# Лекция 1

**Тема 1. Введение**

# §1. Предмет вычислительной математики

В течение большей части истории развития математики главные усилия математиков были направлены на создание строгой логической базы математических методов, расширение множества объектов, к которым эти методы применимы, изучение качественной природы математических объектов. Гораздо меньше внимания уделялось разработке методов доведения математических исследований до числового результата, а это зачатую является интересной, трудной и чрезвычайно важной для практики задачей.

В самых разнообразных областях современной науки и техники приходится встречаться с такими математическими задачами, для которых невозможно получить точное решение классическими методами (как говорят – на кончике пера) или же решение может быть получено в таком сложном виде, который совершенно неприемлем для практического использования. Так, например, очень часто приходится встречаться с необходимостью решения систем линейных алгебраических уравнений с десятками и сотнями неизвестных, с задачей отыскания корней алгебраических уравнений высоких степеней и корней трансцендентных уравнений, с необходимостью решения систем дифференциальных уравнений, которые не интегрируются в элементарных функциях и т.д.

Количество задач такого рода особенно сильно возросло в связи с бурным развитием вычислительной техники. Новые вычислительные средства сделали возможным решение задач, ранее нерешаемых. Возникла также необходимость пересмотреть существующие вычислительные методы с точки зрения реализации их на новых вычислительных средствах.

По этим причинам сложилась новая область математики, которая призвана разрабатывать методы доведения до числового результата решений основных задач математического анализа, алгебры и геометрии и пути использования для этой цели современных

вычислительных средств. Эта область и получила название

*вычислительной математики*.

# §2. Метод и задачи вычислительной математики

Круг задач, с которыми приходится иметь дело в вычислительной математике, очень широк. Разнообразны и методы, применяемые для решения этих задач. Однако можно заметить одну общую идею этих методов. Эта идея проще всего выражается в терминах функционального анализа. Поэтому мы введем предварительно некоторые важнейшие понятия функционального анализа.

**1. Функциональные метрические пространства.** Основным предметом исследования в классическом математическом анализе является числовая функция. С появлением понятия функции одной и нескольких переменных, функции точки в евклидовом пространстве начался современный этап развития математики. Начиная с работ Ньютона и Лейбница и до конца XIX века подавляющее большинство математических исследований так или иначе было связано с этим понятием. Главным предметом изучения были числовые функции и их системы, заданные в *n*-мерной области, т.е. на некотором множестве *n*- мерного евклидова пространства.

Двадцатый век внес много нового в эту картину. Особо важную роль начинают играть понятия о функциональных множествах, о функциональных пространствах и функциональных операторах, т.е. о функциях, аргументами и значениями которых являются элементы функциональных пространств. Вместо евклидовых пространств рассматриваются абстрактные пространства, элементы которых могут иметь самую различную природу. Так, например, вводится понятие *метрического пространства R* как абстрактного множества, для любых двух элементов *x* и *y* которого определено понятие расстояния ρ(*x, y),* удовлетворяющее следующим условиям:

1. ρ(*x, y) ≥ 0,* причем ρ(*x, y)=0* тогда и только тогда, когда *x* совпадает с *y.*

2. ρ(*x, y) =* ρ(*y, x).*

3. ρ(*x, y) ≤* ρ(*x, z) +* ρ(*z, y)* для любых трех элементов *x, y, z,*

принадлежащих *R* (аксиома треугольника).

Евклидовы пространства с обычным определением расстояния удовлетворяют все этим условиям. Но могут быть и другие метрические пространства. Так, рассмотрим множество всевозможных непрерывных функций, заданных на отрезке *[a, b].* Для любых двух таких функций *x(t)* и *y(t)* определим расстояние ρ(*x, y)* равенством:

ρ(*x, y) = max | x(t) – y(t) |* по всем значениям аргумента *t* ϵ *[a, b] (1.1)*

Нетрудно проверить, что так определенное расстояние удовлетворяет всем трем поставленным выше условиям. Таким образом, мы получили функциональное метрическое пространство, которое обычно называют пространством *C[a, b]* – пространством функций, непрерывных на отрезке *[a, b].*

Другим важным классом функциональных пространств являются пространства 𝐿𝑝*.* (Здесь *p* – действительное число *≥* 1*).* Говорят, что измеримая на [a, b] функция *f(t)* принадлежит 𝐿𝑝*,* если суммируема

|𝑓(𝑡)|𝑝. Две функции *x(t)* и *y(t),* принадлежащие 𝐿𝑝, считаются эквивалентными, если они могут отличаться друг от друга лишь на множестве меры нуль. Расстояние ρ(*x, y)* в пространстве 𝐿𝑝 определяется следующим образом:

ρ(*x,y) =* [∫𝑏

𝑎

|𝑥

(𝑡)

− 𝑦

(𝑡

)|𝑝

1|𝑝

] *(1.2)*

Так определенное расстояние удовлетворяет всем трем поставленным выше условиям.

В каждом метрическом пространстве можно говорить об окрестности данной точки. Назовем *ε – окрестностью точки x* некоторого метрического пространства *R* совокупность его точек *y,* для которых выполняется неравенство

ρ(*x,y) < ε (1.3)*

В пространстве *C[a, b]* это будет совокупность всех непрерывных на

*[a, b]* функций, лежащих в полосе *x(t) ± ε.*

В пространстве 𝐿𝑝 это будет совокупность всех функций, принадлежащих 𝐿𝑝, для которых

∫𝑏 |𝑥(𝑡) − 𝑦(𝑡)|𝑝 *dt <* 𝜀𝑝*. (1.4)*

𝑎

При этом в отдельных точках отклонение *y(t)* от *x(t)* может быть очень большим, а зато в других точках будет очень малым.

В вычислительной математике часто приходится заменять одну функцию *x(t)* другой функцией, более удобной для вычислительных целей и в каком-то смысле близкой к первой. Обычно эту вторую функцию берут в некоторой *ε-окрестности* первой. Если *ε- окрестность* берется в пространстве *C[a, b],* то говорят о *равномерном приближении* функции *x(t).* Если же *ε-окрестность* берут в пространстве 𝐿𝑝, то говорят о *приближении в среднем*. В частности, при *p = 2* говорят о *среднеквадратичном* приближении.

# Функции, определенные на функциональных пространствах.

Так же, как в классическом математическом анализе, можно ввести понятие функции, аргументом и значением которой будут элементы абстрактных пространств.

Пусть нам даны два абстрактных пространства 𝑅1 и 𝑅2. Пусть каждому элементу *x ϵ* 𝑅1 поставлен в соответствие элемент *y ϵ* 𝑅2.

Тогда будем говорить, что нам задан оператор

*y = A(x) (1.5)*

с областью определения 𝑅1 и областью значений, принадлежащих 𝑅2. В частности, если 𝑅2 является областью действительных или комплексных чисел, то оператор *A(x)* называется функционалом. Простым примером функционала в пространстве *C[a,b]* является определенный интеграл

𝐼(𝑥) = ∫𝑏 𝑥(𝑡)𝑑𝑡

𝑎

(1.6)

Область математики, изучающая свойства функциональных пространств, носит название *функционального анализа*.

1. **Метод вычислительной математики.** Теперь можно охарактеризовать метод вычислительной математики.

В вычислительной математике приходится сталкиваться с самыми различными задачами. Многие из этих задач могут быть записаны в виде

*y=A(x), (1.7)*

где *x* и *y* принадлежат заданным пространствам 𝑅1 и 𝑅2 и *A(x) –* некоторый заданный оператор*.* Задача состоит в отыскании либо *y* по заданному *x*, либо в отыскании *x* по заданному *y.* Далеко не всегда с помощью средств современной математики удается точно решить эти задачи, применяя конечное число шагов. В этих случаях и прибегают к вычислительной математике. Иногда задача может быть решена и точно, но методы классической математики дают ответ после громоздких и трудоемких вычислений. Поэтому в задачи вычислительной математики входит также разработка приемов и методов наиболее рационального решения конкретных задач. Как это делается в различных случаях мы узнаем из дальнейшего изучения предмета. Сейчас же мы выскажем некоторые общие соображения.

Основным методом, при помощи которого в вычислительной математике решают поставленные выше задачи, является замена пространств 𝑅1 и 𝑅2 и оператора *A* другими пространствами 𝑅̅̅1̅ и

𝑅̅̅2̅ и оператором 𝐴̅*,* более удобными для вычислительных целей.

Иногда бывает достаточно произвести замену только пространств 𝑅1 и

𝑅2 или даже одного из них. Иногда достаточно заменить только оператор. Замена должна быть сделана так, чтобы решение новой задачи

𝑥̅ ϵ

̅𝑅̅̅1̅,

𝑦̅ = 𝐴̅(𝑥̅), (1.*8*)

𝑦̅ ϵ 𝑅̅̅2̅ , было в каком-то смысле близким к точному решению

исходной задачи (7) и его можно было бы отыскать с сравнительно небольшими трудностями.

Например, пусть необходимо вычислить интеграл

y = ∫𝑏 𝑓(𝑥)𝑑𝑥 ,

𝑎

где *f(x) –* непрерывная функция, причем неопределенный интеграл не берется в элементарных функциях. Чтобы получить достаточно точное приближенное значение интеграла, можно идти двумя путями.

1. Заменим функцию *f(x)* алгебраическим многочленом *P(x),* равномерно приближающим функцию *f(x)* на отрезке *[a, b]* с необходимой степенью точности. (Как будет показано дальше, это

всегда можно сделать). Тогда вместо интеграла y = ∫𝑏 𝑓(𝑥)𝑑𝑥

𝑎

будем

находить интеграл y = ∫𝑏 𝑃(𝑥)𝑑𝑥

𝑎

. Вычисление такого интеграла не

составляет труда. Здесь мы, не меняя функционала *A(f)* = ∫𝑏 𝑓(𝑥)𝑑𝑥 , заменяем пространство *C[a, b],* которому принадлежит функция *f(x),* пространством многочленов и вместо функции *f(x)* берем многочлен из некоторой ее *ε-окрестности.*

𝑎

1. Из определения интеграла ∫𝑏 𝑓(𝑥)𝑑𝑥

𝑎

построить интегральную сумму ∑𝑛

𝑖=1

следует, что всегда можно

𝑓(𝑥𝑖)∆𝑖, которая будет

достаточно близка к значению интеграла. Следовательно, вместо

вычисления интеграла y = ∫𝑏 𝑓(𝑥)𝑑𝑥

𝑎

можно решать другую задачу –

задачу вычисления конечной суммы *y* = ∑𝑛 𝑓(𝑥𝑖)∆𝑖. Здесь мы уже

𝑖=1

заменяем функционал *A(f)*=∫𝑏 𝑓(𝑥)𝑑𝑥

𝑎

другим функционалом

̅𝐴̅̅(̅̅𝑓̅̅) = ∑𝑛

𝑖=1

𝑓(𝑥𝑖)∆𝑖.

Резюмируя сказанное выше, мы отметим, что перед вычислительной математикой стоят следующие основные задачи:

1. Приближение множеств в функциональных пространствах.
2. Приближение операторов, заданных на функциональных пространствах.
3. Разработка рациональных алгоритмов и методов решения задач в условиях применения современных вычислительных средств.

# Лекция 2

**Тема 2. Основные понятия элементарной теории погрешностей**

**§ 2.1. Источники и классификация погрешностей результатов численного решения задач.** Как отмечалось в первой лекции, при решении прикладной задачи с использованием ЭВМ получить точное решение задачи практически невозможно. Получаемое решение почти всегда содержит погрешность, т.е. является приближенным.

Наличие погрешности решения обусловлено рядом причин.

Перечислим их.

1. При решении задач на ЭВМ имеют дело, как правило, с математической моделью физического явления, которая является его приближенным описанием. В связи с этим получаемые в результате вычислений характеристики процесса или явления содержат погрешность, величина которой зависит от степени адекватности модели реальному процессу.
2. Исходные данные, как правило, содержат погрешности, поскольку они получаются в результате измерений либо являются результатом решения некоторых задач.
3. Применяемые при решении задачи методы, как отмечено в первой лекции, в большинстве случаев являются приближенными.
4. При вводе исходных данных в ЭВМ, выполнении арифметических операций и выводе результатов на печать или дисплей монитора производятся округления, определяемые разрядностями устройств ввода/вывода и ЭВМ.

Пусть *y* – точное значение величины, вычисление которой является целью поставленной задачи, а 𝑦∗ – ее приближенное значение. Соответствующая первым двум из указанных причин погрешность 𝛿н𝑦∗ называется *неустранимой погрешностью.* Такое название обусловлено тем, что математическая модель и исходные данные вносят в решение ошибку, которая не может быть устранена далее. Единственный способ

уменьшить эту погрешность – перейти к более точной математической модели и задать более точные исходные данные. Но это не всегда возможно.

Погрешность 𝛿м𝑦∗, источником которой является метод решения задачи, называется *погрешностью метода,* а погрешность 𝛿в𝑦∗, возникающая при вводе, выводе и вычислениях, – *вычислительной погрешностью.*

Таким образом, полная погрешность результата решения задачи на ЭВМ 𝛿𝑦∗=y – 𝑦∗ складывается из трех составляющих: неустранимой погрешности, погрешности метода и вычислительной погрешности, т.е.

𝛿𝑦∗ = 𝛿н𝑦∗+ 𝛿м𝑦∗+ 𝛿в𝑦∗.

Будем считать, что математическая модель фиксирована и входные данные задаются извне, так что повлиять на значение величины 𝛿н𝑦∗ в процессе решения задачи нельзя. Однако это не означает, что предварительные оценки величины неустранимой погрешности не нужны. Достоверная информация о порядке величины

𝛿н𝑦∗ позволяет осознанно выбрать метод решения задачи и разумно задать его точность. На практике исходят из того, что погрешность метода должна быть на порядок (в 2 – 10 раз) меньше неустранимой погрешности. Большее значение 𝛿м𝑦∗ощутимо снижает точность результата, меньшее – требует увеличения затрат, практически мало влияя на значение полной погрешности.

Величина вычислительной погрешности в основном определяется характеристиками используемой ЭВМ. Желательно, чтобы величина

𝛿в𝑦∗ была хотя бы на порядок меньше величины погрешности метода.

**§ 2.2. Приближенные числа. Абсолютная и относительная погрешности.** Итак, как отмечено выше, числа, получаемые при решении на ЭВМ прикладных задач, являются приближенными. Следовательно, вопрос о точности результатов, т.е. о мере их уклонения от истинных значений, в теории и практике приближенных вычислений приобретает особое значение. Начнем его рассмотрение с введения основных понятий элементарной теории погрешностей.

1. **Абсолютная и относительная погрешности.** Пусть *а* – точное (неизвестное) значение некоторой величины, 𝑎∗ – приближенное (известное) значение той же величины (*приближенное число*). *Ошибкой (погрешностью)* приближенного числа 𝑎∗ называется разность *а* – 𝑎∗ между точным и приближенным значениями.

