычисление по схеме Эйткена не выделяют вид полинома, поэтому эта схема вычисляет некий абстрактный интерполяционный полином.

### Полином Ньютона

Ньютон предложил другую форму записи интерполяционного полинома (канонический полином и полиномы Лагранжа, Ньютона – это один и тот же полином, но в разной форме записи). Каждая форма записи определяет свой вариант эффективного вычисления полинома. Если для канонического полинома требуется решить систему уравнений, то для полинома Лагранжа используется прямой метод вычисления. Полином в форме Ньютона получает новое достоинство – возможность последовательного добавления новых узлов,

при сравнительно простом пересчете значения (добавляется элемент суммы).  
Общая форма полинома Ньютона и вычисление коэффициентов  $A_k$  :

$$L_n(x) = A_0 + A_1(x - x_0) + A_2(x - x_0) * (x - x_1) + \dots + A_n \prod_{i=0}^{n-1} (x - x_i)$$

$$L_n(x_0) = f_0 = A_0$$

$$L_n(x_1) = f_1 = A_0 + A_1(x_1 - x_0)$$

$$L_n(x_2) = f_2 = A_0 + A_1(x_2 - x_0) + A_2(x_2 - x_0) * (x_2 - x_1)$$

.....

$$L_n(x_n) = f_n = \sum_{k=0}^n A_k \prod_{i=0}^{k-1} (x_n - x_i) = \sum_{k=0}^n A_k W_{k-1}(x_n)$$

$$A_m = \sum_{k=0}^m \frac{f_k}{l_n^{(k)}(x_k)};$$

Для эффективного вычисления коэффициентов  $A_k$  построено несколько схем вычисления – таблицы разделенных и конечных разностей, схема Эйткена и т.д..

Формула полинома Ньютона не симметрична для узлов, поэтому можно выделить алгоритмы вычисления при выборе узлов по возрастанию (вычисление вперед) и по убыванию (вычисление назад). При таких вычислениях погрешность всегда возрастает к последним узлам. Поэтому вычисление вперед дает большую точность в начале отрезка интерполирования, а вычисление назад в конце отрезка. Для общего случая произвольного выбора узлов эти варианты вычисления не отличаются по формулам. Однако при выборе равноотстоящих узлов ситуация меняется.

Для равноотстоящих узлов  $x_i = a + i * h$ , где  $h$  – расстояние между узлами имеем:

$$x_i - x_j = (i-j) * h \Rightarrow h * f_{ij} = (f_i - f_j) / (i-j) \Rightarrow y_{ij} = (f_i - f_j) / (i-j) = (y_i - y_j) / (i-j)$$

Здесь  $y_{ij}$  – величина, которую называют конечной разностью. Главное отличие конечных разностей от разделенных – они не зависят от  $h$ . Расчет конечных разностей так же ведется табличным способом. Таким образом, использование конечных разностей позволяет преобразовать формулу полинома как:

$$L_4(x) = f_0 + y_{01} * (x - x_0) / h + y_{012} * (x - x_0) * (x - x_1) / h^2 + y_{0123} * (x - x_0) * (x - x_1) * (x - x_2) / h^3 + y_{01234} * (x - x_0) * (x - x_1) * (x - x_2) * (x - x_3) / h^4$$

Далее путем замены переменных  $x' = (x - a) / h \Rightarrow x - x_0 = x'$ ,  $x - x_1 = x' - 1 \Rightarrow x - x_k = x' - k$  получим:

$$L_4(x) = f_0 + y_{01} * x' + y_{012} * x' * (x' - 1) + y_{0123} * x' * (x' - 1) * (x' - 2) + y_{01234} * x' * (x' - 1) * (x' - 2) * (x' - 3)$$

Такая формула получится только для интерполирования вперед.

Интерполирование назад определяется для узлов выражением  $x_i = b - i * h$ . Это приведет к изменению знаков для некоторых конечных разностей и формулы

замены переменных  $x''=(x-b)/h$ . Итоговая формула интерполирования назад:  
 $L_4(x'')=f_0-y_{01} * x''+y_{012} * x''(x''+1)+y_{0123} * x''(x''+1)(x''+2)+y_{01234} * x''(x''+1)(x''+2)(x''+3)$

**Замечание 1.** Для вычисления интерполяционного полинома методом Эйткена строится только таблица значений уже самого полинома без предварительного вычисления коэффициентов (что часто удобно для компьютерных вычислений).

**Замечание 2.** Интерполяционный полином Ньютона имеет следующие особенности:

- Есть разница интерполирования вперед и назад.
- Для вычисления полинома более высокого порядка для полинома Ньютона достаточно добавить новый элемент суммы и новую строку в таблицу разделенных разностей. Для других интерполяционных полиномов требуется полный пересчет всей задачи.

### Погрешность интерполяции

Погрешностью интерполяции называют величину  $d(x)=|L(x) - F(x)|$ .



На графике видно, что в узлах интерполяции погрешность равна нулю. Между узлами интерполяции погрешность положительна и не равна нулю, а в областях экстраполяции быстро растет.

Используя вид полинома Лагранжа и свойства производных для полинома, можно построить оценку для погрешности интерполяционного полинома. Это не вычислительная, а неустранимая погрешность, связанная с неточностью замены одной функции на другую. Вычислительная погрешность тоже существует и должна быть добавлена к неустранимой. Правда для малых порядков полиномов она гораздо меньше и может не учитываться для полиномов Лагранжа и Ньютона. Для оценки погрешности интерполяционного полинома можно получить формулу вида (на основе теоремы матанализа):

$$d(x)=|f^{(n+1)}(\Psi)| * |\omega_n(x)| / (n+1)!,$$

где  $x$  - точка на отрезке интерполяции,  $\omega_n(x) = \sum_{k=0}^n (x - x_k)$  функция - простой

полином n-го порядка с равным 1 коэффициентом при высшей степени,  $f^{(n+1)}$  - производная n+1 порядка от интерполируемой функции,  $\Psi$  — некая точка на отрезке интерполирования [a,b]. Для оценки выберем максимальное значение

формулы  $\Rightarrow d(x) \leq \max |f^{(n+1)}(\Psi)| * |\omega_n(x)| / (n+1)!, \text{ при } \Psi \subset [a, b]$

Для полинома Ньютона так же легко оценить погрешность. Так как погрешность полинома n+1 порядка гораздо меньше погрешности n порядка, то можно приближенно считать полином n+1 порядка точным значением. В этом случае оценка погрешности  $d_n = |L_{n+1} - L_n| = |A_{n+1}| = |f_{012...n+1}|$ . Таким образом, оценка погрешности полинома Ньютона равна модулю следующей по порядку разделенной разности.

## Интерполяционные сплайны и тригонометрическая интерполяция

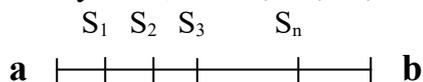
### Сплайн – интерполяция

Сплайн – интерполяция - это частный случай метода конечной интерполяции, в которой общая задача аппроксимации разбивается на подзадачи на каждом отдельном отрезке. Такой метод называется «кусочной» аппроксимацией. Приведем алгоритм данного метода интерполяции:

7. Вся область разбивается на конечное число частей.
8. На каждой части производится интерполяция.
9. В местах соединения найденные решения дополнительно сшиваются.

Обычно для сплайнов используют полиномы. Простейший вариант - линейный сплайн (ломанная). Наиболее распространенный вариант – кубический сплайн.

**Пример.** Пусть задан отрезок  $[a, b]$ , который разбит на  $n$  частей с помощью узлов,  $x_0 = a, x_1, x_2, \dots, x_n = b$ .



$n + 1$  узел и  $n$  отрезков,  $S_k$  –  $k$ -й полином сплайна.

Построим сплайн, состоящий из кусочков. Пусть сплайн кубический, его вид:

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

Всего у сплайна  $n$  кусочков по 4 коэффициента на каждом, поэтому всего имеем  $4n$  неизвестных. Построим условия интерполяции и сшивания:

$$S_i(x_i) = f_i \text{ – по условию интерполяции в узле } x_i. \text{ – } n \text{ условий.}$$

$$S_i(x_{i-1}) = f_{i-1} \text{ – по условию интерполяции в узле } x_{i-1}. \text{ – } n \text{ условий.}$$

Условия сшивания требуют совпадения в общих точках производных:

$$S'_i(x_i) = S'_{i+1}(x_i), \quad S''_i(x_i) = S''_{i+1}(x_i), (i = 1, \dots, n-1), \text{ всего } 2n-2 \text{ условий.}$$

Таким образом, получаем в сумме  $4n-2$  условия. Два оставшихся условия можно задать на концах отрезка. На концах отрезка можно определить первую и вторую производную, следовательно, существует много вариантов различных сплайнов. Наиболее известны 2 варианта:

$$1) S'_1(x_0) = S'_n(x_n) = 0 \text{ - закреплённый сплайн;}$$

$$2) S''_1(x_0) = S''_n(x_n) = 0 \text{ - свободный сплайн.}$$

Для определения коэффициентов сплайнов необходимо решить систему из  $4n$  линейных уравнений. Однако многие уравнения такой системы разрешаются однозначно, например  $a_i = f_{i-1}$ .

### Дискретное преобразование Фурье

Еще одним широко распространенным видом интерполяции является тригонометрическая интерполяция. Интерполяция тригонометрическими

функциями называется разложением в ряд Фурье. Из математического анализа известно, что любая непрерывная функция представляется в виде интеграла Фурье, где под знаком интеграла находится функция, которую называют Фурье-образом исходной функции. Для периодической функции интеграл Фурье превращается в бесконечный ряд Фурье. Пусть  $F(x)$  – периодическая функция с периодом  $T$ . Тогда она на отрезке  $[0, T]$  представляется рядом Фурье:

$$F(x) = \sum_{k=0}^{n-1} \left[ A_k \sin\left(\frac{kx}{T}\right) + B_k \cos\left(\frac{kx}{T}\right) \right] \Rightarrow \text{используя формулу Муавра } e^{ix} = \cos(x)$$

$+i*\sin(x)$  это разложение представляют в комплексном виде:  $F(x) = \sum_{k=0}^{n-1} C_k e^{ikx/T}$ .

Сделаем замену переменных  $y=x/T \Rightarrow F(y) = \sum_{k=0}^{n-1} C_k e^{iky}$ . Для

тригонометрической интерполяции необходимо найти неизвестные коэффициенты  $C_k$ . Для этого используем условие интерполяции:

$$L(y_j) = F(y_j) = f_j = \sum_{k=0}^{n-1} C_k e^{iky_j}. \text{ Если подставить сюда известные значения } f_j \text{ и } y_j$$

то решив систему нелинейных уравнений можно найти неизвестные коэффициенты  $C_k$ .

### Дискретное и быстрое преобразование Фурье

На самом деле все проще. Доказано, что неизвестные коэффициенты находятся из следующего выражения (которое называется обратным преобразованием Фурье):

$$C_k = \sum_{j=0}^{n-1} f_j e^{-iky_j}. \text{ Совокупность формул прямого и обратного преобразования}$$

Фурье для дискретного набора узлов принято называть формулами дискретного преобразования Фурье (ДПФ). Массив значений  $C$  называют Фурье-образом массива значений функции  $F$ . Простая связь между значениями массива функции и массива Фурье-образа определяет удобство использования ДПФ в качестве тригонометрической интерполяции.

На практическое использование ДПФ решающую роль оказало наличие алгоритма быстрого вычисления элементов Фурье-образа. Этот алгоритм получил название быстрого преобразования Фурье (БПФ). Не рассматривая тонкости этого сложного метода, можно выделить наиболее важные факторы, определяющие скорость вычисления для БПФ:

1. В формулах обратного преобразования ДПФ фактически используются повторяющиеся элементы тригонометрических функций. Если учесть периодичность этих функций на отрезке  $[0, T]$ , то происходит экономия вычислительных операций.

2. Для ДПФ можно доказать тождество :  $\sum_{j=0}^{n-1} f_j e^{-iky_j} = \left( \sum_{j=0}^{m-1} f_j e^{-iky_j} \right) \left( \sum_{j=0}^{p-1} f_j e^{-iky_j} \right)$

при условии, что  $n=m*p$ . Таким образом, если число узлов интерполяции

есть произведение двух целых чисел, то вычисления упрощаются, так как для каждой из скобок нужно вычислять меньше тригонометрических функций.

3. В пределе, второе свойство приводит к переходу к числу узлов равному степени числа 2 (например - 8,16,32,64,128). Это дает наиболее оптимальную скорость вычисления. Делим все узлы пополам и по 2 свойству обе суммы будут иметь одинаковые тригонометрические функции. Далее каждая сумма вновь делится пополам и так до набора сумм по 2 узлам.

### **Сравнительная характеристика методов интерполяции**

Методы интерполяции необходимы для решения очень многих задач вычислительной математики и имеют особые варианты использования (например для численного дифференцирования и интегрирования). Это и определило появление такого большого числа вариантов этих методов. Наименее популярным является канонический полином, так как он требует решения СЛАУ, а этот вариант наиболее трудоемкий для объема вычислений. Поэтому выделим положительные стороны различных вариантов интерполяции при сравнении с каноническим вариантом:

1. Полиномы Лагранжа и Ньютона имеют явный вид представления не требующий решения уравнений для расчета полиномов.
2. Полиномы Лагранжа и Ньютона имеют удобные для оценок формулы погрешностей.
3. Сравнивая полиномы Лагранжа и Ньютона можно выделять ряд преимуществ полинома Ньютона (оценка погрешности, переход к полиному более высокого порядка) и недостатки (необходимость отдельного расчета разделенных разностей). Но в целом полином Ньютона более выгоден для применения.
4. Сплайны и тригонометрическая интерполяция (ДПФ) имеют свои области использования. Прежде всего это случай, когда есть много узлов/точек интерполяции. Для обычной интерполяции тогда нужен большой порядок полинома, но это связано не оптимальным использованием вычислительных ресурсов компьютера и проблемами с вычислительной погрешностью. В одних случаях тогда выгоднее кусочная интерполяция сплайнами (компьютерная графика), в других случаях выгоднее ДПФ (сигналы с ограниченным спектром).

## **Лекция №5**

### **Раздел №2. Численные методы матанализа**

#### **Тема 2.1 Интерполяция и Аппроксимация**

#### **Тема 2.2 Численное дифференцирование и интегрирование**

- 1. Полином наилучшего равномерного приближения**
- 2. Методы аппроксимации**
- 3. Сравнительная характеристика аппроксимационных методов**
- 4. Метод обратного интерполирования**
- 5. Задача численного дифференцирования и интегрирования**

Содержание: Многочлены Чебышева и наилучшие равномерные приближения (НРП). Свойства многочленов Чебышева. Построение полинома НРП. Методы аппроксимации. Метод наименьших квадратов (МНК). Выбор базиса. Алгоритм метода. Использование МНК. Метод разложения в ряд. Сравнительная характеристика методов. Метод обратного интерполирования.

Проблема численного дифференцирования и интегрирования зависимостей. Численные формулы дифференцирования. Остаточные члены простейших формул и их оценка.

## 1. Полином наилучшего равномерного приближения

**Определение.** Полином наилучшего равномерного приближения (НРП), построенным по  $n+1$  узлу на отрезке  $[a;b]$  – это такой интерполяционный полином, абсолютная максимальная погрешность (отклонение от точного значения) которого является минимальной для всех интерполяционных полиномов данного порядка на данном отрезке. Для построения полиномов НРП используются полиномы Чебышева.

Полиномы Чебышева

Некоторые факты о полиномах Чебышева:

Общий вид полинома Чебышева

Полиномы Чебышева, рассматриваются на отрезке  $[-1;1]$  и записываются в двух формах:

**Первая форма:** Начальные значения  $T_0(x)=1$ ;  $T_1(x)=x$ .

Итерационная формула для вычисления полиномов  $T_{n+1}(x)=2xT_n(x)-T_{n-1}(x)$ .

$T_0=1$ ;  $\Rightarrow T_1=x$ ;  $\Rightarrow T_2=2x^2-1$ ;  $\Rightarrow T_3=4x^3-3x$   $\Rightarrow$  и т. д.

коэффициент при высшей степени  $T_n \Rightarrow a_n=2^{n-1}$ .

Присоединенные полиномы Чебышева имеют коэффициент 1 при высшей

степени:  $\tilde{T}_n(x) = \frac{T_n(x)}{2^{n-1}} \Rightarrow \tilde{a}_n = 1$ .

**Вторая форма:**  $T_n(x)=\cos(n*\arccos(x))$  – тригонометрическая форма.

**Корни полинома Чебышева:**  $x_k = \cos(\pi(1/2+k)/n)$ ,  $k=0,1,2,\dots,n$

**Свойства многочленов Чебышева**

**Теорема.** Полиномы Чебышева являются полиномами наименьшего отклонения от нуля, т.е. выполнено следующее условие для любого полинома  $P_n(x)$ , заданного на  $[-1;1]$  :

$\sup_{[-1;1]} |P_n(x)| \geq \sup_{[-1;1]} |\tilde{T}_n(x)|$ , при условии, что  $P_n(x)$  имеет коэффициент  $a_n=1$ .

**Следствие1.** На отрезке  $[-1;1]$  полином Чебышева имеет самое маленькое отклонение от нуля (наименьшее уклонение от нуля).

**Следствие2.** Если задан отрезок  $[a;b]$ , то можно сделать замену переменной  $x'=((b+a)+(b-a)*x)/2$  которая переводит отрезок  $[-1;1] \rightarrow [a;b]$ .

Тогда, корни так же будут  $x'_k'=((b+a)+(b-a)*\cos(\pi(1/2+k)/n))/2$ ,  $k=0,1,2,\dots,n$ , здесь  $x'_k'$  - преобразованные корни на новом отрезке.

**Построение полинома НРП**

Пусть задан отрезок  $[a;b]$  на котором необходимо построить полином НРП. Если выбрать преобразованные корни полинома Чебышева в качестве узлов интерполяции, то погрешность интерполяции (в соответствии с формулой оценки погрешности полинома Лагранжа) будет полиномом Чебышева (так как погрешность в узлах равна 0). Построим по найденным узлам интерполяционный полином любым способом (по формулам Лагранжа или Ньютона). Построенный по этим узлам интерполяционный полином будет удовлетворять определению полинома наилучшего равномерного приближения => будет являться полиномом наилучшего равномерного приближения.

## 2. Методы аппроксимации

### Методы аппроксимации

Кроме интерполяционных методов существуют и другие варианты аппроксимации. Рассмотрим некоторые из них – метод разложения функции в ряд, метод наименьших квадратов (МНК).