Простейшей количественной мерой ошибки является *абсолютная погрешность*

*∆(*𝑎∗) *= | а* – 𝑎∗|. (2.1)

Однако по величине абсолютной погрешности не всегда можно сделать правильное заключение о качестве приближения. Действительно, если *∆(*𝑎∗) = 0.1, то следует ли считать точность большой или малой? Ответ существенным образом зависит от принятых единиц измерения и масштабов величин. Например, если *a ≈* 0.3, то точность приближения невысока. Если же *a ≈* 3∙108, то точность следует признать очень высокой. Таким образом, естественно соотнести погрешность величины и ее значение, для чего вводят понятие *относительной погрешности* (при *а ≠ 0)*

𝛿(𝑎∗) = | 𝑎 – 𝑎∗| = ∆(𝑎∗)

. (2.2)

|𝑎| |𝑎|

Очевидно, что относительные погрешности не зависят от масштабов величин. В частности, для приведенного выше примера

𝛿(𝑎∗) *≈* 0.33 = 33% в первом случае и 𝛿(𝑎∗) *≈* 0.33∙10−9 = 0.33∙10−7% во втором.

Так как значение *а* неизвестно, то непосредственное вычисление величин *∆(*𝑎∗) и 𝛿(𝑎∗) по формулам (1) и (2) невозможно. Более реальная и часто поддающаяся решению задача состоит в получении оценок вида

*|* 𝑎 – 𝑎∗| ≤ ̅∆̅̅(𝑎∗), (2.3)

|𝑎 – 𝑎∗|

|𝑎|

≤ ̅𝛿̅̅̅(𝑎∗), (2.4)

где ∆̅*(*𝑎∗) и 𝛿̅(𝑎∗) – известные величины, которые мы будем называть *верхними границами* (или просто *границами*) *абсолютной* и *относительной погрешностей*.

Если величина ∆̅*(*𝑎∗) известна, то неравенство (4) будет

выполнено, если положить

𝛿̅(𝑎∗) = ∆̅(𝑎∗) . (2.5)

|𝑎|

Точно так же если величина 𝛿̅(𝑎∗) известна, то следует положить

∆̅*(*𝑎∗) = |𝑎| 𝛿̅(𝑎∗) (2.6)

Поскольку значение *a* неизвестно, при практическом применении формулы (5), (6) заменяют приближенными равенствами

∗ ∆̅(𝑎∗) ̅ ∗ ∗ ∗

𝛿(𝑎 ) ≈ , ∆*(*𝑎 ) ≈ |𝑎 |𝛿(𝑎

|𝑎∗|

) (2.7)

1. **Правила записи приближенных чисел.** Пусть приближенное число 𝑎∗ задано в виде конечной десятичной дроби:

𝑎∗ = 𝛼𝑛𝛼𝑛−1…𝛼0. 𝛽1𝛽2 … 𝛽𝑚.

Значащими цифрами числа 𝑎∗ называют все цифры в его записи, начиная с первой ненулевой слева.

Пример 1. У чисел 𝑎∗= 0.0103 и 𝑎∗ = 0.0103000 значащие цифры подчеркнуты.

Значащую цифру называют *верной*, если абсолютная погрешность числа не превосходит единицы разряда, соответствующего этой цифре.

Пример 2. Если ∆̅*(*𝑎∗) = 2∙10−6, то число 𝑎∗ = 0.0103000 имеет 4

верные значащие цифры (они подчеркнуты).

Распространенной ошибкой является отбрасывание последних значащих нулей (даже если они представляют верные цифры).

З а м е ч а н и е. Верная цифра приближенного числа, вообще говоря, не обязана совпадать с соответствующей цифрой в записи точного числа. Таким образом, термин «верная цифра» не следует понимать буквально.

Пример 3. Пусть *a* = 1.00000, 𝑎∗= 0.99999. Тогда *∆(*𝑎∗)=0.00001 и у числа 𝑎∗= 0.99999 все подчеркнутые цифры верные, хотя они и не совпадают с соответствующими цифрами числа *a.*

Если число 𝑎∗ имеет ровно N верных значащих цифр, то

𝛿(𝑎∗) ~ 10−𝑁.

Заметим, что границы абсолютной и относительной погрешностей принято записывать с одной или двумя значащими цифрами. Бо́ льшая точность в записи этих величин не имеет смысла, так как обычно они

являются довольно грубыми оценками истинных значений погрешностей.

Пример 6. Информация о погрешности вида 𝛿(𝑎∗) ≈ 0.288754∙10−5 практически равноценна информации 𝛿(𝑎∗) ≈ 3·10−6, причем последняя вызывает больше доверия. Скорее всего, вполне удовлетворительной в данном случае является запись 𝛿(𝑎∗) ~ 10−6.

Вернемся к неравенству (3). Очевидно, что оно эквивалентно двойному неравенству

𝑎∗ – ∆̅*(*𝑎∗) ≤ *a* ≤ 𝑎∗ + ∆̅*(*𝑎∗)

и поэтому тот факт, что число 𝑎∗ является приближенным значением числа *a* с верхней границей абсолютной погрешности ∆̅*(*𝑎∗) (или с *абсолютной точностью* ε = ∆̅*(*𝑎∗) ) принято записывать в виде

*a* = 𝑎∗ ± ∆̅*(*𝑎∗)*.* Как правило, числа 𝑎∗ и ∆̅*(*𝑎∗) указывают с одинаковым числом цифр после десятичной точки.

Пример 7. Пусть для числа *a* известны приближенное значение

𝑎∗ = 1.648 и граница абсолютной погрешности ∆̅*(*𝑎∗) = 0.002832. Тогда можно записать *a* = 1.648 ± 0.003.

Рассмотрим теперь неравенство (4). Из него следует, что значение *a* заключено примерно между 𝑎∗(1 – 𝛿̅(𝑎∗) и 𝑎∗(1 + 𝛿̅(𝑎∗). Поэтому тот факт, что число 𝑎∗ является приближенным значением числа *a* с границей относительной погрешности 𝛿̅(𝑎∗) (или с *относительной*

*точностью* ε = 𝛿̅(𝑎∗)) принято записывать в виде *a =* 𝑎∗(1 ± 𝛿̅(𝑎∗).

Пример 8. Оценим точность приближения 𝜋∗=3.14 к числу π. Известно, что π = 3.14159…, поэтому π – 𝜋∗=0.00159… Следовательно, можно принять *∆(*𝜋∗) = 0.0016 и 𝛿̅(𝜋∗) ≈ 0.0016 / 3.14 ≈ 0.00051 =

0.051%. Итак, π = 3.14 (1 ± 0.051%).

З а м е ч а н и е. Если число 𝑎∗ приводится в качестве результата без указания величины погрешности, то принято считать, что все его значащие цифры являются верными. Начинающий пользователь ЭВМ часто слишком доверяет выводимым из ЭВМ цифрам, предполагая, что вычислительная машина придерживается того же соглашения. Однако это совсем не так: число может быть выведено с таким количеством

значащих цифр, сколько потребует программист заданием соответствующего формата. Как правило, среди этих цифр только небольшое число первых окажутся верными, а возможно, что верных цифр нет совсем. Анализировать результаты вычислений и определять степень их достоверности совсем непросто. Одна из целей изучения вычислительных методов и состоит в достижении понимания того, что можно и чего нельзя ожидать от результатов, полученных на ЭВМ.

1. **Округление.** Часто возникает необходимость в *округлении* числа *a*, т.е. в замене его другим числом 𝑎∗ с меньшим числом значащих цифр. Возникающая при такой замене погрешность называется *погрешностью округления.*

Существует несколько способов округления числа до *n* значащих цифр. Наиболее простой из них – *усечение* состоит в отбрасывании всех цифр, расположенных справа от *n*-й значащей цифры. Более предпочтительным является *округление по дополнению*. В простейшем случае это правило состоит в следующем. Если первая слева от отбрасываемых цифр меньше 5, то сохраняемые цифры остаются без изменения. Если же она больше либо равна 5, то в младший сохраняемый разряд добавляется единица.

Абсолютная величина погрешности при округлении по дополнению не превышает половины единицы разряда, соответствующего последней оставляемой цифре, а при округлении усечением – единицы того же разряда.

Пример 9. При округлении числа *a* = 1.72631 усечением до трех значащих цифр получится число 𝑎∗=1.72, а при округлении по дополнению число 𝑎∗=1.73.

Границы абсолютной и относительной погрешностей принято округлять в сторону увеличения.

Пример 10. Округление по дополнению до двух значащих цифр

величин

̅∆*(*𝑎∗) = 0.003721 и 𝛿̅(𝑎∗) = 0.0005427 дает значения ∆̅*(*𝑎∗) *=*

0.0038 и 𝛿̅(𝑎∗) = 0.00055.

# § 2.3. Погрешности арифметических операций над приближенными числами

Исследуем влияние погрешностей исходных данных на погрешность результатов арифметических операций. Пусть 𝑎∗ и 𝑏∗ - приближенные значения чисел *a* и *b.* Какова соответствующая им величина неустранимой погрешности результата?

Т е о р е м а 1. *Абсолютная погрешность алгебраической суммы или разности не превосходит суммы абсолютных погрешностей слагаемых, т.е.*

*∆(*𝑎∗ *±* 𝑏∗*) ≤ ∆(*𝑎∗*) + ∆(* 𝑏∗*). (2.8)*

*□ Имеем ∆(*𝑎∗ *±* 𝑏∗*) = |(a ± b) – (*𝑎∗ *±* 𝑏∗)|*= |(a -* 𝑎∗*) ± (b -* 𝑏∗*)| ≤*

*∆(*𝑎∗*) + ∆(* 𝑏∗) *■*

*С л е д с т в и е. В силу неравенства (8) естественно положить*

∆̅ (𝑎∗ ± 𝑏∗) = ∆̅ *(*𝑎∗*) +* ∆̅ *(*𝑏∗*). (2.9)*

Оценим относительную погрешность алгебраической суммы и разности. Т е о р е м а 2. *Пусть a и b ненулевые числа одного знака. Тогда справедливы неравенства*

𝛿 *(*𝑎∗ + 𝑏∗*) ≤* 𝛿𝑚𝑎𝑥 *,* 𝛿 *(*𝑎∗ − 𝑏∗*) ≤ ν* 𝛿𝑚𝑎𝑥 *, (2.10)*

*где* 𝛿𝑚𝑎𝑥

*= max {* 𝛿 *(*𝑎∗*),* 𝛿 *(* 𝑏∗*)}, ν =* | 𝑎+𝑏 |*.*

| 𝑎−𝑏 |

*□* Используя формулу (2.2) и неравенство (2.8) имеем

*| a ± b |* 𝛿*(*𝑎∗ ± 𝑏∗*) =* ∆̅(𝑎∗ ± 𝑏∗) *≤ ∆(*𝑎∗*) + ∆(*𝑏∗*) = |a|* 𝛿 *(*𝑎∗*) + |b|* 𝛿 *(*𝑏∗*)≤ ( | a | + | b |)* 𝛿𝑚𝑎𝑥 *= | a + b |* 𝛿𝑚𝑎𝑥 *.*

Из полученного неравенства следуют оценки (2.10). ■

С л е д с т в и е. *В силу неравенств (2.10) естественно положить*

𝛿̅ (𝑎∗ + 𝑏∗) = 𝛿̅ 𝑚𝑎𝑥*,* 𝛿̅ (𝑎∗ − 𝑏∗) = 𝜈 𝛿̅ 𝑚𝑎𝑥*, (2.11)*

*где* 𝛿

𝑚𝑎𝑥

*= max {* ̅𝛿*(*𝑎∗)*,* 𝛿̅*(*𝑏∗) *}, ν =* | 𝑎+𝑏 |*.*

| 𝑎−𝑏 |

Первое из неравенств (2.11) означает, что при суммировании чисел одного знака не происходит потери точности, если точность оценивать в относительных единицах. Совсем иначе обстоит дело при вычитании чисел одного знака. Здесь граница относительной погрешности возрастает в *ν > 1* раз и возможна существенная потеря точности. Если числа *a* и *b* близки настолько, что | *a + b* | >> *|a – b |,* то *ν >> 1*

и не исключена полная или почти полная потеря точности. Когда это происходит, говорят о том, что произошла *катастрофическая потеря точности.*

Итак, получаем следующий важный вывод. При построении численного метода решения задачи следует избегать вычитания близких чисел одного знака. Если же такое вычитание неизбежно, то следует вычислять аргументы с повышенной точностью, учитывая ее потерю

примерно в *ν =* | 𝑎+𝑏 |

| 𝑎−𝑏 |

раз.

Т е о р е м а 3. *Для относительных погрешностей произведения и частного приближенных чисел верны оценки*

𝛿*(*𝑎∗𝑏∗*) ≤* 𝛿*(*𝑎∗*) +* 𝛿*(*𝑏∗*) +* 𝛿*(*𝑎∗*)* 𝛿*(*𝑏∗*), (2.12)*

𝛿 *(* 𝑎∗ *) ≤* 𝛿(𝑎∗) + 𝛿(𝑏∗)*, (2.13)*

𝑏∗

1− 𝛿(𝑏∗)

в последней из которых считается, что 𝛿 *(* 𝑏∗*) < 1.*

*□* Выполним следующие преобразования:

*|ab|* 𝛿*(*𝑎∗𝑏∗*) = ∆(*𝑎∗ 𝑏∗*) = |ab* − 𝑎∗ 𝑏∗*| = |(a* − 𝑎∗) 𝑏 *+(b* − 𝑏∗*)a –*

*(a* − 𝑎∗)(𝑏 − 𝑏∗)*| ≤ |b| ∆(*𝑎∗*) + |a| ∆(*𝑏∗*) + ∆(*𝑎∗*) ∆(*𝑎∗) = |𝑎𝑏|*(*𝛿*(*𝑎∗*) +*

𝛿*(*𝑏∗*) +* 𝛿*(*𝑎∗*)*𝛿*(*𝑏∗*)),*

т.е. *|ab|* 𝛿*(*𝑎∗𝑏∗*) ≤* |𝑎𝑏|*(*𝛿*(*𝑎∗*) +* 𝛿*(*𝑏∗*) +* 𝛿*(*𝑎∗*)*𝛿*(*𝑏∗*)).*

Разделив обе части этого неравенства на *|ab|,* получаем оценку (2.12). Для вывода второй оценки предварительно заметим, что *|* 𝑏∗| =

|𝑏 + (𝑏∗ − 𝑏)| ≥ |𝑏| − *∆(* 𝑏∗*) = |b| (1* − 𝛿*(*𝑏∗*)).* Тогда

𝑎 𝑎∗

𝛿*(* 𝑎∗ *) =* |𝑏 − 𝑏∗ |

|𝑎𝑏∗−𝑏 𝑎∗| *=* |𝑎(𝑏∗ −𝑏 )+𝑏 (𝑎−𝑎∗)|

*≤* |𝑎|∆( 𝑏∗)+|𝑏|∆( 𝑎∗) *=*

𝑏∗

𝑎 *=*

|𝑏|

|𝑎𝑏∗|

|𝑎𝑏∗|

|𝑎𝑏|(1− 𝛿(𝑏∗))

𝛿(𝑎∗)+ 𝛿(𝑏∗) . ■

1− 𝛿(𝑏∗)

*С л е д с т в и е. Если* 𝛿*(*𝑎∗*) << 1* и 𝛿*(*𝑏∗*) << 1, то для оценки границ относительных погрешностей можно использовать следующие приближенные равенства:*

̅𝛿*(*𝑎∗𝑏∗) *≈* 𝛿̅*(*𝑎∗) + 𝛿̅*(*𝑏∗)*,* 𝛿̅*(*𝑎∗/𝑏∗) *≈* 𝛿̅*(*𝑎∗) + 𝛿̅*(*𝑏∗)*, (2.14)*

Именно равенства (2.14) чаще всего и используют для практической оценки погрешности.