#### Разложение функции в степенной ряд

Одним из распространенных в вычислительной математике не интерполяционным методом приближения функции служит ее разложение в ряд Тейлора. Как известно, любая непрерывная и бесконечно раз дифференцируемая функция может быть разложена в бесконечный ряд Тейлора:

$$f(x) = f(x_0) + f'(x_0) \cdot (x-x_0)/1! + f''(x_0) \cdot (x-x_0)^2/2! + f'''(x_0) \cdot (x-x_0)^3/3! + \dots$$

Здесь  $x_0$  – некая фиксированная точка, как правило, это ноль. Для многих функций производные в нуле легко вычисляются явно, поэтому ряд имеет вполне обычный для полинома вид. Например для функции

$$\sin(x) = \sum_{k=0}^{\infty} \frac{(-1)^k x^{2k+1}}{(2k+1)!} .$$

Этот ряд является основой для разработки

эффективных методов вычисления стандартной функции  $\sin(x)$ , которой мы пользуемся при вычислениях на ЭВМ. Данный ряд быстро сходится, особенно для небольших значений аргумента  $x$ . Свойство периодичности позволяет вычисления на отрезке  $[0, \pi/2]$  распространить на все значения  $x$ .

#### Метод наименьших квадратов

Наиболее важный для практического применения метод аппроксимации – метод наименьших квадратов (МНК). В методе МНК принято аппроксимационную

функцию  $L$  искать в виде суммы  $L(x) = \sum_{i=0}^m C_i \cdot \varphi_i(x)$ , где  $C_i$  – const,  $\varphi_i$  – выбранные

функции (базисные функции),  $\varphi_0(x), \varphi_1(x), \varphi_2(x), \dots, \varphi_m(x)$  – базис,  $m < n$ .

Коэффициенты  $C_i$  находятся как решение системы линейных уравнений:

$$\begin{cases} \sum_{j=0}^m C_j (\varphi_0, \varphi_j) = (f, \varphi_0) \\ \dots\dots\dots \\ \sum_{j=0}^m C_j (\varphi_m, \varphi_j) = (f, \varphi_{m+1}) \end{cases} - \text{формулы, которые определяют матрицу системы}$$

уравнений. Здесь используются  $(f, \varphi_m)$  — скалярные произведения функций. Система имеет единственное решение, если определитель матрицы системы не равен нулю (матрица не вырождена). Формула расчета скалярного произведения

$$(\phi_j, \phi_i) = \sum_{k=0}^n \phi_j(x_k) \cdot \phi_i(x_k).$$

Если система уравнений имеет вид  $\sum_{j=0}^m C_j \cdot A_{ij} = b_i$ . Тогда матрица системы вычисляется как скалярное произведение  $A_{ij} = (\varphi_i, \varphi_j)$ , вектор столбец правой части как  $b_i = (f, \varphi_i)$

Матрица вида  $\begin{bmatrix} (\varphi_0, \varphi_0), (\varphi_0, \varphi_1), \dots, (\varphi_0, \varphi_n) \\ (\varphi_1, \varphi_0), (\varphi_1, \varphi_1), \dots, (\varphi_1, \varphi_n) \\ \dots \\ (\varphi_n, \varphi_0), (\varphi_n, \varphi_1), \dots, (\varphi_n, \varphi_n) \end{bmatrix}$  называется матрицей Грамма.

Матрица Грамма для степенного базиса:  $\varphi_0=1, \varphi_1=x, \varphi_2=x^2, \dots, \varphi_m=x^m$

$$\begin{pmatrix} n+1 & \sum x_i & \sum x_i^2 & \dots & \sum x_i^m \\ \sum x_i & \sum x_i^2 & \sum x_i^3 & \dots & \sum x_i^{m+1} \\ \dots & \dots & \dots & \dots & \dots \\ \sum x_i^m & \sum x_i^{n+1} & \dots & \dots & \sum x_i^{m+2} \end{pmatrix}$$

Окончательно получим СЛАУ для степенного базиса МНК:

$$\begin{pmatrix} n+1 & \sum x_i & \sum x_i^2 & \dots & \sum x_i^m \\ \sum x_i & \sum x_i^2 & \sum x_i^3 & \dots & \sum x_i^{m+1} \\ \dots & \dots & \dots & \dots & \dots \\ \sum x_i^m & \sum x_i^{n+1} & \dots & \dots & \sum x_i^{m+2} \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ \dots \\ C_m \end{pmatrix} = \begin{pmatrix} \sum f_i \\ \sum f_i(x_i) \\ \dots \\ \sum f_i(x_i)^n \end{pmatrix}.$$

Алгоритм метода наименьших квадратов:

1. Выбираем  $m \ll n$ . Определяем заданные погрешности  $d(f_i)$  в узлах  $x_i$ .
2. Строим матрицу Грамма и находим  $C_k$  (решая СЛАУ).  
 $\sum (L(x_i) - f(x_i))^2 = \sum (\sum C_k \varphi_k(x_i) - f(x_i))^2 = \Phi$
3. Вычисляем  $\Phi$
4. Если  $\Phi \geq \sum (L(x_i) - f(x_i))^2$  то  $m$  можно увеличить,  $m = m + 1$ . Теперь возвращаемся в п.2. Если  $\Phi \approx \sum (L(x_i) - f(x_i))^2$  то завершаем вычисления (увеличивать  $m$  нет смысла). Таким образом, полученное значение  $m$  будет оптимальным выбором для данной задачи.

**Замечание.** МНК может использоваться с различными базисами в зависимости от решаемой задачи. Среди таких базисов особо выделяют ортогональные. Для

того чтобы базис был ортогональным необходимо потребовать выполнения следующего ограничения:  $(\phi_j, \phi_i) = \begin{cases} = 0, i \neq j \\ \neq 0, i = j \end{cases}$ . В этом варианте базиса матрица

Грамма имеет ненулевые элементы только на главной диагонали и решать СЛАУ нет необходимости. Это значительно упрощает построение аппроксимирующей функции.

### 3. Сравнительная характеристика аппроксимационных методов

Методы аппроксимации имеют особые варианты использования. Это прежде всего случай, когда в заданных точках (узлах) значения аппроксимируемых функций известны с погрешностью. Тогда нет смысла строить интерполяцию с точным совпадением в узлах. При этом как правило число узлов может быть взято значительно больше чем нужно для интерполяции. Здесь удобны методы, основанные на принципе МНК (наименьших квадратов). Эти методы лежат в основе алгоритмов регрессионной обработки информации, которая часто используется для обработки экспериментальных данных (в том числе и в педагогике).

Другой подход реализован в методе разложения в ряд, когда используются известные значения производных в некой точке. Это дает возможность выразить многие функции в виде бесконечных степенных рядов. Это базовый метод для расчета многих стандартных функций ( $\sin(x)$ ,  $\cos(x)$ ,  $\operatorname{tg}(x)$ ,  $\operatorname{ctg}(x)$ ,  $e^x$ ,  $\ln(x)$  и т.д.).

### 4. Метод обратного интерполирования

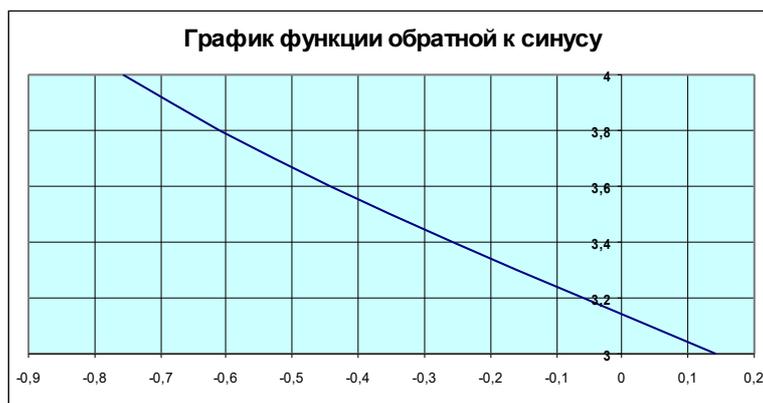
Интерполирование может быть использовано для численного решения нелинейного уравнения. Пусть задано нелинейное уравнение вида  $F(x)=0$  и на отрезке  $[a,b]$  отделен корень данного уравнения. Рассмотрим функцию  $X(F)$  – обратную к  $F(x)$ . Ее характерной особенностью будет тот факт, что  $X(0) = x_k$ , где  $x_k$  – искомый корень. Таким образом, если бы нам был известен вид обратной функции или алгоритм расчета, то можно было бы легко вычислить корень. С другой стороны интерполяция является методом получения алгоритма расчета функции заданной в виде таблицы значений. Такую таблицу для обратной функции легко получить.

Рассмотрим пример. Пусть задано уравнение  $\sin(x)=0$ . На отрезке  $[3,4]$  оно имеет корень в точке  $\pi$ . Зададим таблицу значений синуса на данном отрезке:

$\sin(x)$	0,141	-0,058	-0,26	-0,4	-0,61	-0,8
$x$	3	3,2	3,4	3,6	3,8	4

Если рассматривать нижнюю строку таблицы как аргумент, а верхнюю строку как функцию, то это таблица функции синус. Если же рассматривать нижнюю строку таблицы как функцию, а верхнюю строку как аргумент, то это таблица функции обратной к синусу.

График этой обратной функции будет выглядеть так:



На графике видно, что корень (точка пересечения оси Y) расположен вблизи точки 3.2. Так же как построение графика, возможно и построение полинома Лагранжа для обратной функции.