Итак, выполнение арифметических операций над приближенными числами, как правило, сопровождается потерей точности. Единственная операция, при которой потеря не происходит, это сложение чисел одного знака. Наибольшая потеря точности может произойти при вычитании близких чисел одного знака.

# § 2.4. Погрешность функции

* 1. **Погрешность функции многих переменных.** Пусть *f (x) = f(*𝑥1*,* 𝑥2*,…,*𝑥𝑚*)* – дифференцируемая в области *G* функция *m* переменных, вычисление которой производится при приближенно

заданных аргументах 𝑥∗, 𝑥∗, … , 𝑥∗ . Такая ситуация возникает,

1 2 𝑚

например, всякий раз, когда на ЭВМ производится расчет по формуле. Важно знать, какова величина неустранимой ошибки, вызванной тем, что вместо значения *y = f( x )* в действительности вычисляется значение

𝑦∗*= f(* 𝑥∗*), где* 𝑥∗*=(*𝑥∗, 𝑥∗, … , 𝑥∗ ).

1 2 𝑚

Введем обозначения: пусть [*x*, 𝑥∗] – отрезок, соединяющий точки *x*

и 𝑥∗, и 𝑓′ = ∂*f* / ∂𝑥 .

𝑥𝑗 𝑗

Т е о р е м а 4. Для абсолютной погрешности значения 𝑦∗*= f(*𝑥∗)

*справедлива следующая оценка:*

*∆(*𝑦∗*)≤* ∑𝑚 max |𝑓′ | ∆(𝑥∗)*.* (2.15)

𝑗=1 [𝑥,𝑥∗]

𝑥𝑗 𝑗

□ Оценка (2.15) вытекает из формулы конечных приращений Лагранжа:

*f(x) – f(*𝑥∗*) =* ∑𝑚

𝑓′ (𝑥̃)( 𝑥

− 𝑥∗)*,* 𝑥̃*ϵ[x,* 𝑥∗].

𝑗=1

𝑥𝑗

𝑗 𝑗

Далее берем модуль от правой и левой частей уравнения и правую часть заменяем на максимум. Получаем требуемое соотношение. ■

С л е д с т в и е. *Если* 𝑥∗ *≈ x, то в силу оценки (2.15) можно положить*

∆̅ *(*𝑦∗*) ≈* ∑𝑚 |𝑓′ ( 𝑥∗ )| ∆̅(𝑥∗*),* (2.16)

𝑗=1 𝑥𝑗 𝑗

∆̅ *(*𝑦∗*) ≈* ∑𝑚 |𝑓′ (𝑥)| ∆̅(𝑥∗*),* (2.17)

𝑗=1 𝑥𝑗 𝑗

Равенство (2.16) удобно для практических оценок, а равенством (2.17) мы воспользуемся в дальнейшем для теоретических построений.

Из формул (2.16) и (2.17) вытекают приближенные равенства для оценки границ относительных погрешностей:

𝛿̅ (𝑦∗) ≈ ∑𝑚

𝜈∗ 𝛿̅ ( 𝑥∗), 𝛿̅ (𝑦∗) ≈ ∑𝑚

𝜈𝑗 𝛿̅ ( 𝑥∗). (2.18)

Здесь

𝑗=1 𝑗 𝑗

𝑗=1 𝑗

|𝑥∗|∙|𝑓𝘍 (𝑥∗ )| | 𝑥 |∙|𝑓𝘍 (𝑥 )|

𝜈∗ =

𝑗 𝑥𝑗 , 𝜈 =

𝑗 𝑥𝑗

, (2.19)

𝑗 |𝑓( 𝑥∗)| 𝑗 |𝑓( 𝑥)|

* 1. **Погрешность функции одной переменной.** Формулы для границ погрешностей функции одной переменной являются частным случаем формул (2.16) – (2.18) при *m* = 1:

∆̅ *(*𝑦∗*) ≈ |* 𝑓′*(*𝑥∗)| ∆̅ *(* 𝑥∗*),* ∆̅ *(*𝑦∗*) ≈ |* 𝑓′*(*𝑥)| ∆̅ *(* 𝑥∗*),* (2.20)

𝛿̅*(*𝑦∗*) ≈* 𝜈∗ 𝛿̅*(* 𝑥∗*),* 𝛿̅*(*𝑦∗*) ≈* 𝜈 𝛿̅*(* 𝑥∗*),* (2.21) где 𝜈∗ = | 𝑥∗| |𝑓′ (𝑥∗*)* | / | 𝑓 (𝑥∗)| *, ν = |x|* |𝑓′ (𝑥)| / | 𝑓 (𝑥)|.

# Лекция 3

**Тема 3. Вычислительные задачи, методы и алгоритмы**

# § 3.1. Корректность вычислительной задачи

Будем считать, что постановка вычислительной задачи включает в себя задание *множества допустимых входных данных X и множества возможных решений Y*. Цель вычислительной задачи состоит в нахождении решения *yϵY* по заданному входному данному *xϵX*. Для простоты ограничимся рассмотрением задач, в которых входные данные и решения могут быть только числами, наборами чисел (векторами, матрицами, последовательностями) и функциями. Предположим, что для оценки величин погрешностей приближенных входных данных 𝑥∗ и приближенного решения 𝑦∗ введены абсолютные и относительные погрешности ∆(𝑥∗), ∆(𝑦∗), δ(𝑥∗), δ(𝑦∗), а также

их границы ∆̅(𝑥∗), ∆̅(𝑥∗), 𝛿̅(𝑥∗), 𝛿̅(𝑦∗). Определения этих величин в

случае, когда *x* и *y –* числа, были даны выше. В тех случаях, когда *x* и *y* не являются числами, эти характеристики погрешностей также можно ввести естественным образом, например, так, как указывалось в первом разделе курса.

Перейдем теперь к определению понятия корректности вычислительной задачи, которое было впервые сформулировано французским математиком Ж.Адамаром и развито затем российским математиком И.Г.Петровским. Вычислительная задача называется *корректной,* если выполнены следующие три требования:

1). Ее решение *y ϵ Y* существует при любых входных данных *x ϵ X*; 2). Это решение единственно;

3). Решение устойчиво по отношению к малым возмущениям входных данных.

В том случае, когда хотя бы одно из этих требований не выполнено, задача называется *некорректной*.

**Существование решения** вычислительной задачи – естественное требование к ней. Отсутствие решения может свидетельствовать, например, о непригодности принятой математической модели либо о неправильной постановке задачи. Иногда отсутствие решения является следствием неправильного выбора множества допустимых входных данных *X* или множества возможных решений *Y.*

Пример 3.1. Рассмотрим задачу о решении квадратного уравнения

𝑥2 *+ bx +c = 0.* (3.1)

Старший коэффициент *а = 1.* Если *считать входным данным пару коэффициентов b, c* и искать решение в множестве вещественных чисел, то существование решений

𝑥1 *= (*− *b* − √𝑏2 − 4𝑐 *)/2,* 𝑥2 *= (*− *b +* √𝑏2 − 4𝑐 *)/2* (3.2)

будет гарантировано только в том случае, если ограничить множество входных данных коэффициентами, удовлетворяющими условию

𝑏2 − 4𝑐 ≥ 0. Если же расширить множество возможных решений и считать, что корни (3.2) могут принимать комплексные значения, то задача будет иметь решение при любых *b, c.*

***Единственность*** решения для некоторых задач является естественным свойством; для других задач решение может не быть единственным. Например, квадратное уравнение (3.1) имеет два корня (3.2). Как правило, если задача имеет реальное содержание, то неединственность может быть ликвидирована введением дополнительных ограничений на решение (т.е. сужением множества *Y*). В некоторых случаях проблема снимается тем, что признается целесообразным найти набор всех решений, отвечающих входным данным, и тогда за решение принимается этот набор. Например, для квадратного уравнения (3.1) решением можно назвать пару (𝑥1 , 𝑥2).

Решение *y* вычислительной задачи называется ***устойчивым*** *по входным данным x,* если оно зависит от входных данных непрерывным образом. Это означает, что для любого ε > 0 существует 𝛿 = 𝛿 (ε) > 0 такое, что всякому исходному данному 𝑥∗, удовлетворяющему условию

∆(𝑥∗) < 𝛿, отвечает приближенное решение 𝑦∗, для которого ∆(𝑦∗) < ε. Таким образом, для устойчивой вычислительной задачи ее решение теоретически можно найти со сколь угодно высокой точностью ε, если обеспечена достаточно высокая точность 𝛿 входных данных.

Неустойчивость решения *y* означает, что существует такое 𝜀0 > 0, что какое было малое 𝛿 > 0 ни было задано, найдутся такие исходные данные 𝑥∗, что ∆(𝑥∗) < 𝛿, но ∆(𝑦∗) *≥* 𝜀0.

Приведем несколько примеров устойчивых и неустойчивых задач.

Пример 3.2. Задача вычисления корней квадратного уравнения (3.1) устойчива, т.к. корни (3.2) являются непрерывными функциями коэффициентов *b* и *c*.

Пример 3.3. Задача о вычислении ранга матрицы в общем случае

неустойчива. В самом деле, для матрицы А = 1 0

0 0

ранг равен 1,

поскольку detA = 0 и существует ненулевой элемент 𝑎11=1. Однако сколь угодно малое возмущение коэффициента 𝑎22 на величину ε ≠ 0

приводит к матрице А = 1 0

𝜀

0 𝜀

, для которой det𝐴𝜀 = ε ≠ 0 и

следовательно ранг равен 2.

Пример 3.4. Покажем, что задача вычисления определенного

интеграла I = ∫𝑏 𝑓(𝑥)𝑑𝑥

𝑎

устойчива.

Пусть 𝑓∗(x) – приближенно заданная интегрируемая функция и

𝐼∗ = ∫𝑏 𝑓∗(x) 𝑑𝑥. Определим абсолютную погрешность функции 𝑓∗ с помощью равенства ∆(𝑓∗) = max𝑥𝗀[𝑎,𝑏] |𝑓(𝑥) − 𝑓∗*(x*)| . Так как

𝑎

*∆(*𝐼∗) = | *I -* 𝐼∗| = | ∫𝑏(𝑓(𝑥) − 𝑓∗(𝑥))𝑑𝑥 | ≤ *(b – a) ∆(*𝑓∗)*,* (3.3) то для любого ε > 0 неравенство ∆ (𝐼∗) < ε будет выполнено, если потребовать выполнение условия *∆(*𝑓∗) *<* 𝛿 *= ε / (b – a).*

𝑎

Пример 3.5. Покажем, что задача вычисления производной *u(x) =*

𝑓′*(x)* приближенно заданной функции является неустойчивой.

Пусть 𝑓∗(x) – приближенно заданная на отрезке [*a, b*] непрерывно дифференцируемая функция и 𝑢∗(*x*) = ( 𝑓∗)' *(x*). Определим абсолютные погрешности с помощью равенств

*∆(*𝑓∗) *=* max𝑥𝗀[𝑎,𝑏] |𝑓(𝑥) − 𝑓∗(*x*) | и

*∆(*𝑢∗) *=* max𝑥𝗀[𝑎,𝑏] |𝑢(𝑥) − 𝑢∗(*x*)|.

(3.4)

Возьмем, например, 𝑓∗(*x*) = 𝑓(𝑥) + α sin (*x* / 𝛼2), где 0 < α << 1.

Тогда 𝑢∗(*x*) = 𝑢(𝑥) + 𝛼−1𝐶𝑜𝑠 (𝑥/𝛼2 ) и ∆( 𝑢∗) = 𝛼−1, в то время как

*∆(*𝑓∗) *= α.* Таким образом, сколь угодно малой погрешности задания функции *f(x)* может отвечать сколь угодно большая погрешность

производной 𝑓′*(x),* т.е. задача дифференцирования является неустойчивой.

Часто требование малости абсолютной погрешности является неоправданным или трудно проверяемым. В таких случаях полезно рассмотреть *относительную устойчивость* решения, определение которой отличается от данного выше определения абсолютной устойчивости только тем, что ∆(𝑥∗) и ∆(𝑦∗) заменяются на δ(𝑥∗) и δ(𝑦∗) соответственно.

# § 3.2. Обусловленность вычислительной задачи

**Определения.** Пусть вычислительная задача корректна (ее решение существует, единственно и устойчиво по входным данным). Теоретически наличие у задачи устойчивости означает, что ее решение может быть найдено со сколь угодно малой погрешностью, если только гарантировать, что погрешности входных данных достаточно малы. Однако на практике погрешности входных данных не могут быть сделаны сколь угодно малыми, точность их ограничена. Даже то, что входные данные нужно ввести в ЭВМ, означает, что их точность будет заведомо ограничена конечной величиной разрядной сетки. В реальности уровень ошибок в исходной информации будет заведомо выше.

Для ответа на вопрос, как повлияют малые, но конечные погрешности входных данных на решение, введем новые понятия.

Под *обусловленностью вычислительной задачи* понимают чувствительность ее решения к малым погрешностям исходных данных. Задачу называют *хорошо обусловленной*, если малым погрешностям исходных данных отвечают малые погрешности решения, и *плохо обусловленной*, если возможны сильные изменения

решения.

Часто оказывается возможным ввести количественную меру степени обусловленности вычислительной задачи – *число обусловленности.* Эту величину можно интерпретировать как коэффициент возможного возрастания погрешностей в решении по отношению к вызвавшим их погрешностям входных данных.

Пусть между абсолютными погрешностями входных данных *x* и решения *y* установлено неравенство

∆( 𝑦∗) ≤ 𝜈∆ ∆(𝑥∗). (3.5)

Тогда величина 𝜈∆ называется *абсолютным числом обусловленности*. Если же установлено неравенство

𝛿( 𝑦∗) ≤ 𝜈𝛿 𝛿(𝑥∗) (3.6)

между относительными ошибками входных данных и решения, то величину 𝜈𝛿 называют *относительным числом обусловленности.* В неравенствах (3.5), (3.6) вместо погрешностей ∆(𝑥∗), ∆(𝑦∗), δ(𝑥∗),

δ(𝑦∗), могут фигурировать их границы ∆̅(𝑥∗), ∆̅(𝑥∗), 𝛿̅(𝑥∗), 𝛿̅(𝑦∗).

Обычно под числом обусловленности понимают одну из величин 𝜈∆ или

𝜈𝛿, причем выбор бывает ясен из смысла задачи. Чаще все же под числом обусловленности понимают относительное число обусловленности. Для плохо обусловленной задачи ν>>1. В некотором смысле неустойчивость задачи – это крайнее проявление плохой обусловленности, отвечающее значению ν = ∞. Конечно, ν – это максимальный коэффициент возможного возрастания уровня ошибок, и для конкретных входных данных действительный коэффициент возрастания может оказаться существенно меньше. Однако значение ν>>1 все же свидетельствует о реальной возможности существенного роста ошибок. Грубо говоря, если 𝜈𝛿 ≈ 10𝑁, то порядок N показывает число верных цифр, которое может быть утеряно в результате по сравнению с числом верных цифр входных данных.

Каково то значение ν, при котором следует признать задачу плохо обусловленной? Ответ на этот вопрос существенно зависит, с одной стороны, от предъявляемых требований к точности решения и, с другой стороны, от уровня обеспечиваемой точности входных данных. Например, если требуется найти решение с точностью 0.1%, а входная информация задается с точностью 0.02%, то уже значение ν=10 свидетельствует о плохой обусловленности. Однако (при тех же требованиях к точности результата) гарантия, что входные данные задаются с точностью не ниже 0.0001%, означает, что и при ν = 103 задача хорошо обусловлена.

Важным примером плохо обусловленной задачи является вычитание приближенных чисел одного знака. Для нее относительное число обусловленности ν= |a + b| / |a – b|.

# Приведем примеры хорошо и плохо обусловленных задач. 1.Обусловленность задачи вычисления функции одной

**переменной.** Пусть задача состоит в вычислении по заданному *x*

значения *y = f(x)* дифференцируемой функции *f(x).*

В силу формул (2.21) и (2.22) для этой задачи имеем

𝜈∆ ≈ |𝑓′*(x)|,* (3.7)

𝜈𝛿

≈ |𝑥||𝑓𝘍(𝑥)|. (3.8)

|𝑓(𝑥)

Воспользуемся этими формулами для оценки обусловленности задачи вычисления некоторых простейших функций.

Пример 3.6. Для задачи вычисления значения функции y = 𝑒𝑥 в силу формулы (3.8) относительное число обусловленности 𝜈𝛿=|*x*| и при реальных вычислениях эта величина не может быть очень большой. Например, при вычислении экспоненты на компьютере типа IBM PC всегда |*x*| < 88, так как в противном случае возможно переполнение или антипереполнение. Следовательно, задача вычисления значения этой функции хорошо обусловлена, однако в случае 10 < |*x*| < 102 следует ожидать потери 1 – 2 верных значащих цифр по сравнению с числом верных цифр аргумента *x.* Подчеркнем, что эта потеря точности объективно обусловлена погрешностью задания аргумента и не связана с используемым алгоритмом.

Пример 3.7. Для задачи вычисления значения функции *y = sin x* в силу формулы (3.7) имеем 𝜈∆ = | cos 𝑥 | ≤ 1, что говорит о хорошей абсолютной обусловленности этой задачи при всех *x.* Однако если важен результат с определенным числом верных знаков, то нужно исследовать относительную обусловленность. Согласно формуле (3.8) имеем

𝜈𝛿 = | *x ctg x|.*

Очевидно, что 𝜈𝛿 → ∞ при *x → πк* (для *к = ±*1, ±2, ±3, ...), т.е. при

*x ≈ πк* задача обладает плохой относительной обусловленностью.

**2.Обусловленность задачи вычисления интеграла** ∫𝑏 𝑓(𝑥)𝑑𝑥. Как следует из оценки (3.3) в этом случае абсолютное число обусловленности имеет вид 𝜈∆ = (*b – a).* Если же перейти к рассмотрению относительных погрешностей и положить 𝛿(𝑓∗) =

𝑎

max𝑎≤𝑥≤𝑏 | 𝑓∗(𝑥) − 𝑓(𝑥)|/|𝑓(𝑥)| для тех *x,* где *f(x) ≠ 0*, то используя неравенство

∆( 𝐼∗) ≤ ∫𝑏 | 𝑓∗(𝑥) − 𝑓(𝑥)|𝑑𝑥 ≤ ∫𝑏 | 𝑓(𝑥)|𝑑𝑥∙ 𝛿(𝑓∗), получим оценку

𝑎 𝑎

𝛿(𝐼∗) ≤ 𝜈𝛿 𝛿(𝑓∗), (3.9)

в которой 𝜈𝛿≤ ∫𝑏 | 𝑓(𝑥)|𝑑𝑥/ |∫𝑏 𝑓(𝑥) 𝑑𝑥|.

𝑎 𝑎

Если подынтегральная функция знакопостоянна, то 𝜈𝛿 = 1 и задача хорошо обусловлена. Если же функция *f* на *[a, b]* принимает значения разных знаков, то 𝜈𝛿 > 1. Для сильно осциллирующих (колеблющихся) около нуля функций может оказаться, что 𝜈𝛿 >> 1 и тогда задача вычисления интеграла является плохо обусловленной.

# § 3.3. Вычислительные методы

Обсудив некоторые важные особенности вычислительных задач, обратим внимание на методы, которые используются в вычислительной математике для преобразования задач к виду, удобному для реализации на ЭВМ, и которые позволяют конструировать вычислительные алгоритмы. Эти методы конкретизируют тот общий метод вычислительной математики, о котором шла речь во введении. Будем называть эти методы *вычислительными.*

С некоторой степенью условности можно разбить вычислительные методы на следующие классы:

1. **Методы эквивалентных преобразований.** Эти методы позволяют заменить исходную задачу другой, имеющей то же решение. Выполнение эквивалентных преобразований оказывается полезным, если новая задача проще исходной или для нее существует известный метод решения или готовая программа. Эти методы мы будем рассматривать в дальнейшем при решении конкретных задач, например, нелинейных уравнений.
2. **Методы аппроксимации.** Эти методы позволяют аппроксимировать исходную задачу другой, решение которой в определенном смысле близко к решению исходной задачи. Погрешность, возникающая при такой замене, называется

*погрешностью аппроксимации.* Принято говорить, что метод сходится, если погрешность аппроксимации стремится к нулю при стремлении параметров метода к некоторому предельному значению. Методы аппроксимации будем применять, например, при численном интегрировании.

1. **Итерационные методы.** Это специальные методы построения последовательных приближений к решению задачи. Применение итерационного метода начинают с задания начальных приближений. Для получения каждого из последующих приближений выполняют однотипный набор действий с использований найденных ранее приближений – *итерацию.* Неограниченное продолжение этого *итерационного процесса* позволяет построить бесконечную последовательность приближений к решению – *итерационную последовательность.* Если эта последовательность сходится к решению задачи, то говорят, что *итерационный метод сходится.* Множество начальных приближений, для которых метод сходится, называется *областью сходимости метода.* Итерационные методы широко используются при решении самых разнообразных задач с применением ЭВМ. Мы будем их подробно рассматривать, например, при решении нелинейных уравнений.

Практическая реализация итерационных методов всегда связана с необходимостью выбора *критерия окончания итерационного процесса.* Для формирования критерия окончания итерационного процесса по достижению заданной точности используют *апостериорные оценки погрешности* – неравенства, в которых величина погрешности оценивается через величины, получаемые в ходе вычислительного процесса.

# § 3.4. Корректность вычислительных алгоритмов

Вычислительный метод, доведенный до степени детализации, позволяющей реализовать его на ЭВМ, принимает форму вычислительного алгоритма.

Определим вычислительный алгоритм как точное предписание действий над входными данными, задающее вычислительный процесс, направленный на преобразование произвольных входных данных в полностью определяемый этими входными данными результат.

К вычислительным алгоритмам предъявляется ряд весьма жестких требований. Первое из них – корректность алгоритма. Будем называть вычислительный алгоритм *корректным*, если выполнены три условия:

1. Он позволяет после выполнения конечного числа элементарных для ЭВМ операций преобразовать любое входное данное *x є X* в результат *y;*
2. результат *y* устойчив по отношению к малым возмущениям входных данных (это требование аналогично требованию устойчивости вычислительной задачи) при отсутствии вычислительной погрешности;
3. результат *y* обладает вычислительной устойчивостью. При вводе в ЭВМ входных данных и в процессе вычислений неизбежно появление вычислительной погрешности, величина которой определяется машинной точностью 𝜀м=2−𝑡 , где *t* – разрядность представления мантиссы. Назовем алгоритм *вычислительно устойчивым*, если вычислительная погрешность результата стремится к нулю при 𝜀м→0.

Если хотя бы одно из перечисленных условий не выполняется, то будем называть алгоритм *некорректным.*

# § 3.5. Обусловленность вычислительных алгоритмов

По аналогии с понятием обусловленности вычислительной задачи можно ввести понятие обусловленности вычислительного алгоритма, отражающее чувствительность результата работы алгоритма к малым, но неизбежным ошибкам округления. Вычислительно устойчивый алгоритм называют *хорошо обусловленным*, если малые относительные погрешности округления (характеризуемые машинной точностью 𝜀м) приводят к малой относительной вычислительной погрешности δ(𝑦∗) результата 𝑦∗, и *плохо обусловленным*, если вычислительная погрешность может быть недопустимо большой.

Если δ(𝑦∗) и 𝜀м связаны неравенством δ(𝑦∗) ≤ 𝜈А𝜀м, то число 𝜈А следует называть *числом обусловленности вычислительного алгоритма*. Для плохо обусловленных алгоритмов 𝜈А>>1.

# Лекция 4

**Тема 4. Методы решения нелинейных уравнений**

Рассмотрим задачу отыскания корней нелинейных уравнений и основные методы ее решения. Решение нелинейных уравнений представляет собой редкий пример задачи, которая может быть сравнительно полно исследована элементарными средствами. В то же время многие проблемы, возникающие при отыскании корней нелинейных уравнений, типичны, а некоторые методы их решения (в особенности метод Ньютона и метод простой итерации) допускают широкие обобщения и играют в вычислительной математике фундаментальную роль.

# § 4.1. Постановка задачи. Основные этапы решения

Задача отыскания корней нелинейного уравнения с одним неизвестным вида

*f* (*x*)*= 0* (4.1)

имеет многовековую историю, но не потеряла свою актуальность и в наши дни, так как она часто возникает как элементарный шаг при решении различных научных и технических проблем.

Напомним, что *корнем* (или *решением*) уравнения (4.1) называется значение 𝑥̅, при котором *f* (𝑥̅) *= 0.*

Будем далее предполагать, что в окрестности каждого из искомых корней функция *f(x)* дважды непрерывно дифференцируема.

Корень 𝑥̅ уравнения (4.1) называется *простым,* если 𝑓′(𝑥̅) ≠ 0. В противном случае (т.е. в случае 𝑓′(𝑥̅) = 0) корень 𝑥̅ называется *кратным.* Число *m* назовем кратностью корня 𝑥̅, если 𝑓(𝑘)(𝑥̅) = 0 для *k = 1, 2,…, m-1* и 𝑓(𝑚)(𝑥̅) ≠ 0.

Геометрически корень 𝑥̅ соответствует точке пересечения графика функции *y = f (x)* с осью *Ox.* Корень 𝑥̅ является простым, если график пересекает ось *Ox* под ненулевым углом, и кратным, если пересечение происходит под нулевым углом. Задача отыскания простых корней является значительно более простой (и чаще встречающейся), чем задача отыскания кратных корней. Большинство методов решения уравнения (4.1) ориентировано именно на вычисление простых корней.

2

В подавляющем большинстве случаев представить решение уравнения (4.1) в виде конечной формулы оказывается невозможным. Даже для простейшего алгебраического уравнения *n*-й степени

𝑎𝑛𝑥𝑛 + 𝑎𝑛−1𝑥𝑛−1 +… + 𝑎1𝑥 + 𝑎0 = 0 (4.2) явные формулы найдены лишь при *n* = 2, 3, 4. Однако уже для уравнений пятой и более высоких степеней таких формул не существует (теорема Абеля). При этом нужно иметь в виду, что даже точное решение, если оно существует, оказывается выраженным достаточно громоздкой формулой, вычисление по которой все равно дает приближенное значение, так как содержит вычислительную погрешность. Поэтому в дальнейшем мы откажемся от попыток найти точное решение и сосредоточимся на методах приближенного вычисления корней с заданной точностью ε.

Решение задачи вычисления корней нелинейного уравнения, как правило, осуществляется в два этапа. Первый этап называется *этапом локализации (*или *отделения)* корней, второй – *этапом итерационного уточнения корней.*

Л о к а л и з а ц и я к о р н е й. Отрезок [*a, b*], содержащий только один корень 𝑥̅ уравнения (4.1), называют отрезком *локализации* корня

𝑥̅. Цель этапа локализации считается достигнутой, если для каждого из подлежащих локализации корней удалось указать отрезок локализации (его длину стараются по возможности сделать минимальной).

Способы локализации корней многообразны, и указать универсальный метод не представляется возможным. Иногда отрезок локализации известен заранее или он определяется из физических соображений. В простых случаях хороший результат может дать графический метод. Широко применяют также построение таблиц значений функции *f* вида 𝑦𝑖 = *f* (𝑥𝑖 *), i = 1, 2, …, n.* При этом способе локализации о наличии на отрезке [𝑥𝑖−1,𝑥𝑖 ] корня судят по перемене знака функции на концах отрезка. Основанием для применения указанного способа служит известная теорема математического анализа: Т е о р е м а 4.1. *Пусть функция f непрерывна на отрезке [a, b] и принимает на его концах значения разных знаков, т.е. f(a)· f(b) < 0.*

*Тогда отрезок [a, b] содержит по крайней мере один корень уравнения f(x) = 0.*

3

И т е р а ц и о н н о е у т о ч н е н и е к о р н е й. На этом этапе решения задачи для вычисления каждого из корней с точностью ε > 0 используют тот или иной итерационный метод, позволяющий построить последовательность 𝑥(0), 𝑥(1), … , 𝑥(𝑛), … приближений к корню 𝑥̅.

Общие представления об итерационных методах и основные определения были даны в разделе 3. Введем дополнительно некоторые определения.

Итерационный метод называется *одношаговым*, если для вычисления очередного приближения 𝑥(𝑛+1) используется только одно предыдущее приближение 𝑥(𝑛) и *k-шаговым*, если для вычисления

𝑥(𝑛+1) используются *k* предыдущих приближений 𝑥(𝑛−𝑘+1),

𝑥(𝑛−𝑘+2), … , 𝑥(𝑛). Заметим, что для построения итерационной последовательности одношаговым методом требуется задание только одного начального приближения, в то время как при использовании *k*- шагового метода – *k* начальных приближений 𝑥(0), 𝑥(1), … , 𝑥(𝑘−1).

Скорость сходимости – одна из важнейших характеристик итерационных методов. Говорят, что *метод сходится со скоростью геометрической прогрессии,* знаменатель которой *q* < 1, если для всех *n* справедлива следующая оценка:

|𝑥(𝑛) − 𝑥̅| ≤ 𝑐0𝑞𝑛 (4.5)

Как нетрудно видеть, из оценки (4.5) действительно вытекает сходимость метода.

Пусть одношаговый итерационный метод обладает следующим свойством: существует ϭ-окрестность корня 𝑥̅ такая, что если приближение 𝑥(𝑛) принадлежит этой окрестности, то справедлива оценка

|𝑥(𝑛+1) − 𝑥̅| ≤ С |𝑥(𝑛) − 𝑥̅| 𝑝, (4.6) где C > 0 и *p* ≥ 1 – постоянные. Если *p* = 1 и C < 1, то говорят, что метод обладает *линейной скоростью сходимости* в указанной ϭ-окрестности корня. Если *p > 1,* то принято говорить о *сверхлинейной скорости сходимости.* При *p =* 2 скорость сходимости называют *квадратичной,* а при *p =* 3 – *кубической*.