В простейшем случае полинома 1-го порядка имеем:

- Узлы – 2 точки. Выбираем крайние точки  $x_0=0,141$ ,  $x_1=-0,8$ .
- Значения обратной функции в этих точках  $Y_0=3$ ,  $Y_1=4$ .
- Полином Лагранжа  $L(x) = Y_0*(x - x_1)/(x_0 - x_1) + Y_1*(x - x_0)/(x_1 - x_0)$ .
- Определим корень как значение полинома в точке ноль  $x_k = L(0) = Y_0*(-x_1)/(x_0 - x_1) + Y_1*(-x_0)/(x_1 - x_0) = 3*0,8/(0,141+0,8)+4*(-0,141)/(-0,141-0,8) = 3,1498$ .

Получено значение близкое к числу  $\pi = 3,141593$ , погрешность 0.008.

Приведенный пример показывает, что метод обратного интерполирования довольно точен даже для невысоких степеней интерполяционного полинома. Так как в этом методе трудно заранее оценить погрешность, то рекомендуется использование полинома Ньютона. Вычисление полинома Ньютона может быть организовано последовательно с увеличением порядка. Тогда этот процесс будет аналогичен итерационному, с использованием косвенной оценкой погрешности.

На практике метод обратного интерполирования удобен для ручных расчетов. Для ЭВМ вычисление интерполяционного полинома достаточно сложная операция по сравнению, например, с методом дихотомии. С другой стороны такие методы бывают выгодны в задачах, где функция  $F(x)$  вычисляется очень сложно и долго. Тогда главные затраты времени вычисления – вычисление функции. Чем реже будет вычисляться функция, тем быстрее будет алгоритм. В таком случае метод обратного интерполирования становится конкурентным.

### 5. Задача численного дифференцирования и интегрирования

Численное дифференцирование и интегрирование – вычисление производных и интегралов по приближенным разностным формулам. Если функция задана в виде таблицы или имеет очень сложный вид и алгоритм расчета, то ее можно дифференцировать или интегрировать только численным способом. Для численного дифференцирования и интегрирования существуют ряд формул численного приближенного вычисления. Для получения таких формул можно использовать два варианта построения или вывода:

1. Использование разложения в ряд Тейлора.
2. Дифференцирование интерполяционных полиномов или аппроксимационных функций.

Рассмотрим 2 формулы разложения в ряд Тейлора:

$$F(x_i+h) = F(x_i) + F'(x_i)*h + F''(x_i)*h^2/2 + F'''(x_i)*h^3/6 + F^{(4)}(x_i)*h^3/24 + \dots$$

$$F(x_i-h) = F(x_i) - F'(x_i)*h + F''(x_i)*h^2/2 - F'''(x_i)*h^3/6 + F^{(4)}(x_i)*h^3/24 - \dots$$

Многоточием в формуле обозначается все последующие (до бесконечности) элементы ряда. Существуют специальные оценки этого остатка. В частности в матанализе доказывается, что остаток формулы Тейлора всегда можно представить как  $O(h^n)$ . Это значит, что в формуле оставлена максимальная степень  $n-1$ .

**Замечание.** Выражение  $O(h^n)$  является асимптотической оценкой роста или уменьшения некой величины, которая зависит от  $h$ . Практически можно считать, что это произведение неизвестной константы и  $h^n \Rightarrow O(h^n) = \text{const} * h^n$ . Смысл такого выражения в том, что при стремлении  $h$  к бесконечности величина будет изменяться со скоростью пропорциональной  $h^n$ . Таким образом, асимптотические оценки позволяют сравнивать скорости роста или уменьшения различных формул.

Теперь наши формулы будут иметь вид:

$$F(x_i+h) = F(x_i) + F'(x_i)*h + F''(x_i)*h^2/2 + F'''(x_i)*h^3/6 + F^{(4)}(x_i)*h^3/24 + O(h^4) \quad (A)$$

$$F(x_i-h) = F(x_i) - F'(x_i)*h + F''(x_i)*h^2/2 - F'''(x_i)*h^3/6 + F^{(4)}(x_i)*h^3/24 + O(h^4) \quad (B)$$

Удобство такого представления заключается так же в том, что  $O(h^n)$  является так же оценкой погрешности для формулы. Говорят, что формула имеет порядок погрешности  $n$ , если последний элемент в формуле  $O(h^n)$ . В нашем случае мы ограничились 4 порядком погрешности. Фактически порядок оценки погрешности определяет точность формулы, поэтому порядок погрешности можно называть порядком точности формулы.

Теперь, используя эти разложения, построим формулы для разностных производных. Если в формуле (A) оставить только 2-ю степень в разложении и выразить из этой формулы производную, то мы получим:

$$F(x_i+h) = F(x_i) + F'(x_i)*h + O(h^2) \Rightarrow F'(x_i) = (F(x_i+h) - F(x_i))/h + O(h^1)$$

Это выражение определяет формулу правой разностной производной

$(F(x_i+h)-F(x_i))/h$ , имеющей 1-й порядок точности. Аналогично из формулы (B) получим левую разностную производную  $(F(x_i)-F(x_i-h))/h$ , имеющую так же 1-й порядок точности.

Среднее между этими производными – центральная разностная производная, имеет 2-й порядок точности и формулу  $(F(x_i+h) - F(x_i-h))/(2h)$  (это легко доказать, если вычесть из (A) (B) и разделить разность на 2).

Для получения формулы 2-й производной нужно сложить (A) и (B) и разделить на  $h^2$ . Получим формулу второй разностной производной  $(F(x_i+h)+F(x_i-h)-2*F(x_i))/h^2$ , которая имеет 2-й порядок точности.

**Замечание.** Очевидно, что использование формул с более высоким порядком точности более выгодно. Это можно пояснить простым примером. Пусть есть 3 формулы, имеющие 1,2,3 порядки точности соответственно. Уменьшим шаг  $h$  в 10 раз. Тогда погрешность 1-й формулы уменьшится в 10 раз, 2-й в 100 раз, а 3-

й в 1000 раз. Таким образом, достичь заданной погрешности легче при использовании 3-й формулы.

Получить формулы численного дифференцирования можно и другим способом. Если записать формулу для полинома Лагранжа и продифференцировать ее, то можно получить различные формулы разностных производных. При этом существует некоторое правило (которое обычно верно, но в некоторых случаях не соблюдается) – порядок погрешности при дифференцировании интерполяционного полинома уменьшается на 1. Как нам уже известно, формула оценки погрешности полинома Лагранжа,

пропорциональна полиному  $\omega_n(x) = \sum_{k=0}^n (x - x_k)$ . Если считать, что  $(x - x_k) = h$ , то

оценка погрешности интерполирования является  $O(h^{n+1})$ . Таким образом, порядок точности интерполяционного полинома на 1 больше его порядка. После дифференцирования порядок точности уменьшится на 1  $\Rightarrow$  для получения формулы аналогичной правой разностной нужно дифференцировать полином 1-го порядка, для центральной разностной полином 2-го порядка, а для 2-й разностной уже полином 3-го порядка (так как формула дифференцируется дважды).

## Лекция №6

### Раздел №2. Численные методы математического анализа

#### Тема 2.2 Численное дифференцирование и интегрирование

##### 1. Методы Рунге практической оценки погрешностей

##### 2. Задача численного интегрирования

##### 3. Формула Ньютона-Котеса. Коэффициенты Котеса и их свойства

##### 4. Формулы однократного интегрирования

##### 5. Методы многократного интегрирования

##### 6. Другие квадратурные формулы численного интегрирования

Содержание: Методы Рунге практической оценки погрешностей. Сравнительная характеристика методов. Задача численного интегрирования. Формула Ньютона-Котеса. Коэффициенты Котеса и их свойства. Однократный и многократный методы. Квадратурные формулы прямоугольников, трапеций и Симпсона. Практическая оценка погрешности. Квадратурные формулы Чебышева и Гаусса. Сравнительная характеристика методов.

##### 1. Методы Рунге практической оценки погрешностей

Для любого численного метода, который имеет асимптотическую оценку погрешности можно использовать специальные методы Рунге оценки погрешности и уточнения вычисления. Формулы Рунге можно использовать для оценки погрешности численного дифференцирования, интегрирования, решения дифференцированных уравнений, где погрешность представляется  $O(h^n) = C * h^n$ , где  $C = \text{const}$  и не зависит от  $h$ .

Формулы Рунге:

Пусть производятся вычисления с шагом  $h$  некой величины  $I_h = I \pm O(h^n) = I + C * h^n$

и вычисление той же величины с шагом  $h/2$ ,  $I_{h/2} = I \pm O((h/2)^n) = I + C * (h/2)^n$

тогда из этих формул можно выразить неизвестную константу

асимптотического приближения :  $I_h - I_{h/2} = C * h^n * (1 - 1/2^n) \Rightarrow C = (I_h - I_{h/2}) / [h^n * (1 - 1/2^n)]$

Это первая формула Рунге, которая определяет оценку погрешности по формуле

$d = Ch^n$ . Первая формула Рунге позволяет определить коэффициент для погрешности

и оценить само значение погрешности. Если формулу для  $C$  подставить в

выражение  $I_h = I + Ch^n$ , то можно выразить точное значение  $I$ . Конечно это

значение тоже будет неточное, но гораздо точнее чем  $I_h$ . Так получается 2-я

формула уточнения Рунге:  $I_{\text{точ}} = I_h + (I_h - I_{h/2}) / (1 - 1/2^n)$

вторая формула Рунге уточняет вычисленные значения.