**Л е м м а 4.1.** *Пусть одношаговый итерационный метод обладает линейной скоростью сходимости в некоторой ϭ-окрестности корня* 𝑥̅*. Тогда при любом выборе начального приближения* 𝑥(0) *из*

4

*ϭ-окрестности корня* 𝑥̅ *итерационная последовательность* 𝑥(𝑛) *не выходит за пределы этой окрестности, метод сходится со скоростью геометрической прогрессии со знаменателем q = C и имеет место следующая оценка погрешности:*

|𝑥(𝑛) − 𝑥̅| ≤ 𝑞𝑛 | 𝑥(0) − 𝑥̅ |, *n* ≥ 0. (4.7)

□ Заметим, что принадлежность 𝑥(𝑛) окрестности ( 𝑥̅ − *ϭ,* 𝑥̅ *+ ϭ* ) является следствием неравенства (4.7). В самом деле, так как *q* < 1, то

|𝑥(𝑛) − 𝑥̅| ≤ | 𝑥(0) − 𝑥̅ | < *ϭ.* Сходимость 𝑥(𝑛) к 𝑥̅ также вытекает из (4.7).

Справедливость (4.7) установим методом индукции. При *n* = 0 оно переходит в очевидное: | 𝑥(0) − 𝑥̅ | ≤ | 𝑥(0) − 𝑥̅ |. Пусть неравенство (4.7) выполнено при *n = m –* 1*.* Тогда

| 𝑥(𝑚) − 𝑥̅ | ≤ *q* | 𝑥𝑚−1) − 𝑥̅ | ≤ 𝑞𝑚 | 𝑥(0) − 𝑥̅ |, т.е. неравенство выполнено при *n = m.* ■

Можно доказать аналогичное утверждение и для многошагового итерационного метода. Мы этого делать не будем.

С помощью указанной леммы исследование сходимости итерационных методов сводится только к получению оценки (4.6).

# § 4.2. Обусловленность задачи вычисления корня

Пусть 𝑥̅ – корень уравнения (4.1), подлежащий определению. Будем считать, что входными данными для задачи вычисления корня 𝑥̅ являются значения функции *f(x)* в малой окрестности корня. Так как значения *f(x)* будут вычисляться на ЭВМ по некоторой программе, то в действительности задаваемые значения являются приближенными и их следует обозначать через 𝑓∗(*x*). Ошибки в 𝑓∗(*x*) могут быть связаны не только с неизбежными ошибками округления, но и с использованием приближенных методов. К сожалению, нельзя ожидать, что в окрестности корня относительная погрешность 𝛿(𝑓∗) окажется малой. Достаточно обратиться к примеру 3.7, чтобы установить, что для достаточно простой функции *y = sin x* в окрестности корней *x = π·к, к = 0, ±1, ±2, …* значения этой функции не могут быть найдены с малой относительной погрешностью. Реально рассчитывать можно лишь на то, что малой окажется абсолютная погрешность вычисления значений функции.

5

Будем предполагать, что в достаточно малой окрестности корня выполняется неравенство | *f*(*x*) – 𝑓∗(*x*)| < ∆̅, где ∆̅ = ∆̅(𝑓∗) – граница абсолютной погрешности.

Если функция *f* непрерывна, то найдется такая малая окрестность (𝑥̅ – 𝖼̅ , 𝑥̅ + 𝖼̅ ) корня 𝑥̅ , имеющая радиус 𝖼̅ > 0, в которой выполняется неравенство

| *f( x)* | < ∆̅. (4.9)

Для 𝑥 ϵ (𝑥̅ – 𝖼̅ , 𝑥̅ + 𝖼̅ ) знак вычисленного значения 𝑓∗(*x*), вообще говоря, не обязан совпадать со знаком *f( x)* и, следовательно, становится невозможным определить, какое именно значение *x* из интервала (𝑥̅ – 𝖼̅ , 𝑥̅ + 𝖼̅ ) обращает функцию *f(x)* в нуль. Будем называть этот интервал интервалом неопределенности корня 𝑥̅.

Найдем оценку величины 𝖼̅. Пусть корень 𝑥̅ – простой. Для близких к 𝑥̅ значений *x* справедливо приближенное равенство

*f( x) ≈ f(*𝑥̅) *+* 𝑓′*(*𝑥̅)*(x* – 𝑥̅) = 𝑓′*(*𝑥̅)*(x* – 𝑥̅)

Поэтому неравенство (4.9) примет вид |𝑓′*(*𝑥̅)*(x* – 𝑥̅)| < ∆̅, откуда получаем

𝑥̅ – ∆̅

𝘍

|𝑓 (𝑥̅)|

< *x* < 𝑥̅ + ∆̅ .

|𝑓 (𝑥̅)|

𝘍

Следовательно,

𝖼̅ *≈* 𝜈∆ ∆̅(𝑓∗), (4.10)

1

здесь 𝜈∆ =

|𝑓𝘍(𝑥̅)|

– число, которое в рассматриваемой задаче играет

роль абсолютного числа обусловленности. Действительно, если 𝑥̅ \* – корень уравнения 𝑓∗(*x*) = 0, то *|f(*𝑥̅ *\*)| <* ∆̅ и тогда выполнено неравенство

| 𝑥̅ – 𝑥̅ \*| ≤ ∆̅(𝑥̅ \*) ≤ 𝖼̅ *≈* 𝜈∆ ∆̅(𝑓∗). (4.11)

Заметим, что радиус интервала неопределенности прямо пропорционален погрешности ∆̅ вычисления значения *f.* Кроме того, 𝖼̅ возрастает (обусловленность задачи ухудшается) с уменьшением

6

|𝑓′*(*𝑥̅)|*,* т.е. с уменьшением модуля тангенса угла наклона, под которым график функции пересекает ось *Ох.*

Если же 𝑓′*(*𝑥̅) *= 0* (т.е. корень 𝑥̅ – кратный), то формула (4.10) уже не верна. Пусть кратность корня равна *m.* Тогда в силу формулы Тейлора справедливо приближенное равенство

*f* ( *x*) *≈ f*( 𝑥̅ ) *+* 𝑓′*(*𝑥̅) *(x* – 𝑥̅ ) + 𝑓𝘍𝘍(𝑥̅ ) (*x* – 𝑥̅)2*+ …+* 𝑓(𝑚)(𝑥̅ ) (*x* – 𝑥̅)𝑚*,*

2! 𝑚!

в правой части которого все слагаемые, кроме последнего, равны нулю. Следовательно, неравенство (4.9) имеет вид

| 𝑓(𝑚)(𝑥̅ ) (*x* – 𝑥̅)𝑚| < ∆̅.

𝑚!

Решая его, получаем аналогично (4.10) оценку радиуса интервала неопределенности:

𝖼̅ *≈* | 𝑚!

(𝑚)

𝑓 (𝑥̅ )

|1⁄𝑚∆̅1⁄𝑚*.*

Эта оценка означает, что для корня кратности *m* радиус интервала неопределенности пропорционален ∆̅1⁄𝑚, что свидетельствует о плохой обусловленности задачи вычисления кратных корней.

В реальных условиях оценить величину и даже порядок радиуса интервала неопределенности довольно сложно. Однако знать о его существовании нужно, по крайней мере, по двум причинам. Во-первых, не имеет смысла ставить задачу о вычислении корня 𝑥̅ с точностью ԑ <

𝖼̅. В условиях неопределенности, вызванных приближенным заданием функции, любое значение 𝑥̅ \* ϵ (𝑥̅ – 𝖼̅, 𝑥̅ + 𝖼̅ ) может быть с одной и той же степенью достоверности принято за решение уравнения. Во-вторых, нельзя требовать от алгоритмов отыскания корня получения достоверных результатов после того, как очередное приближение попало в интервал неопределенности или оказалось очень близко от него; в этой ситуации вычисления следует прекратить и считать, что получен максимум действительно возможного.

Для большинства итерационных методов определить этот момент можно, поскольку, начиная с него поведение приближений 𝑥(𝑛)

7

становится крайне нерегулярным. Если вдали от интервала неопределенности величина

𝑞(𝑛) = | 𝑥(𝑛) − 𝑥(𝑛−1)| / | 𝑥(𝑛−1) − 𝑥(𝑛−2)| (4.12)

обычно бывает меньше единицы (т.е. | 𝑥(𝑛) − 𝑥(𝑛−1)| < | 𝑥(𝑛−1) −

𝑥(𝑛−2)|), то появление при некотором при некотором *n* значений

𝑞(𝑛) > 1 свидетельствует о начале «разболтки» − хаотического поведения итерационной последовательности. В этой ситуации вычисления следует прекратить и принять правильное решение. Лучшим из последовательностей приближений к решению следует считать, конечно, 𝑥(𝑛−1). Использование для контроля вычислений величины (4.12) называют *правилом Гарвика*.

# Лекция 5

**4.3. Метод бисекции**

1. **Описание метода.** Пусть требуется с заданной точностью ԑ найти корень 𝑥̅ уравнения (4.1). Отрезок локализации [*a, b*] (т.е. отрезок, содержащий только один корень 𝑥̅) будем считать заданным. Предположим, что функция *f* непрерывна на отрезке [*a, b*] и на его концах принимает значения разных знаков, т.е.

*f( a) f( b) < 0.* (4.13)

Для дальнейшего будет удобно обозначить отрезок [*a, b*] через [𝑎(0), 𝑏(0)]. Примем за приближенное значение корня середину отрезка – точку 𝑥(0) = (𝑎(0) + 𝑏(0)) / 2. Так как положение корня на отрезке [𝑎(0), 𝑏(0)] неизвестно, то можно лишь утверждать, что погрешность этого приближения не превышает половины длины отрезка:

| 𝑥(0) − 𝑥̅ | ≤ (𝑏(0) − 𝑎(0)) / 2.

Уменьшить погрешность приближения можно уточняя отрезок локализации, т.е. заменяя начальный отрезок [𝑎(0), 𝑏(0)] отрезком [𝑎(1), 𝑏(1)] меньшей длины. Согласно *методу бисекции (половинного деления)* в качестве [𝑎(1), 𝑏(1)] берут тот из отрезков [𝑎(0), 𝑥(0)] и [𝑥(0), 𝑏(0)], на концах которого выполняется условие *f(*𝑎(1)) 𝑓*(*𝑏(1)) ≤

0. Этот отрезок содержит искомый корень. Середина полученного

отрезка 𝑥(1) = (𝑎(1) + 𝑏(1)) / 2 дает теперь приближение к корню, оценка погрешности которого составляет

| 𝑥(1) − 𝑥̅ | ≤ (𝑏(1) − 𝑎(1)) / 2 = *(b – a) /* 22.

За очередное уточнение отрезка локализации [𝑎(2), 𝑏(2)] снова берут тот из отрезков [𝑎(1), 𝑥(1)] и [𝑥(1), 𝑏(1)], на концах которого выполняется условие *f(*𝑎(2)) 𝑓*(*𝑏(2)) ≤ 0.

Опишем очередную (*n +* 1)-ю итерацию метода. Пусть отрезок [𝑎(𝑛), 𝑏(𝑛)] уже найден и вычислены значения 𝑥(𝑛), *f(*𝑎(𝑛)), 𝑓*(*𝑏(𝑛))*.* Тогда производятся следующие действия:

2

а) Вычисляется *f(*𝑥(𝑛)).

б) Если *f(*𝑎(𝑛)) 𝑓*(*𝑥(𝑛)) ≤ 0*,* то в качестве отрезка локализации [𝑎(𝑛+1), 𝑏(𝑛+1)] принимается отрезок [𝑎(𝑛), 𝑥(𝑛)]. В противном случае *f(*𝑥(𝑛)) 𝑓*(*𝑏(𝑛)) < 0 и за [𝑎(𝑛+1), 𝑏(𝑛+1)] принимается отрезок [𝑥(𝑛),

𝑏(𝑛)].

в) Вычисляется 𝑥(𝑛+1) = (𝑎(𝑛+1) + 𝑏(𝑛+1)) / 2.

Неограниченное продолжение итерационного процесса дает последовательность отрезков [𝑎(0), 𝑏(0)], [𝑎(1), 𝑏(1)] , …, [𝑎(𝑛), 𝑏(𝑛)],…, содержащих искомый корень.

1. **Скорость сходимости.** Середина *n*-го отрезка – точка 𝑥(𝑛) = (𝑎(𝑛) + 𝑏(𝑛)) / 2 дает приближение к корню 𝑥̅ , имеющее оценку погрешности

| 𝑥(𝑛) − 𝑥̅ | ≤ (𝑏(𝑛) − 𝑎(𝑛)) / 2 = *(b – a) /* 2𝑛+1. (4.14)

Из этой оценки видно, что метод бисекции сходится со скоростью геометрической прогрессии, знаменатель которой *q =* ½. По срвнению с другими методами метод бисекции сходится довольно медленно. Однако он очень прост и непритязателен; для его применения достаточно, чтобы выполнялось неравенство (4.13), функция *f* была непрерывна и верно определялся ее знак. В тех ситуациях, где не нужна сверхвысокая скорость сходимости, этот метод весьма привлекателен.

1. **Критерий окончания.** Итерации следует вести до тех пор, пока не будет выполнено неравенство (𝑏(𝑛) − 𝑎(𝑛)) < 2ԑ. При его выполнении в силу оценки (4.14) можно принять 𝑥(𝑛) за приближение к корню с точностью ԑ.
2. **Влияние вычислительной погрешности.** При использовании метода бисекции принципиально важно правильное определение знака функции *f.* В случае, когда 𝑥(𝑛) попадает в интервал неопределенности корня (см. § 4.2), знак вычисленного значения 𝑓∗(𝑥(𝑛)) не обязан быть верным, и последующие итерации не имеют смысла. Однако этот метод

3

следует признать очень надежным; он гарантирует точность приближения, примерно равную радиусу интервала неопределенности

𝖼̅, а большего требовать и нельзя, как было отмечено ранее.

# § 4.4. Метод простой итерации

* 1. **Описание метода.** Чтобы применить метод простой итерации для решения нелинейного уравнения (4.1), необходимо преобразовать это уравнение к следующему виду:

*x = φ(x). (4.15)*

Это преобразование *(приведение уравнения к виду, удобному для итераций)* можно выполнить различными способами; один из них мы рассмотрим далее. Функцию *φ* далее будем называть *итерационной функцией.*

Выберем каким-либо образом приближенное значение корня 𝑥(0)

и подставим его в правую часть уравнения (4.15). Получим значение

𝑥(1) = *φ (*𝑥(0)). Подставляя теперь 𝑥(1) в правую часть уравнения (4.15), получим 𝑥(2) = *φ (*𝑥(1)). Продолжая этот процесс неограниченно, получим последовательность приближений к корню, вычисляемых по формуле

𝑥(𝑛+1) = *φ (*𝑥(𝑛)), *n ≥ 0.* (4.16)

Очевидно, что метод простой итерации − одношаговый.