И наконец, если мы проведем дополнительно вычисления с шагом  $h/4$ , то

можем выразить порядок погрешности метода через результаты вычисления:

$C = (I_h - I_{h/2}) / [h^n * (1 - 1/2^n)] = (I_{h/2} - I_{h/4}) * 2^n / [h^n * (1 - 1/2^n)] \Rightarrow (I_h - I_{h/2}) = (I_{h/2} - I_{h/4}) * 2^n \Rightarrow$

$(I_h - I_{h/2}) / (I_{h/2} - I_{h/4}) = 2^n \Rightarrow \ln((I_h - I_{h/2}) / (I_{h/2} - I_{h/4})) = n * \ln(2) \Rightarrow n = \ln((I_h - I_{h/2}) / (I_{h/2} - I_{h/4})) / \ln(2)$

$n = \ln((I_h - I_{h/2}) / (I_{h/2} - I_{h/4})) / \ln(2)$  - Это третья формула Рунге, которая оценивает пор-

ядок погрешности  $n$  используемого метода. Третья формула дает возможность

узнать порядок точности вычислительного метода экспериментальным путем

без математического вывода.

## 2. Задача численного интегрирования

В общем случае задача численного интегрирования сводится к нахождению

интеграла  $I = \int_a^b f(x) dx$  с помощью приближенных формул. Численные методы

при решении задачи интегрирования используются в следующих случаях:

а) Функция задаётся в виде таблицы значений или её значение вычисляется, как решение другой задачи (функция задаётся неявным видом).

б) Функция задана явно, но первообразную не возможно записать в явном виде.

Методы, используемые для решения задачи численного интегрирования можно разделить на однократные и многократные.

**Однократный метод** реализуется путем замены интегрируемой функции на ее аппроксимацию, которая имеет явную первообразную. Это дает возможность указать явную формулу для вычисления определенного интеграла.

**Многократный метод** - используем однократный метод, но весь отрезок интегрирования от  $a$  до  $b$  делим на  $n$  частей с шагом  $h = (b-a)/n$  с помощью точек  $x_i = a + ih$ .

На каждом участке  $[x_{i-1}, x_i]$  интеграл вычисляется однократным методом, а затем используя свойство аддитивности интегралов общий интеграл вычисляется как сумма интегралов на отдельных участках:

$$I = \int_a^b f(x) dx = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} f(x) dx .$$

Таким образом, основа формулы многократного метода – формула метода однократного. Смысл использования многократного метода – увеличение точности вычисления за счет уменьшения отрезка интегрирования, так как погрешность уменьшается нелинейно при уменьшении отрезка. Как правило формулы однократных методов таковы, что их погрешность быстро уменьшается при уменьшении отрезка интегрирования.

### 3. Формула Ньютона-Котеса. Коэффициенты Котеса и их свойства

Одним из основных вариантов получения формул однократного метода является метод интегрирования интерполяционных полиномов. Если использовать полином Лагранжа, то получаются формулы, которые принято называть формулами Ньютона-Котеса:

$\int_a^b L_n(x) dx = (b-a) \sum_{i=1}^n f_i C_i$  - формула Ньютона – Котеса. Здесь коэффициенты  $C$

$C_i = \frac{1}{n \prod_{i \neq j} (i-j)} \int_0^n \prod_{i \neq j} (x'-j) dx'$  - коэффициенты Котеса. Для получения этого

выражения для коэффициентов необходимо подставить в интеграл формулу полинома Лагранжа для равноотстоящих узлов. Затем делается замена переменных  $x' = n*(x-a)/(b-a) \Rightarrow$  и находим выражения, которые необходимо подставить в интеграл  $(x_i - x_j) = (b-a)*(i-j)/n$ ,  $(x - x_j) = (b-a)*(x' - j)/n$ ,  $dx' = n*dx/(b-a)$ .

#### Коэффициенты Котеса и их свойства

Главным свойством (по смыслу определения) коэффициентов Котеса является, то что они не зависят от пределов интегрирования и вида интегрируемой функции. Таким образом, их можно вычислить заранее, поместить в таблицу и использовать при любом вычислении интеграла. Это значительно упрощает использование на практике формулы Ньютона-Котеса. Кроме того, вычисление коэффициентов упрощается, если используется ряд математических свойств коэффициентов Котеса:

1)  $\sum_{i=0}^n C_i = 1$  - нормировка

Свойство нормировки легко доказать, если рассмотреть вариант интегрирования функции тождественно равной 1 на отрезке  $[0,1]$ . Тогда формула Ньютона-Котеса принимает вид свойства нормировки.

2)  $C_i = C_{n-i}$  симметричность.

Свойство симметричности доказывается долгим и сложным способом, поэтому приводим без доказательства.

3)  $C_i = \frac{1}{n} \frac{(-1)^{n-i}}{i!(n-1)!} * \int_0^n q * (q-1) * (q-2) * ... * (q-n) * dq$  - выражения для коэффициентов не

зависят от пределов интегрирования. Данное свойство заключается в приведении формулы вычисления коэффициентов к максимально удобному и простому виду.

## 4. Формулы однократного интегрирования

Частные случаи формул:

- Если порядок полинома  $n=0$ , то  $C_0=1 \Rightarrow I = (b-a)*f(a)$  - формула левого прямоугольника или  $I = (b-a)*f(b)$  - формула правого прямоугольника.
- Если  $n=1$ , то  $C_0 = C_1 = 1/2 \Rightarrow I = (b-a)*[f(a)+f(b)]/2$  - формула трапеции.
- Если  $n=2$ ,  $C_0 = C_2 = 1/6$ ,  $C_1 = 4/6 \Rightarrow I = (b-a)*[f(a)+f(b)+4*f(c)]/6$ , где точка  $c=(b+a)/2$ . это формула парабол или Симпсона.

Название этих формул имеет геометрический смысл. Как известно, интеграл равен площади фигуры, ограниченной графиком функции и осью X. Полином нулевого порядка – константа  $\Rightarrow$  его график прямая параллельная оси X. Это значит, что в методе прямоугольников мы заменяем площадь фигуры прямоугольником. Его основание – отрезок интегрирования  $[a,b]$ , а высота – значение функции  $f(x)$ . Значение функции можно выбирать на левом конце отрезка  $f(a)$  или на правом  $f(b)$ .

Для метода трапеций полином имеет 1-й порядок – это прямая, которая имеет наклон  $(f(a)+f(b))/2$ . Таким образом, мы заменяем площадь фигуры трапецией.

Для метода Симпсона полином имеет 2-й порядок – это парабола, которая должна проходить через 3 точки. Поэтому, мы вынуждены кроме точек  $a$  и  $b$  брать еще точку середины отрезка  $c=(a+b)/2$ . Таким образом, мы заменяем площадь фигуры площадью под параболой.

Порядок точности этих однократных методов может быть оценен по правилу подобному для формул численного дифференцирования. При этом нужно учитывать, что интегрирование обратное дифференцированию, поэтому порядок точности после интегрирования увеличивается на 1. Учитывая, что порядок точности для интерполяционных полиномов равен  $n+1$ , получим для методов прямоугольников  $O(h^2)$  (2-й порядок), метода трапеций  $O(h^3)$  (3-й порядок), метода Симпсона  $O(h^4)$  (4-й порядок). На самом деле, метод Симпсона имеет еще более высокий порядок точности  $O(h^5)$  (5-й порядок), что связано с его симметричной формой записи.

Характерно, что формула трапеции может быть получена как среднее (полу сумма) формул левых и правых прямоугольников. Теперь эти формулы можно использовать для получения многократных методов.

## 5. Методы многократного интегрирования

Для многократного метода нужно подставить формулы однократных методов под знак суммы. При этом получаем:

получения многократных методов.

$$I_{л\_np} = \sum_{i=0}^{n-1} f(a+i*h)*h - \text{многократный метод левых прямоугольников } h=(b-a)/n$$

$$I_{np}^n = \sum_{i=1}^n f(a+i*h)*h - \text{многократный метод правых прямоугольников.}$$

$$I_{mp} = \frac{I_m^n + I_{np}^n}{2} = \frac{f(a) + f(b)}{2} * h + \sum_{i=1}^{n-1} f(a + i * h) * h - \text{многократный метод трапеций.}$$

Более сложно выглядит формула метода парабол. Здесь необходимо учитывать, что формула Симпсона требует не четного количества узлов (четного количества отрезков  $n$ ), либо дополнительного включения промежуточных узлов. Таким образом, если число отрезков  $n$  - четно, то все узлы кроме крайних делятся на четные и нечетные:

$$I = \frac{f(a) + f(b)}{6} * h + \frac{h}{3} * \sum_{\text{нечетн}} f_i + \frac{4}{6} * h * \sum_{\text{четн}} f_i$$

Здесь суммирование ведется либо по узлам с четным, либо с нечетным номером.

**Замечание.** Следует учитывать, что для многократного метода погрешность является суммой погрешностей каждого отрезка. Поэтому, общая погрешность будет больше и даже уменьшится порядок точности. Это связано с тем, что сумма погрешностей пропорциональна числу отрезков  $n$  и обратно пропорциональна длине отрезка  $h$ . Это значит, что суммирование погрешности аналогично умножению на  $n$  или делению на  $h \Rightarrow$  уменьшится порядок точности на 1. В результате порядок точности многократного метода прямоугольников  $O(h^1)$  (1-й порядок), метода трапеций  $O(h^2)$  (2-й порядок), метода Симпсона  $O(h^4)$  (4-й порядок). Таким образом, увеличение числа отрезков в 10 раз даст уменьшение погрешности в 10, 100 и 10000 раз соответственно. Такой результат и определил практику использования формул Ньютона-Котеса. Однократные методы используют только для ручных вычислений. Для ЭВМ многократные методы существенно проще реализовать. При этом для простых задач достаточно метода трапеций, а более сложные задачи решают методом парабол. Формулы более высоких порядков применяются на практике редко, так как они гораздо сложнее, а точности метода Симпсона хватает с избытком.

## 6. Другие квадратурные формулы численного интегрирования

### Метод интегрирования разложением в ряд

Помимо интерполяционных полиномов, для получения формул численного интегрирования можно использовать другие варианты аппроксимации. В некоторых случаях оказывается очень удобным использовать аппроксимацию

разложения в ряд Тейлора. Пусть вычисляется интеграл  $I = \int_a^b f(x) dx$  и имеется

общий вид разложения подинтегральной функции в ряд Тейлора:

$$f(x) = f(x') + \frac{f'(x')}{1!} (x - x') + \frac{f''(x')}{2!} (x - x')^2 + \frac{f'''(x')}{3!} (x - x')^3 ..$$

Прямая подстановка этого ряда в интеграл математически не всегда корректна (существуют ряды, сумма интегралов элементов которых расходится или неограниченна). Однако для конечной суммы ряда такая подстановка всегда корректна. Пусть  $x' = 0$  и мы ограничиваемся выбором  $n$  элементов суммы:

$$f(x) = f(0) + \frac{f'(0)}{1!} x + \frac{f''(0)}{2!} x^2 + \frac{f'''(0)}{3!} x^3 + \dots + \frac{f^{(n)}(0)}{n!} x^n$$

тогда интегрирование приведет к следующему выражению:

$$I_n = \int_a^b f(x)dx = f(0) \int_a^b dx + \frac{f'(0)}{1!} \int_a^b x dx + \frac{f''(0)}{2!} \int_a^b x^2 dx + \frac{f'''(0)}{3!} \int_a^b x^3 dx + \dots + \frac{f^{(n)}(0)}{n!} \int_a^b x^n dx$$

Все интегралы в этой формуле легко вычисляются в явном виде. Величина  $I_n$  при этом является n-м приближением к точному значению. Можно построить итерационную формулу вычисления следующего приближения:

$$I_{n+1} = I_n + \frac{f^{(n+1)}(0)}{(n+1)!} \int_a^b x^{n+1} dx.$$

Характерно, что метод интегрирования разложением в ряд абсолютно точен для полинома. То есть если разложение в ряд дает конечный ряд, то можно вычислить все коэффициенты в явном виде и уйти от погрешности метода, остается только погрешность округления.

### Метод интегрирования Гаусса

Гаусс предложил способ улучшения (уточнения) вычисления по методу Ньютона-Котеса. Его главная идея – оптимальный выбор узлов интерполяции. В методе Ньютона-Котеса для удобства вычисления коэффициентов Котеса были выбраны равноотстоящие узлы интерполяции. Однако этот вариант по точности не самый лучший выбор узлов. Выбирая узлы интегрирования Гаусса стремился увеличить реальный порядок погрешности интеграла.

Пусть подинтегральная функция есть полином m порядка  $f(x) = \sum_{k=0}^m p_k x^k$

Но ее интеграл будем оценивать по n+1 узлу  $\int_a^b f(x)dx = \sum_{k=0}^m p_k \int_a^b x^k dx \approx \sum_{i=0}^n c_i f_i$

Воспользуемся теперь исходным представлением в этих узлах  $f(x_i) = \sum_{k=0}^m p_k x_i^k$  и

подставим в формулу  $I = \int_a^b f(x)dx \approx \sum_{i=0}^n c_i \sum_{k=0}^m p_k x_i^k \approx \sum_{k=0}^m p_k \int_a^b x^k dx \approx \sum_{k=0}^m p_k \left( \sum_{i=0}^n c_i x_i^k \right)$

Отсюда мы получаем дополнительное ограничение - равенство  $\int_a^b x^k dx = \sum_{i=0}^n c_i x_i^k$

Из данного ограничения получаем систему линейных уравнений для коэффициентов  $c_i$ , откуда они могут быть найдены любым численным способом. С другой стороны значения этих коэффициентов будут зависеть от выбора узлов, что дает возможность подобрать узлы оптимальным образом. Для подбора узлов необходимо решить систему нелинейных уравнений, так как найденное ограничение для узлов  $x_i$  - нелинейное. Таким образом, получаем совместную систему линейных и нелинейных уравнений. Главная ее особенность – она может быть решена так, что решение будет удовлетворять любому выбору пределов интегрирования. В свою очередь это дает возможность вычисления оптимального расположения узлов и весовых коэффициентов  $C$  без учета конкретной задачи. Результаты вычислений помещаются в справочник и при необходимости могут быть использованы при конкретных вычислениях интегралов.

Очевидно, что при  $m=n$  метод Гаусса совпадает с методом Ньютона-

Котеса. Нет смысла и рассматривать вариант  $m < n$ , так как это ухудшение точности метода Ньютона-Котеса. Эффект применения метода Гаусса дает выбор  $n < m$ . Теперь порядок погрешности определяется величиной  $m$  а не  $n$ , что дает возможность повысить точность формулы Ньютона-Котеса до порядка  $m > n$ .

**Замечание 1.** Метод Гаусса дает возможность повышать точность однократного численного интегрирования. С помощью данного метода получают более точные формулы интегрирования для ручных расчетов в вычислительной математике. Использование формул Гаусса для машинных расчетов не является в большинстве случаев необходимым.

**Замечание 2.** Гаусс не единственный, кто предложил специальный метод уточнения формул численного интегрирования. В частности в методе Чебышева выбирают узлы так, чтобы формула была абсолютно точна для полинома порядка  $N$ . Расчет значений узлов требует решения СЛАУ из  $N$  уравнений. Для различных конкретных задач, были предложены формулы повышающие точность интегрирования. В целом эти методы учитывали особенность подинтегральных выражений. Среди наиболее известных можно отметить формулы Маркова, Эрмита, Бесселя.

## Лекция №7

### Раздел №2. Численные методы математического анализа

#### Тема 2.2 Численное дифференцирование и интегрирование

1. Дифференциальные уравнения. Задача Коши и краевая задача
2. Численное интегрирование дифференциальных уравнений
3. Решение задачи Коши. Одношаговые и многошаговые методы
4. Методы Рунге-Кутты : метод Эйлера, методы 2-го и 4-го порядка
5. Понятие о многошаговых методах Адамса, Башфорта и Милна
6. Сравнительная характеристика методов

Содержание: Дифференциальные уравнения. Задача Коши и краевая задача. Численное интегрирование дифференциальных уравнений. Решение задачи Коши. Одношаговые и многошаговые методы. Методы Рунге-Кутты: метод Эйлера, методы 2-го и 4-го порядка. Понятие о многошаговых методах Адамса, Башфорта и Милна. Сравнительная характеристика методов.

#### 1. Дифференциальные уравнения. Задача Коши и краевая задача

Решение дифференциальных уравнений (ДУ) – важнейшая задача прикладной математики. Значительное количество ДУ имеет явное решение, но еще большее количество уравнений может быть решено только численно. Вспомним некоторые определения и свойства ДУ:

01. ДУ – уравнение, содержащие производные.
02. Степенью или порядком ДУ называют максимальный порядок производной в уравнении.
03. ДУ называется нелинейным, если оно содержит нелинейную функцию от производных.

**04.** ДУ называется Обыкновенным ДУ, если оно не содержит частных производных. Существуют одиночные ОДУ и системы ОДУ. Можно показать, что любое ОДУ имеющее степень  $n$  может быть представлено системой из  $n$  ОДУ 1-го порядка. Таким образом, будем рассматривать решение ОДУ 1-го порядка и системы таких ОДУ.

Решение ДУ (в отличие от линейных или нелинейных уравнений) – функция, которая определяется на некотором отрезке. Как правило решение находится с точностью до неизвестной константы или несколько констант. Количество констант равно порядку ДУ. Для определения значения констант необходимы дополнительные начальные или краевые условия.

**05.** Задача решения ДУ называется начальной или задачей Коши, если заданы дополнительные условия только в начале отрезка, где ищется решение.

**06.** Задача решения ДУ называется краевой, если заданы дополнительные условия в начале и конце отрезка, где ищется решение.

**07.** Задача решения ДУ называется смешанной, если заданы дополнительные условия в начале и конце отрезка для одной переменной и начальные условия для другой переменной.

## **2. Численное интегрирование дифференциальных уравнений**

Для поиска решения ДУ необходимо провести интегрирование этого уравнения. Некоторые типы ДУ интегрируются в явном виде, то есть решение получается как явно заданная функция. Но большая часть ДУ могут быть решены только с помощью численных методов.

Методы численного решения ДУ принято делить на сеточные и аппроксимационные. Аппроксимационные методы основаны на поиске решения ДУ как некой аппроксимационной функции, которая строится на каком-либо методе аппроксимации (интерполяционный полином, аппроксимация в виде разложения по базисных функциям и т. д.). Сеточные методы основаны на сеточной аппроксимации.

## **3. Решение задачи Коши. Одношаговые и многошаговые методы**

Рассмотрим методы решения начальной задачи Коши для обыкновенного линейного дифференциального уравнения. Численные методы решения подобной задачи делятся на 2 группы:

6. Приближенно-аналитические методы (метод степенного ряда Тейлора, метод приближений Пикара).