Если существует предел построенной последовательности 𝑥̅ = lim𝑛→∞ 𝑥(𝑛), то, переходя к пределу в равенстве (4.16) и предполагая функцию *φ* непрерывной, получим равенство

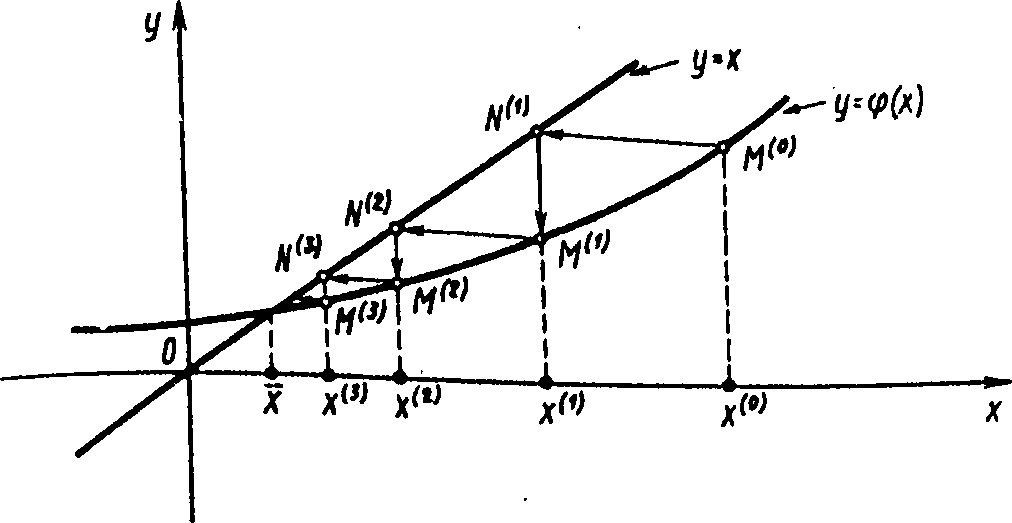
𝑥̅ = *φ(*𝑥̅). (4.17)

Это значит, что 𝑥̅ – корень уравнения (4.15).

* 1. **Геометрическая иллюстрация.** Из рис. 4.1 видно, что корень

𝑥̅ уравнения (4.15) является абсциссой точки пересечения графиков двух функций: *y = x* и *y = φ(x).* Возьмем некоторое начальное приближение 𝑥(0), которому отвечает расположенная на кривой *y = φ(x)* точка 𝑀(0) с координатами (𝑥(0), 𝑥(1)) (напомним, что 𝑥(1) = *φ (*𝑥(0))). Соединим точку 𝑀(0) отрезком прямой *y =* 𝑥(1) с лежащей на прямой *y = x* точкой 𝑁(1) с координатами (𝑥(1), 𝑥(1)). Проведем

4

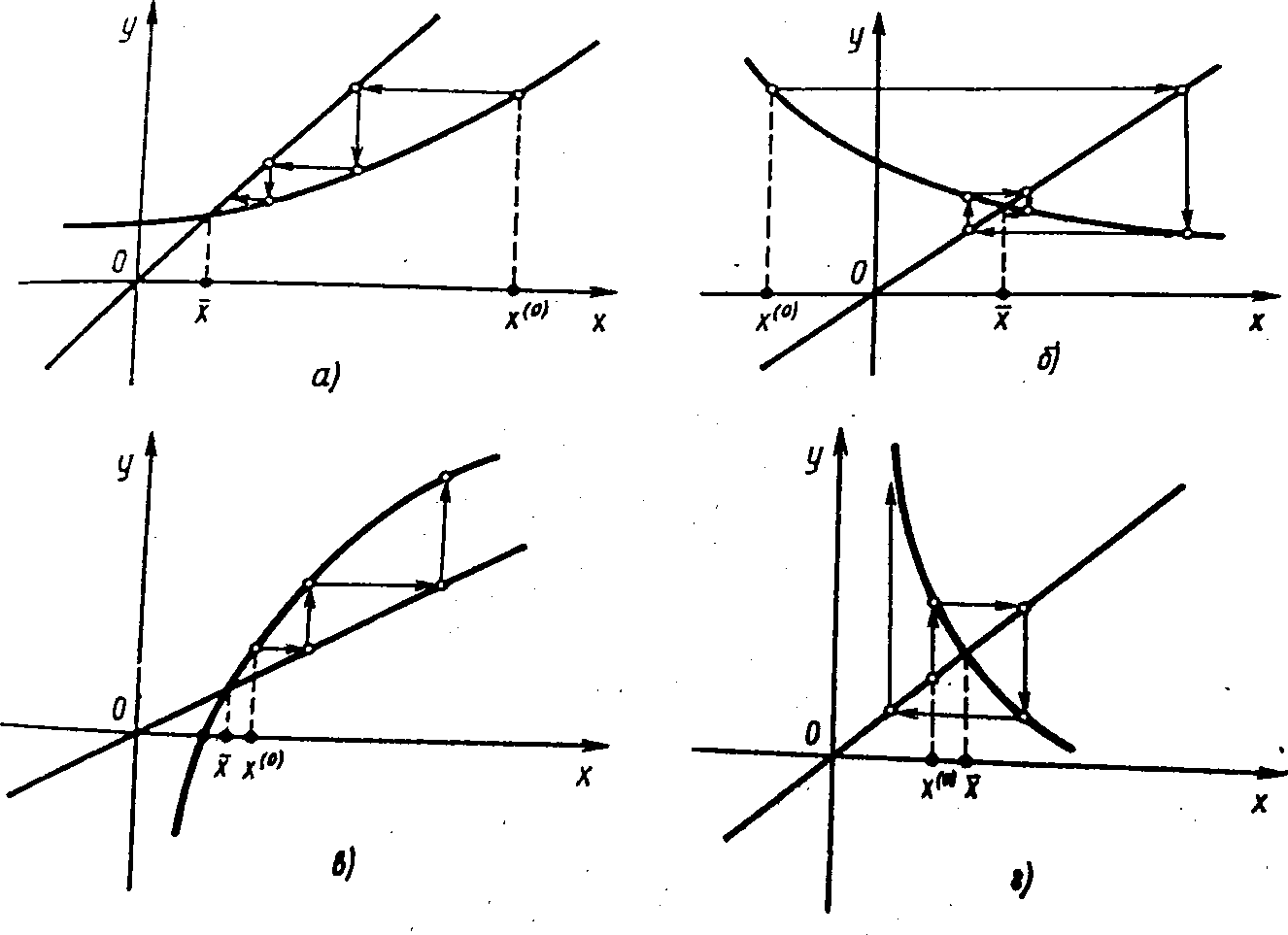
теперь через точку 𝑁(1) прямую *х =* 𝑥(1) до пересечения с кривой *y = φ(x)* в точке 𝑀(1) с координатами (𝑥(1), 𝑥(2)).

*Рис. 4.1*

Продолжая этот процесс далее, получаем ломаную линию

𝑀(0)𝑁(1)𝑀(1)𝑁(2) 𝑀(2)𝑁(3)…, для которой абсциссы точек 𝑀(𝑛)

представляют собой последовательные приближения к решению 𝑥̅.

* 1. **Сходимость метода.** На рис. 4.2, *а–г* представлена геометрическая иллюстрация поведения итерационного процесса в четырех простейших случаях взаимного расположения прямой *y = x* и кривой *y = φ(x).*

*Рис. 4.2*

5

В случаях *(а)* и *(б)* метод простой итерации сходится, причем, как нетрудно заметить, – при произвольном начальном приближении. Напротив, в случаях *(в)* и *(г)* метод расходится при любом выборе начального приближения. Заметим, что в случаях *(а)* и *(б)* |𝜑′*(x)| < 1,* а в случаях *(в)* и *(г)* наоборот, |𝜑′*(x)| > 1.* Таким образом, можно предположить, что сходимость метода простой итерации связана с выполнением условия |𝜑′*(x)| < 1.* Действительно, имеет место следующий результат.

Т е о р е м а 4.2. *Пусть в некоторой σ-окрестности корня* 𝑥̅

*функция φ дифференцируема и удовлетворяет неравенству*

*|*𝜑′*(x)| ≤ q, (4.18)*

*где 0 ≤ q < 1– постоянная.*

*Тогда независимо от выбора начального приближения* 𝑥(0) *из указанной σ-окрестности корня* 𝑥̅ *итерационная последовательность не выходит за пределы этой окрестности, метод сходится со скоростью геометрической прогрессии и справедлива следующая оценка погрешности:*

*|*𝑥(𝑛) − 𝑥̅| *≤* 𝑞𝑛*|*𝑥(0) − 𝑥̅|*. (4.19)*

*□* Вычитая из равенства (4.16) равенство (4.17) и используя формулу конечных приращений Лагранжа, получим

*|*𝑥(𝑛+1) − 𝑥̅| = *φ(* 𝑥(𝑛)*)* − *φ(*𝑥̅ *)* = 𝛼(𝑛+1)(𝑥(𝑛) − 𝑥̅*). (4.20)*

Здесь 𝛼(𝑛+1) = 𝜑′(𝜉(𝑛)), где 𝜉(𝑛) – некоторая точка, расположенная между 𝑥(𝑛) и 𝑥̅*.* Если 𝑥(𝑛) *ϵ (*𝑥̅ − 𝜎, 𝑥̅ + 𝜎), *|* 𝛼(𝑛+1)| *≤ q* в силу условия (4.18). Тогда на основании равенства (4.20) получаем

*|*𝑥(𝑛+1) − 𝑥̅| *≤ q |*𝑥(𝑛) − 𝑥̅|*.*

Это означает, что метод простой итерации обладает линейной скоростью сходимости и поэтому доказательство теоремы завершается применением леммы 4.1. *■*

Оценка погрешности (4.19) является априорной. Априорные оценки погрешности позволяют еще до вычислений дать некоторое заключение о качестве метода. В данном случае она показывает, что

6

метод простой итерации сходится со скоростью геометрической прогрессии со знаменателем *q.* Чем меньше *q,* тем выше скорость сходимости. Видна и роль правильного выбора начального приближения: чем меньше погрешность начального приближения, тем меньше итераций потребуется сделать для достижения заданной точности ε.

Неравенство (4.19) используется для теоретических оценок метода и практически непригодно для практической оценки погрешности. Это связано с тем, что значение 𝑥̅, входящее в правую часть оценки, неизвестно. Кроме того, использование неравенства (4.19) приводит к существенно завышенной оценке погрешности.

* 1. **Критерий окончания.** Выведем апостериорную оценку погрешности, пригодную для практического применения.

Т е о р е м а 4.3. *Пусть выполнены условия теоремы 4.2 и* 𝑥(0) *ϵ (*𝑥̅ − 𝜎, 𝑥̅ + 𝜎). *Тогда верна следующая апостериорная оценка погрешности:*

*|*𝑥(𝑛) − 𝑥̅| *≤* 𝑞

1−𝑞

*|*𝑥(𝑛) − 𝑥(𝑛−1) |*, n ≥1. (4.21)*

*□* В силу равенства (4.20) имеем

𝑥(𝑛) − 𝑥̅ = 𝛼(𝑛)(𝑥(𝑛−1) − 𝑥̅*) =* 𝛼(𝑛)(𝑥(𝑛−1) − 𝑥(𝑛)*)+* 𝛼(𝑛)(𝑥(𝑛) − 𝑥̅*)*

Отсюда

𝑥(𝑛) − 𝑥̅ = 𝛼(𝑛)

(𝑛)

1−𝛼

(𝑥(𝑛−1) − 𝑥(𝑛)*). (4.22)*

Взяв модуль от левой и правой частей этого равенства и

воспользовавшись неравенством | 𝛼(𝑛)

(𝑛)

1−𝛼

| < 𝑞 1−𝑞

, получим требуемое

соотношение (4.21). *■*

Если величина *q* известна, то неравенство (4.21) дает эффективный метод контроля погрешности и можно сформулировать следующий критерий окончания итерационного процесса: вычисления

следует вести до выполнения неравенства 𝑞

1−𝑞

| 𝑥(𝑛) − 𝑥(𝑛−1)| *< ε* или

равносильного ему неравенства

| 𝑥(𝑛) − 𝑥(𝑛−1)| < 1−𝑞 *ε. (4.23)*

𝑞

7

Если это условие выполнено, то можно считать, что 𝑥(𝑛) является приближением к 𝑥̅ с точностью ε.

Использование критерия (4.23) предполагает знание величины *q,* входящей в условие (4.18). Однако далеко не всегда эта величина известна либо может быть легко вычислена. В тех же случаях, когда удается оценить величину *q,* эта оценка оказывается довольно грубой.

Исключим из критерия окончания итераций величину *q.* Заметим, что в малой окрестности корня величина производной 𝜑′ практически постоянна: 𝜑′*(x) ≈* 𝜑′*(*𝑥̅). Поэтому в равенстве (4.22) величину 𝛼(𝑛) =

𝜑′*(*𝜉(𝑛−1)*)* можно приближенно заменить на 𝜑′*(*𝑥̅). Далее в силу равенства

𝑥(𝑛) − 𝑥(𝑛−1) = *φ(* 𝑥(𝑛−1)*)* − *φ(*𝑥(𝑛−2)*)* = 𝜑′*(*𝜉̃(𝑛)*)* (𝑥(𝑛−1) −𝑥(𝑛−2)),

где 𝜉̃(𝑛) – промежуточная между 𝑥(𝑛−1) и 𝑥(𝑛−2) точка, имеем

𝛼̃(𝑛) = (𝑥(𝑛) − 𝑥(𝑛−1)) / (𝑥(𝑛−1) − 𝑥(𝑛−2)) = 𝜑′*(*𝜉̃(𝑛)*) ≈* 𝜑′*(*𝑥̅). Таким образом, в равенстве (4.22) можно положить 𝛼(𝑛) *≈* 𝛼̃(𝑛) и поэтому при определенных условиях можно использовать следующий практический критерий окончания итерационного процесса:

| 𝑥(𝑛) − 𝑥(𝑛−1)| < | 1−𝛼̃(𝑛) | *ε* (4.25)

(𝑛)

𝛼̃

* 1. **Приведение уравнения к виду, удобному для итераций.** Ключевой момент в методе простых итераций – эквивалентное преобразование уравнения (4.1) к виду (4.15). Конечно, такое преобразование имеет смысл только тогда, когда оказывается выполненным условие (4.18) при 0 < *q* < 1. Рассмотрим один из простых способов такого преобразования.

Предположим, что производная 𝑓′ на отрезке [*a, b*] непрерывна и положительна. Тогда существуют постоянные *m* и *M* такие, что 0 < *m ≤* 𝑓′(𝑥) *≤ M, x ϵ* [ *a, b*]. Приведем уравнение (4.1) к виду

*x = x – αf(x),* где *α > 0.* (4.26)

8

В этом случае итерационная функция *φ* имеет вид *φ(x) = x – αf(x).* Как выбрать *α*, чтобы выполнялось условие (4.18), причем *q* было бы по возможности минимальным?