7. Сеточные методы.

Наиболее распространены в практике решения вычислительных задач на ЭВМ сеточные методы, которые можно разделить на одношаговые и многошаговые.

Сеточные методы предполагают введение сеточной аппроксимации, когда непрерывную функцию решения задачи заменяют сеточной функцией, которая определена только в узлах сетки (где она обычно совпадает с непрерывным решением задачи).

Рассмотрим стандартную задачу:

Будем искать решение линейного дифференциального уравнения 1-го порядка

$dy/dx = y' = f(x,y)$  (1) здесь  $y'$  это первая производная функции  $y(x)$  по  $x$ . Решение ищем на отрезке  $[a,b]$  при заданном начальном условии  $y(a)=y_0$ . Построим на отрезке сетку, для этого зададим шаг  $h=(b-a)/n$  и узлы  $x_i=a+ih$ , тогда можно определить сеточную аппроксимацию как совокупность значений  $y_i=y(x_i)$ . Необходимо найти решение  $y_i$  в узлах  $x_i$ , которое затем с помощью интерполяции полиномами Лагранжа или Ньютона можно представить в виде непрерывной функции приближенного решения.

Для решения задачи Коши служат множество сеточных методов, которые принято делить на одношаговые и многошаговые. Сеточный метод называется одношаговым, если значение на новом шаге зависит только от одного предшествующего шага. Множество одношаговых методов называют методами Рунге-Кутты. Одношаговые методы характеризуются итерационной формулой вычисления сеточной функции и порядком глобальной погрешности. Сеточные метод называется многошаговым если значение на новом шаге зависит от нескольких предшествующих шагов. Многошаговые методы обычно делятся по количеству начальных шагов, которые необходимы для старта решения.

Многошаговые методы построены так, что значение последующего шага определяется по значению нескольких предшествующих. При этом потребуется дополнительно вычислить несколько начальных значений. Для этих целей используются одношаговые методы (например методы Рунге-Кутты). Таким образом, многошаговый метод не используется в чистом виде, а всегда образует композицию с каким либо одношаговым методом. Таким образом, приходится учитывать взаимосвязь методов, которые могут иметь разные порядки погрешности. Наиболее известными примерами многошаговых методов являются методы Адамса и Милна.

#### 4. Методы Рунге-Кутта : метод Эйлера, методы 2-го и 4-го порядка

Для решения заданной задачи служит множество одношаговых методов, которые называются методами Рунге-Кутты. Сеточный метод называется одношаговым, если значение на новом шаге зависит только от одного предшествующего шага. Одношаговые методы характеризуются итерационной формулой вычисления сеточной функции и порядком глобальной погрешности.

**Замечание.** Ранее уже рассматривалась разница порядка погрешности вычисления на 1 шаге и в результате выполнения всех шагов вычисления. Первую погрешность принято называть локальной, а вторую глобальной. Было установлено, что локальная погрешность однократного метода интегрирования имеет на 1 больший порядок точности, чем глобальная погрешность в многократном варианте. Эта ситуация характерна для всех сеточных вычислений, когда задается сетка с равным шагом.

Рассмотрим несколько методов Рунге-Кутта различных порядков точности.

#### Метод Эйлера

Наиболее простым и неточным методов Рунге-Кутта является метод предложенный Л.Эйлером. Рассмотрим разложение решения уравнения (1) в

ряд Тейлора в точке  $x_0$ :  $y_1 = y_0 + y'(x_0) * h + O(h^2)$ , где  $y_0 = y(x_0)$ ,  $y_1 = y(x_0 + h)$ . Из уравнения (1) можно выразить производную как  $y'(x_0) = f(x_0, y_0)$ . Тогда, после подстановки получаем:  $y_1 = y_0 + f(x_0, y_0) * h + O(h^2)$ . По аналогии, считая что все шаги строятся так же, получим итерационную формулу Эйлера

$$y_{i+1} = y_i + f(x_i, y_i) * h + O(h^2) \quad (2)$$

Ее локальная погрешность имеет второй порядок по  $h$ , глобальная имеет порядок на 1 меньше, т.е. первый. Поэтому метод Эйлера это метод Рунге–Кутта первого порядка.

### Усовершенствованный метод Эйлера

Уточним метод Эйлера. Для этого учтем следующий элемент в разложении в ряд Тейлора.

$$y_1 = y_0 + y'(x_0) * h + y''(x_0) * h^2 / 2 + O(h^3) \quad (3)$$

При этом учтем, что  $y'(x_0) = f(x_0, y_0)$ . При разложении в ряд Тейлора с шагом  $h/2$  получаем  $y_0 + y'(x_0) * h/2 + O(h^2) = y(x_0 + h/2)$ , аналогично для производной  $y'(x_0 + h/2) = y'(x_0) + y''(x_0) * h/2 + O(h^2) \Rightarrow h * y'(x_0 + h/2) = y'(x_0) * h + y''(x_0) * h^2 / 2 + O(h^3) \quad (4)$

Подставив (4) в (3), получим  $y_1 = y_0 + y'(x_0 + h/2) * h + O(h^3)$ . Тогда,  $y(x_0 + h/2) = f(x_0 + h/2, y^*)$ , где введено обозначение  $y^* = y(x_0 + h/2)$ .

Тогда усовершенствованный метод Эйлера имеет двойную формулу:

$$y^*_i = y_i + f(x_i, y_i) * h/2 \text{ и } y_{i+1} = y_i + h * f(x_i + h/2, y^*_i) \quad (5).$$

Построенный нами усовершенствованный метод называется методом средней точки. Метод имеет 3-й порядок для локальной и 2-й порядок глобальной погрешности.

### Метод «предиктор-корректор»

Другим вариантом метода Рунге-Кутта 2-го порядка точности является метод «предсказания-уточнения» («предиктор-корректор»). Он предполагает использование 2-х формул грубого приближения (предсказания) и уточняющей формулы. Рассмотрим теперь формулы данного метода и докажем, что они справедливы:

- грубое приближение  $y^*_0 = y_0 + h * f(x_0, y_0) + O(h^2)$
- уточнение  $y_1 = y_0 + h/2 [f(x_0, y_0) + f(x_0 + h, y^*_0)]$

Продифференцируем первую формулу и учтем формулу исходного ОДУ (1):

$$f(x_0, y_0) = y'_0 \Rightarrow f(x_0 + h, y^*_0) = y^*_0{}' = y'_0 + h y''_0,$$

Теперь рассмотрим разложение решения в ряд Тейлора до 2-го порядка (3) и подставим полученное выше выражение в формулу уточнения:

$$y_1 = y_0 + h/2 [2y'_0 + h * y''_0] = y_0 + h y'_0 + h^2 / 2 * y''_0$$

Сравним полученное выражение с разложением (3) и видим, что разница только в остаточном члене 3-го порядка  $\Rightarrow$  можно утверждать, что формулы метода («предиктор-корректор» эквивалентны следующей формуле, имеющей 3-й порядок локальной погрешности и 2-й порядок глобальной погрешности.

$$y_1 = y_0 + h y'_0 + h^2 / 2 * y''_0 + O(h^3)$$

### Методы Рунге-Кутта 4-го порядка

Наиболее используемые на практике методы Рунге-Кутта имеют 4-й порядок глобальной погрешности. Этот вариант компромиссный между точностью и сложностью формул вычисления.

$$y_{i+1} \approx y_i + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4), \quad (6)$$

$$k_4 = hf(x_i + h, y_i + k_3); k_3 = hf\left(x_i + \frac{h}{2}, y_i + \frac{k_2}{2}\right); k_1 = hf(x_i, y_i); k_2 = hf\left(x_i + \frac{h}{2}, y_i + \frac{k_1}{2}\right).$$

Этот метод называется стандартным методом 4-го порядка. Он предполагает предварительное вычисление 4-х коэффициентов, а затем вычисление следующего приближения по формуле (6).

Мерсон предложил несколько усложненный вариант метода 4-го порядка, который называется методом Рунге-Кутты-Мерсона:

$$\begin{aligned} P_1 &= f(x_i, y_i), P_2 = f\left(x_i + \frac{h}{3}, y_i + \frac{h}{3}P_1\right) \\ P_3 &= f\left(x_i + \frac{h}{3}, y_i + \frac{h}{6}(P_1 + P_2)\right), P_4 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{8}P_1 + \frac{3h}{8}P_2\right), \\ P_5 &= f\left(x_i + h, y_i + \frac{h}{2}P_1 - \frac{3h}{2}P_3 + 2hP_4\right). \\ \tilde{y}_{i+1} &= y_i + \frac{h}{2}(P_1 - 3P_3 + 4P_4); \text{ и } y_{i+1} = y_i + \frac{h}{6}(P_1 + 4P_4 + P_5) \end{aligned}$$

Особенность метода Мерсона состоит в вычислении 2-х значений сеточной функции в следующем узле. Такое усложнение алгоритма дает возможность оценить погрешность на каждом шаге по следующей формуле:

$$D \approx \frac{1}{5} |y_{i+1} - \tilde{y}_{i+1}|.$$

При выводе своих формул Мерсон исходил фактически из 2-х методов 4-го и 5-го порядка глобальной погрешности. Если считать формулу 5-го порядка (как гораздо более точную) за точное значение, то разность этих 2-х формул по модулю примерно равны погрешности формулы 4-го порядка. Таким образом, увеличение числа коэффициентов и 2 формулы результата позволяют не только оценить значение на следующем шаге, но и сделать оценку погрешности. Локальная погрешность при этом данных методов одинакова и имеет 5 порядок; глобальная – 4 порядок.

В методах Рунге-Кутты можно выделить следующие особенности:

1. Любой метод может быть выведен с учетом разложения в ряд Тейлора. Для этого необходимо раскладывать в ряд Тейлора функцию в одной или нескольких точках.
2. Для получения уравнения первого порядка достаточно использования разложения в одной точке, для метода второго порядка – добавляется еще одна точка и т.д.. Аналогичным образом увеличивается и количество дополнительных формул (которые иногда называют этапами). Таким образом, за повышение точности требуется заплатить добавлением новых этапов.
3. Формулы Рунге-Кутты имеют прямую аналогию с методами численного интегрирования (например метод средней точки).

### **Оценка погрешности метода Рунге-Кутты и выбор шага**

При вычислении решения дифференциального уравнения часто задается предельная погрешность. Таким образом, максимальная погрешность решения

должна быть ограничена этой величиной. Этот критерий требует подбора такого шага сетки, чтобы глобальная погрешность не превысила предельной погрешности. Необходимо выбрать оптимальный шаг. Если шаг велик, то погрешность превышает заданную. Если выбран слишком малый шаг, то потребуются вычисления, которые совершенно излишни. Для оценки шага в методах Рунге-Кутты существуют 2 варианта:

1. Воспользовавшись формулами Рунге можно оценить глобальную погрешность и найти оптимальный шаг. Для использования формул Рунге необходимо произвести вычисления дважды с шагом  $h$  и  $h/2$ .
2. Можно использовать метод Мерсона, где имеется возможность контроля шага на каждом вычислении. Таким образом, выбираем начальный шаг  $h$ , вычисляем значение в следующей точке и оценку погрешности. Если погрешность больше заданной, шаг уменьшаем в два раза, если погрешность в 64 или более раз меньше заданной, то шаг можно увеличить в два раза.

Первый вариант можно назвать выбором оптимального равномерного шага, а второй вариант методом корректировки или контроля шага. Какой из вариантов выбрать – определяется требованием задачи и ограничений на вычислительные ресурсы.

## 5. Понятие о многошаговых методах Адамса, Башфорта и Милна

### Многошаговые методы

Многошаговые методы построены так, что значение последующего шага определяется по значению нескольких предшествующих. При этом потребуется дополнительно вычислить несколько начальных значений. Для этих целей используются одношаговые методы (например методы Рунге-Кутты). Таким образом, многошаговый метод не используется в чистом виде, а всегда образует композицию с каким либо одношаговым методом. Таким образом, приходится учитывать взаимосвязь методов, которые могут иметь разные порядки погрешности. Наиболее известными примерами многошаговых методов являются методы Адамса и Милна.

### Метод Адамса

Пусть задано дифференциальное уравнение типа (1). Для его решения проинтегрируем левую и правую часть уравнения на отрезке между соседними узлами:

$$y' = f(x, y), [a; b],$$

$$\int_{x_i}^{x_{i+1}} \frac{dy}{dx} = \int_{x_i}^{x_{i+1}} f(x, y(x)) dx = I$$

$$y(x_{i+1}) - y(x_i) = I \Rightarrow y_{i+1} = y_i + I$$

$[x_i, x_{i+1}]$  – отрезок \_интегрирования

В результате получим итерационную формулу, в которой используется интеграл

(что показывает глубокую связь и возможность взаимного преобразования между ДУ и интегральными уравнениями). Таким образом, значение в следующей точке можно вычислить, если вычислен или известен интеграл. Чтобы вычислить интеграл  $I$  воспользуемся стандартным методом - подынтегральное выражение заменим на интерполяционный полином Ньютона (в отличие от метода Ньютона-Котеса где использовался полином Лагранжа).

Следует учитывать, что интеграл зависит от значения в узле  $i+1$ . Это приводит к формированию уравнения относительно  $y_{i+1}$ . Можно избавиться от решения уравнения, если использовать формулу Ньютона как экстраполяцию. Следовательно, возможно получение 2-х видов формул – экстраполяционной и интерполяционной. Интерполяционная формула имеет вид итерационной формулы и может вычисляться многократно. В зависимости от выбора порядка полинома Ньютона  $n$  можно построить разные формулы, разной точности, с разным количеством дополнительно заданных начальных точек.

Формулы которые используют экстраполяцию называются формулами предсказания. Интерполяционные формулы называются формулами уточнения. Формулы предсказания в методе Адамса названы в честь математика построившего их для основных порядков - формулами Башфорта, а формулы уточнения были выведены другим ученым и носят имя Моултона:

$$n = 0$$

$$y_{i+1} = y_i + h * f(x_i, y_i) - \text{формула прогноза,}$$

$$y_{i+1} = y_i + hf(x_{i+1}, y_{i+1}) - \text{формула уточнения}$$

другие формулы прогноза:

$$n = 1; \quad f_i = f(x_i, y_i)$$

$$y_{i+1} = y_i + h[3f_i - f_{i-1}]/2$$

$$n = 2$$

$$y_{i+1} = y_i + h[23f_i - 16f_{i-1} + 5f_{i-2}]/12$$

$$n = 3$$

$$y_{i+1} = y_i + h[55f_i - 59f_{i-1} + 37f_{i-2} - 9f_{i-3}]/24$$

другие формулы коррекции:

$$n = 1 \quad y_{i+1} = y_i + h/2(f_{i+1} + f_i)$$

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

$$n = 3$$

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

При увеличении порядка полинома возрастает точность формул, но возрастает так же и их сложность и количество требуемых начальных точек. Таким образом, при использовании метода Адамса появляется значительное количество вариантов использования пар формул прогноз-коррекция. Кроме того, формулы коррекции имеют итерационный вид и могут использоваться для многократного уточнения. Это дает возможность реализовать дополнительные варианты с многократным уточнением (например до достижения заданной

погрешности). Фактически это означает, что многократные вычисления могут быть значительно точнее чем однократные => повышается порядок точности метода.

Характерной особенностью многошаговых методов стала возможность оценки погрешности в этих методах. Итерационный характер формулы коррекции дает возможность вывести оценку для погрешности. Так для формул 4-го порядка получена оценка:  $D \approx |19*(Y^M_{i+1} - Y^B_{i+1})/270|$   
Здесь буквы М и Б обозначают формулы Моултона и Башфорта соответственно.

### Метод Милна

В целом метод Милна очень похож на метод Адамса. В нем так же используется интегрирование исходного ДУ. Однако в методе Милна интегрирование ведется по отрезку  $[x_{i-3}, x_{i+1}]$ , что является серьезным отличием от метода Адамса. Фактически Милн использует не отдельные значения в узлах, а интегральную характеристику по всему начальному отрезку. С одной стороны это дает большую точность, а с другой стороны требует фиксации порядка интерполяционного полинома, что не дает Милну строить множество вариантов формул прогноза и коррекции.

Рассмотрим интегрирование ДУ Милном:

$$y_{i+1} = y_{i-3} + \int_{x_{i-3}}^{x_{i+1}} f(x, y(x)) dx$$

Милн при выводе своей формулы зафиксировал порядок полинома (n=4), взял 5 дополнительных точек и интерполирование производил на всем отрезке, что дало возможность использовать симметрию задачи. В результате получены следующие формулы:

$n=4, y'_{i+1} = y_{i-3} + 4*h*(2f_i - f_{i-1} + 2f_{i-2})/3$  - формула прогноза

$y_{i+1} = y_{i-1} + 4*h*[4f_i + f_{i-1} + f(x_{i+1}, y'_{i+1})]/3$  - формула коррекции

Аналогично методу Адамса для формул Милна можно вывести формулу оценки погрешности  $D \approx |(Y_{i+1} - Y'_{i+1})/29|$ .

### 6. Сравнительная характеристика методов

Сравнение одношаговых и многошаговых методов достаточно условно, так как их отличие определяется методом использования разностных формул. Порядок точности и сложность формул и скорость вычисления в целом зависят больше от выбора конкретного метода чем от его одношагового или многошагового характера. Разумеется, одношаговые методы проще в том смысле, что они однородны, а многошаговый метод может выступать только в композиции с одношаговым (для расчета начальных точек). С другой стороны, сеточные методы могут обладать неприятным для применения свойством неустойчивости (это свойство будет рассмотрено позже). Это явление чаще встречается для многомерных ДУ, но для некоторых задач возможно и в одномерных ДУ. Устойчивость многошаговых схем, имеющих формулы коррекции всегда выше. Удобны многошаговые методы для оценочных вычислений (без итераций) и контроля шага.

Сравнение между собой методов Рунге-Кутты определяется прежде всего сложностью задачи. Метод Эйлера сильно не точен и используется обычно только для учебных целей. Методы 2-го порядка достаточны для решения простых задач. Сложные задачи требуют методов 4-го порядка. Более высокий порядок точности редко используется на практике. Метод Мерсона используется при необходимости автоматического контроля шага.

Сравнение многошаговых методов дает предпочтение прежде всего методу Милна. Этот метод имеет существенно более простой вид по сравнению с аналогичными по точности вариантами метода Адамса. Поэтому на практике метод Милна предпочтительнее. Метод Адамса интересен многообразием вариантов его реализации. Выбор различных формул прогноза и коррекции, итерационный вариант позволяют подобрать наиболее удобный метод для конкретной задачи.