Заметим, что *1 – αM ≤* 𝜑′*(x) = 1 – α*𝑓′*(x) ≤ 1 – αm* и поэтому

*|*𝜑′*(x)| ≤ q(α) = max {|1 – αM |, |1 – αm* |}. Для того, чтобы было выполнено неравенство *q(α) <1,* достаточно взять любое *α ϵ (0, 2/M).* Конкретный выбор параметра *α* зависит от наличия информации о числах *m* и *M.* Если известны обе эти величины, то лучшим является выбор α = 𝛼0 = 2/( *M + m*). В этом случае *q(*𝛼0*) = (M – m)/ (M + m).* Если же известно только *M*, то можно положить *α =* 𝛼1 = 1/*M*. В этом

случае *q(*𝛼 *) = 1 –* 𝑚 *.*

1

𝑀

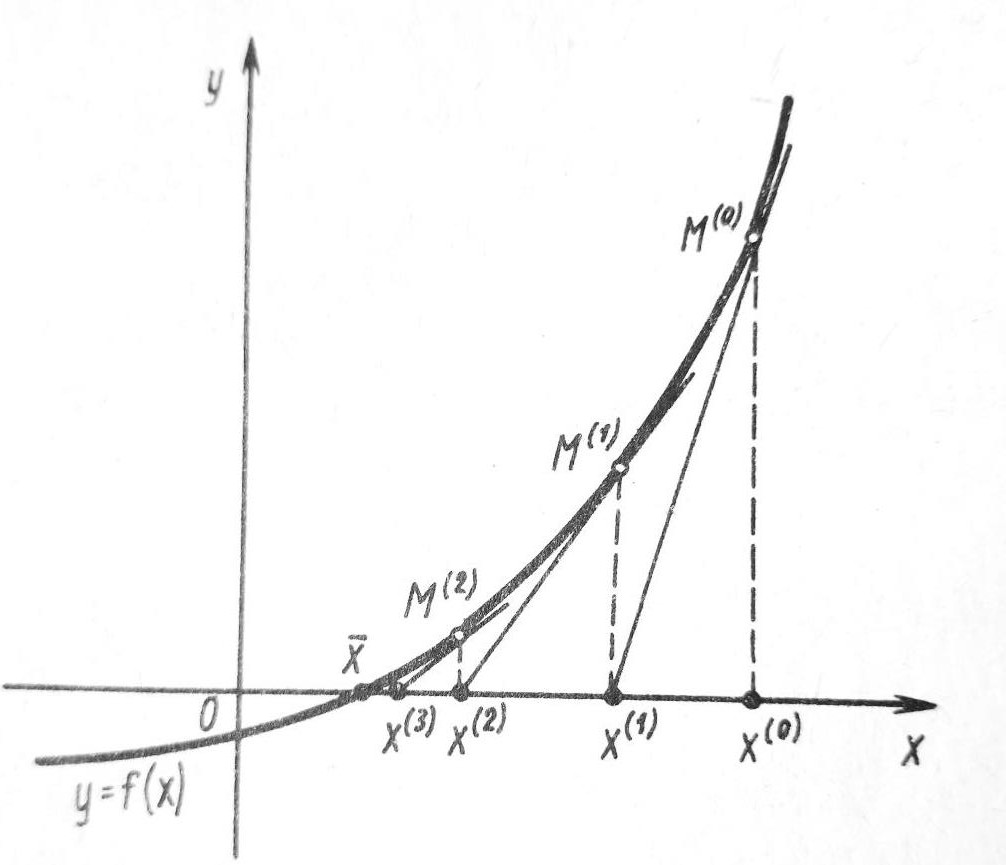
З а м е ч а н и е. Случай, когда производная 𝑓′ отрицательна, сводится к рассмотренному выше умножением уравнения *f(x)=0* на *–*1.

# Лекция 6

**§ 4.5. Метод Ньютона**

Знаменитый метод Ньютона является одним из наиболее эффективных методов решения самых разных нелинейных задач. Расчетную формулу метода можно получить, используя различные подходы. Рассмотрим два из них.

1. **Метод касательных.** Выведем расчетную формулу метода для решения нелинейного уравнения (4.1) из простых геометрических соображений. Соответствующая иллюстрация приведена на рис. 4.3.



*Рис. 4.3*

Пусть 𝑥(0) – заданное начальное приближение к корню 𝑥̅. В точке

𝑀(0) с координатами (𝑥(0), 𝑓(𝑥(0))) проведем касательную к графику функции *y = f*(*x*) и за новое приближение 𝑥(1) примем абсциссу точки пересечения этой касательной с осью *Ox.* Аналогично за приближение

𝑥(2) примем абсциссу точки пересечения с осью *Ox* касательной, проведенной к графику в точке 𝑀(1) с координатами (𝑥(1), 𝑓(𝑥(1))).

Продолжая этот процесс далее, получим последовательность

𝑥(0), 𝑥(1), 𝑥(2), …, 𝑥(𝑛), … приближений к корню 𝑥̅.

Напомним, что уравнение касательной, проведенной к графику функции *y = f* (*x*) в точке (𝑥(𝑛), 𝑓(𝑥(𝑛))), имеет вид

*y =* 𝑓(𝑥(𝑛)) +𝑓′(𝑥(𝑛))(*x* − 𝑥(𝑛)). (4.27) Полагая в равенстве (4.27) *y =* 0, замечаем, что при выполнении условия

𝑓′(𝑥(𝑛)) ≠ 0 абсцисса 𝑥(𝑛+1) точки пересечения касательной с осью *Ox*

удовлетворяет равенству

0 *=* 𝑓(𝑥(𝑛)) +𝑓′(𝑥(𝑛))(𝑥(𝑛+1) − 𝑥(𝑛)). (4.28) Выражая из него 𝑥(𝑛+1), получаем расчетную формулу *метода Ньютона:*

𝑥(𝑛+1) = 𝑥(𝑛) − 𝑓(𝑥(𝑛)) , *n* ≥ 0. (4.29)

𝘍 (𝑛)

𝑓 (𝑥 )

Благодаря такой геометрической интерпретации этот метод часто называют *методом касательных*.

1. **Метод линеаризации.** С более общих позиций метод Ньютона можно рассматривать как итерационный метод, использующий специальную линеаризацию задачи и позволяющий свести решение исходного нелинейного уравнения к решению последовательности линейных уравнений.

Пусть приближение 𝑥(𝑛) уже получено. Представим функцию в окрестности точки 𝑥(𝑛) по формуле Тейлора:

*f(x) =* 𝑓(𝑥(𝑛)) +𝑓′(𝑥(𝑛))(*x*− 𝑥(𝑛)) +𝑓𝘍𝘍(𝜉)

2

(𝑥 − 𝑥(𝑛)) 2. (4.30)

Здесь ξ − некоторая точка, расположенная между *x* и 𝑥(𝑛). Заменяя в уравнении *f(x) = 0* функцию *f(x)* главной линейной частью разложения (4.30), получим линейное уравнение

𝑓(𝑥(𝑛)) +𝑓′(𝑥(𝑛)) (*x* − 𝑥(𝑛)) = 0. (4.31)

Принимая решение уравнения (4.31) за новое приближение 𝑥(𝑛+1) , приходим к формуле (4.29).

# Основная теорема о сходимости метода Ньютона.

*Т е о р е м а 4.4. Пусть* 𝑥̅ *– простой корень уравнения f(x) = 0, в некоторой окрестности которого функция f дважды непрерывно дифференцируема. Тогда найдется такая малая ԑ-окрестность корня* 𝑥̅, *что при произвольном выборе начального приближения* 𝑥(0) *из этой окрестности итерационная последовательность метода Ньютона не выходит за пределы этой окрестности и справедлива оценка*

*|*𝑥(𝑛+1) − 𝑥̅| ≤ C|𝑥(𝑛) − 𝑥̅|2, *n* ≥ 0, (4.32) *где C =* 𝜎−1*, означающая, что метод сходится с квадратичной скоростью.*

*Следствием оценки (4.32) является априорная оценка погрешности*

*|*𝑥(𝑛) − 𝑥̅| ≤ σ𝑞2𝑛, *n ≥ 0, (4.33)*

*в которой q =* 𝜎−1 *|* 𝑥(0) − 𝑥̅|.

□ Так как 𝑓′(𝑥̅) ≠ 0 (по определению простого корня), то в силу непрерывности функций 𝑓′ и 𝑓′′ найдется 𝛿0-окрестность корня, в которой при некоторых постоянных α и β выполнены неравенства

0 < α ≤ |𝑓′(𝑥)|, |𝑓′′(𝑥)| ≤ β.

Пусть 𝑥(𝑛) ϵ (𝑥̅ *–* 𝞂*,* 𝑥̅ *+* 𝞂*),* где 𝞂 = *min {*𝛿 , 2α *}.* Подставляя *x =* 𝑥̅ в

0

𝛽

(4.30), получим равенство

0 = 𝑓(𝑥(𝑛)) +𝑓′(𝑥(𝑛))(𝑥̅ − 𝑥(𝑛)) + 𝑓𝘍𝘍(𝜉)

2

(𝑥̅ − 𝑥(𝑛))2, в котором

ξ ϵ (𝑥̅ *–* 𝞂*,* 𝑥̅ *+* 𝞂). Вычитая из него равенство (4.28), имеем

𝑓′(𝑥(𝑛))(𝑥(𝑛+1) − 𝑥̅) = 𝑓𝘍𝘍(𝜉)

2

(𝑥̅ − 𝑥(𝑛))2.

Тогда, приравняв модули обеих частей этого равенства и используя условия ограниченности |𝑓′(𝑥)| и |𝑓′′(𝑥)|, приходим к неравенству

α|𝑥(𝑛+1) − 𝑥̅ | ≤ 𝛽

2

(𝑥̅ − 𝑥(𝑛))2, откуда следует справедливость оценки

(4.32). Справедливость оценки (4.33) устанавливается методом индукции. ■

Таким образом, при выборе начального приближения из достаточно малой окрестности корня метод Ньютона сходится

квадратично. Это означает, грубо говоря, что на каждой итерации число верных цифр приближения примерно удваивается.

Приведенные в теореме 4.4 оценки погрешности являются априорными и их использование в практике вычислений невозможно.

1. **Критерий окончания.** На практике предпочтительнее пользоваться простой апостериорной оценкой

*|*𝑥(𝑛) − 𝑥̅| ≤ *|*𝑥(𝑛) − 𝑥(𝑛−1)|, (4.34) справедливость которой обосновывается следующим утверждением.

*Т е о р е м а 4.5. Пусть выполнены условия теоремы 4.4 и* 𝑥(0) ϵ (𝑥̅

*–* 𝞂|2*,* 𝑥̅ *+* 𝞂|2*). Тогда для всех n ≥ 1 верна оценка (4.34).*

*□* Из оценки (4.33) следует, что (𝑥(𝑛−1) − 𝑥̅) ≤ σ𝑞2𝑛−1 ≤ 𝞂q =

\𝑥(0) − 𝑥̅\ < 𝞂|2. Поэтому, применяя неравенство (4.32), получим цепочку неравенств 2 *|*𝑥(𝑛) − 𝑥̅| ≤ 2 𝜎−1 |𝑥(𝑛−1) − 𝑥̅|2 ≤ *|*𝑥(𝑛−1) − 𝑥̅| ≤

*|*𝑥(𝑛−1) − 𝑥(𝑛)| + *|*𝑥(𝑛) − 𝑥̅|, из которой вытекает оценка (4.34). *■*

Наличие оценки (4.34) позволяет сформулировать следующий практический критерий окончания итераций метода Ньютона. При заданной точности ԑ > 0 вычисления нужно вести до тех пор, пока не окажется выполненным неравенство

*|*𝑥(𝑛) − 𝑥(𝑛−1)| < ԑ (4.35)

Теоремы 4.4 и 4.5 утверждают лишь о существовании малой окрестности корня, в которой имеет место сходимость метода Ньютона. Однако о критериях выбора такой окрестности речь в них не идет. Приведем без доказательства упрощенный вариант теоремы об условиях, предъявляемых к выбору отрезка локализации корня, поведению функции на нем и выбору начального приближения для гарантированной сходимости метода Ньютона:

*Т е о р е м а 4.6. Пусть [a, b] - отрезок локализации простого корня* 𝑥̅ *уравнения f(x) = 0. Предположим, что на этом отрезке функция f дважды непрерывно дифференцируема, а ее производные*

𝑓′(𝑥) и 𝑓′′(*x*) *знакопостоянны.*

*Тогда начиная с n = 1 итерационная последовательность метода Ньютона* 𝑥(𝑛) *сходится к* 𝑥̅ *монотонно с той стороны отрезка [a, b], где f(x)* 𝑓′′(*x*) > 0.

Иллюстрацией монотонного характера сходимости может служить рис. 4.3.

С л е д с т в и е. *Пусть* 𝑥̅ *- корень уравнения f(x) = 0, функция f дважды непрерывно дифференцируема на всей числовой оси, а ее производные* 𝑓′(𝑥) и 𝑓′′(*x*) *знакопостоянны. Тогда метод Ньютона сходится при любом начальном приближении* 𝑥(0) *(т. е. является глобально сходящимся), причем начиная с n=1 итерационная последовательность* 𝑥(𝑛) *сходится к* 𝑥̅ *монотонно с той стороны от корня, где f(x)* 𝑓′′(*x*) > 0.

Таким образом, метод Ньютона обладает в общем случае только *локальной сходимостью.* Это означает, что областью его сходимости является некоторая малая 𝞂-окрестность корня, в которой функция должна быть гладкой (производные первого и второго порядка знакопостоянны), и для гарантии сходимости необходимо правильно выбирать начальное приближение.

Для практического применения метода Ньютона необходимо на каждом шаге вычислять производную 𝑓′(𝑥(𝑛)), однако часто бывает невозможно найти аналитическое выражение для 𝑓′, а определить приближенное значение с высокой точностью либо очень трудно, либо очень трудоемко делать это на каждом шаге. В этих случаях приходится модифицировать метод, избегая непосредственного вычисления производной. Рассмотрим некоторые из таких модификаций.

# § 4.6. Модификации метода Ньютона

Рассматриваемые ниже итерационные методы решения нелинейного уравнения на каждой итерации используют некоторую процедуру его линеаризации, т.е. исходное нелинейное уравнение заменяется приближенно более простым линейным уравнением.

1. **Упрощенный метод Ньютона.** Если производная 𝑓′(𝑥) непрерывна, то ее значение вблизи простого корня 𝑥̅ почти постоянно. Поэтому можно вычислить 𝑓′ лишь однажды в точке 𝑥(0), а затем заменить в формуле (4.29) значение 𝑓′(𝑥(𝑛)) постоянной 𝑓′(𝑥(0)). В

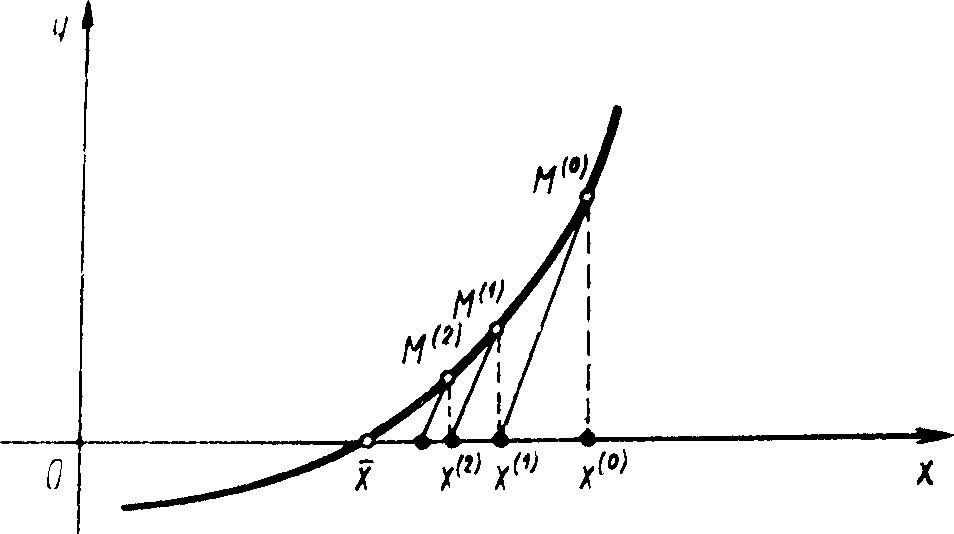
результате получим расчетную формулу *упрощенного метода Ньютона:*

𝑥(𝑛+1) = 𝑥(𝑛) − 𝑓(𝑥(𝑛)) , *n* ≥ 0. (4.36)

𝘍 (0)

𝑓 (𝑥 )

Геометрическая иллюстрация метода приведена на рис. 4.4.



*Рис. 4.4*

В точке (𝑥(0), 𝑓(𝑥(0))) к графику функции *y = f(x)* проводится касательная 𝑙0 и за приближение 𝑥(1) принимается абсцисса точки пересечения этой касательной с осью *Ох* (как в методе Ньютона). Каждое следующее приближение 𝑥(𝑛+1) получается здесь как абсцисса точки пересечения с осью *Ох* прямой, проходящей через точку 𝑀𝑛 с координатами (𝑥(𝑛), 𝑓(𝑥(𝑛))) и параллельной касательной 𝑙0.

Упрощение вычислений по сравнению с методом Ньютона достигается здесь ценой резкого падения скорости сходимости. Сходимость этого метода является уже не квадратичной, а линейной.

Метод (4.36) можно рассматривать как метод простой итерации с

итерационной функцией *φ(x) = x* − 𝑓(𝑥) *.*

𝘍 (0)

𝑓 (𝑥 )

Так как 𝜑′(𝑥) = 1 − 𝑓𝘍(𝑥) , то для знаменателя *q*

𝑓𝘍(𝑥(0))

соответствующей геометрической прогрессии имеем

𝑓𝘍(𝑥̅)

*q* ≈ | 1 –

|.

𝑓𝘍(𝑥(0))

Следовательно, скорость сходимости тем выше, чем ближе начальное приближение 𝑥(0) к решению 𝑥̅.

1. **Метод хорд.** В основе этой и следующей модификаций лежит приближенное равенство

𝑓′(𝑥(𝑛)) *≈* 𝑓(𝑧(𝑛))−𝑓(𝑥(𝑛))*.* (4.37)

(𝑛) (𝑛)

𝑧 − 𝑥

Оно верно при условии 𝑧(𝑛) ≈ 𝑥(𝑛) и следует из определения производной: 𝑓′(𝑥) = lim 𝑓(𝑧)−𝑓(𝑥).

𝑧 → 𝑥

𝑧−𝑥

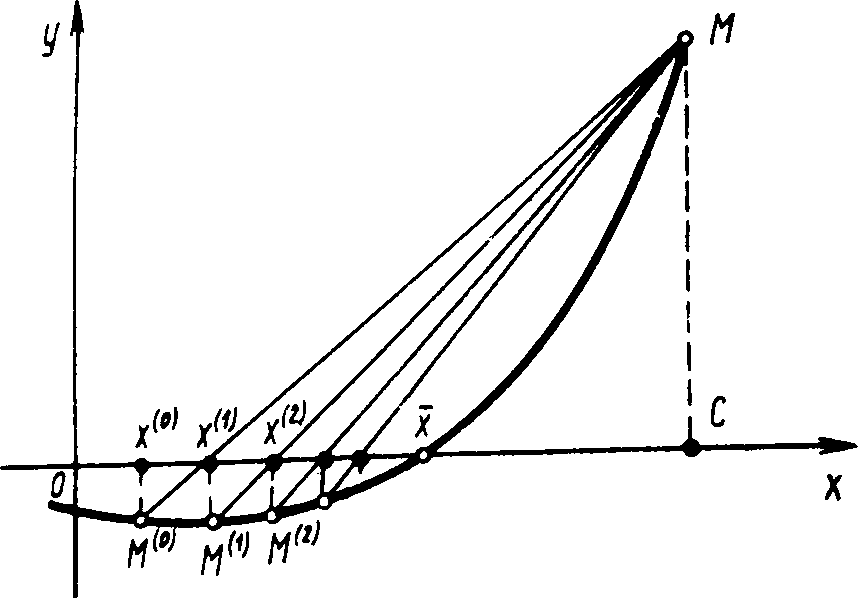
Пусть *с* – фиксированная точка, расположенная в окрестности простого корня 𝑥̅. Заменим в расчетной формуле метода Ньютона производную 𝑓′(𝑥(𝑛)) правой частью приближенного равенства (4.37), полагая 𝑧(𝑛) = *с.* В результате придем к расчетной формуле *метода хорд:*

𝑥(𝑛+1) = 𝑥(𝑛) − с − 𝑥(𝑛)

(𝑛)

𝑓(𝑐)−𝑓(𝑥 )

· 𝑓(𝑥(𝑛), *n* ≥ 0 (4.38)

Геометрическая иллюстрация метода приведена на рис. 4.5.

*Рис. 4.5*

Очередное приближение 𝑥(𝑛+1) получается здесь как абсцисса точки пересечения с осью *Ох* прямой, проходящей через расположенные на графике функции *y = f(x)* точки *M* и 𝑀(𝑛) с координатами *(c, f(c))* и (𝑥(𝑛), 𝑓(𝑥(𝑛))).

Метод (4.38) обладает только линейной сходимостью. Его можно

рассматривать как метод простой итерации с итерационной функцией

*φ(x) = x* − 𝑐−𝑥

𝑓(𝑐)−𝑓(𝑥)

*f(x).* Так как скорость сходимости определяется

вблизи корня величиной *q ≈ |*𝜑′(𝑥)*| = | 1 -* (𝑐− 𝑥̅)𝑓𝘍(𝑥̅)

𝑓(𝑐)−𝑓(𝑥̅)

выше, чем ближе окажется выбранная точка *с* к 𝑥̅.

*|,* то она тем

1. **Метод секущих.** Замена в формуле метода Ньютона

производной 𝑓′(𝑥(𝑛)) приближением 𝑓(𝑥(𝑛−1))−𝑓(𝑥(𝑛))

(𝑛−1) (𝑛)

𝑥 − 𝑥

приводит к

расчетной формуле *метода секущих*:

(𝑛+1) (𝑛) 𝑥(𝑛−1)− 𝑥(𝑛)

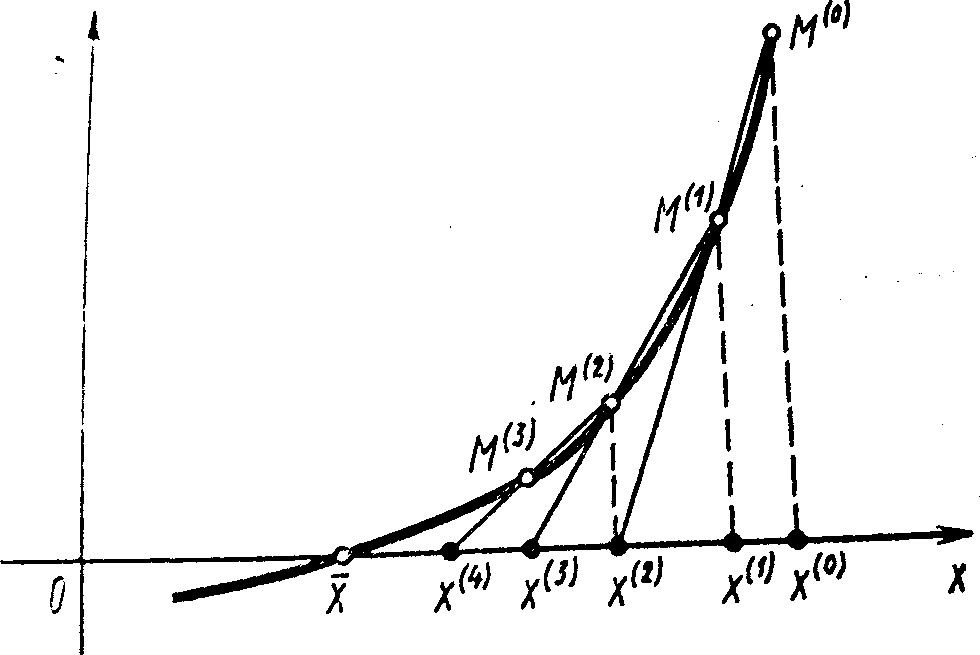
𝑥 = 𝑥 −

𝑓(𝑥(𝑛−1))−𝑓(𝑥(𝑛))

𝑓(𝑥(𝑛)), *n* ≥ 1. (4.39)

Заметим, что этот метод двухшаговый, так как для нахождения очередного приближения 𝑥(𝑛+1) требуется знание двух предыдущих приближений 𝑥(𝑛) и 𝑥(𝑛−1). В частности, для того чтобы начать вычисления, необходимо задать два начальных приближения 𝑥(0) и 𝑥(1).

На рис. 4.6 приведена геометрическая иллюстрация метода.



*Рис. 4.6*

Очередное приближение 𝑥(𝑛+1) получается здесь как абсцисса точки пересечения с осью *Ох* секущей, соединяющей точки 𝑀(𝑛−1) и

𝑀(𝑛) графика функции *f(x)* с координатами (𝑥(𝑛−1), 𝑓(𝑥(𝑛−1))) и (𝑥(𝑛), 𝑓(𝑥(𝑛))).

Примечательно, что эта модификация метода Ньютона сохраняет сверхлинейную скорость сходимости, если вычисляется простой корень

𝑥̅. Точнее, верно следующее утверждение.

Т е о р е м а 4.7. *Пусть* 𝑥̅ *– простой корень уравнения f(x) = 0, в некоторой окрестности которого функция f дважды непрерывно дифференцируема, причем* 𝑓′′*(*𝑥̅) ≠ 0. *Тогда существует* 𝞂*- окрестность корня* 𝑥̅ *такая, что при произвольном выборе приближений* 𝑥(0) и 𝑥(1) *из этой* 𝞂*-окрестности*

Метод сходится с порядком p = 1+√5

2

≈ 1.618, т.е. для n ≥ 1

справедлива оценка

|𝑥(𝑛+1) − 𝑥̅| ≤ c |𝑥(𝑛) − 𝑥̅ |𝑝, р = 1+√5.

2

Так как одна итерация метода секущих требует только одного нового вычисления значения функции *f*, а метод Ньютона – двух вычислений (*f* и 𝑓′), то трудоемкость двух итераций метода секущих приблизительно эквивалентна трудоемкости одной итерации метода Ньютона. Две итерации метода секущих дают порядок 𝑝2 ≈ 2.618 > 2, поэтому его можно расценивать как более быстрый по сравнению с методом Ньютона.

К сожалению, метод обладает только локальной сходимостью. Он требует выбора двух близких к корню начальных приближений

𝑥(0) и 𝑥(1). Если эти приближения выбраны неудачно, то метод расходится.

# Лекция 7

**Тема 5. Решение систем линейных алгебраических уравнений**

# § 5.1. Постановка задачи

В вычислительной линейной алгебре особое значение имеет задача решения систем линейных алгебраических уравнений и связанные с ней задачи вычисления определителей и нахождения обратных матриц. Далее мы их рассмотрим. В основном же в данной теме мы рассмотрим прямые методы решения систем линейных алгебраических уравнений (СЛАУ) с вещественными коэффициентами:

𝑎11𝑥1*+* 𝑎12𝑥2*+*𝑎13𝑥3*+…*+ 𝑎1𝑚𝑥𝑚 *=* 𝑏1*,*

𝑎21𝑥1*+* 𝑎22𝑥2*+*𝑎23𝑥3*+…*+ 𝑎2𝑚𝑥𝑚 *=* 𝑏2*,*

𝑎31𝑥1*+* 𝑎32𝑥2*+*𝑎33𝑥3*+…*+ 𝑎3𝑚𝑥𝑚 *=* 𝑏3*, (5.1)*

*. . . . . . . . . . . . . . . . . . . . . .*

𝑎𝑚1𝑥1*+*𝑎𝑚2𝑥2*+*𝑎𝑚3𝑥3*+…*+𝑎𝑚𝑚𝑥𝑚*=* 𝑏𝑚*.*

В матричной форме записи эта система принимает вид

***Ax = b****, (5.2)*

где

𝑎11 𝑎12 𝑎13 … 𝑎1𝑚

𝖥 𝑎21 𝑎22 𝑎23 … 𝑎2𝑚 1

I I

𝑥1

𝖥 𝑥21

I I

𝖥𝑏1 1

I 𝑏2I

***A*** *=* I𝑎31 𝑎32 𝑎33 … 𝑎3𝑚 I, ***x****=*I 𝑥3I, ***b****=*I 𝑏3I.

I I

[𝑎𝑚1𝑎𝑚2𝑎𝑚3 … 𝑎𝑚𝑚]

I . . . I

[𝑥𝑚]

I . . . I

[𝑏𝑚]

Уделим основное внимание задаче вычисления вектора ***x****,*

являющегося решением системы (5.2), по входному данному – вектору

***b****.* Будем предполагать, что матрица ***А*** задана и является невырожденной. Известно, что в этом случае решение системы существует, единственно и устойчиво по входным данным. Это означает, что задача решения СЛАУ корректна.

2

Пусть 𝒙∗ = (𝑥∗, 𝑥∗, … , 𝑥∗ )т – приближенное решение системы

1 2 𝑚

(5.1). Будем стремиться к получению решения, для которого погрешность ***e = x* –** 𝒙∗ мала (количественные значения величины погрешности рассмотрим далее). Заметим, однако, что качество решения не всегда характеризуется тем, насколько мала погрешность ***x* –** 𝒙∗. Иногда вполне удовлетворительным является критерий малости *невязки* ***r = b – A***𝒙∗***.*** Вектор ***r*** показывает, насколько отличается правая часть системы от левой, если в нее подставить приближенное решение. Заметим, что ***r = b* – *A***𝒙∗ ***= A (x* –** 𝒙∗) и поэтому погрешность и невязка связаны равенством

***e = x* –** 𝒙∗***=*** 𝑨−𝟏***r.*** (5.3)

# § 5.2. Нормы вектора и матрицы

1. **Норма вектора.** Решением системы линейных алгебраических уравнений является вектор ***x =*** *(*𝑥1, 𝑥2, … , 𝑥𝑚)т, который будем рассматривать как элемент векторного пространства 𝑹𝒎**.** Приближенное

решение 𝒙∗ = (𝑥∗, 𝑥∗, … , 𝑥∗ )т и погрешность ***e = x* –** 𝒙∗*= (*𝑥 − 𝑥∗, … ,

1 2 𝑚 1 1

𝑥𝑚 − 𝑥∗ *)* также являются элементами пространства 𝑹𝒎. Для того, чтобы анализировать методы решения систем, необходимо уметь количественно оценивать «величины» векторов ***x*** и ***x* –** 𝒙∗***,*** а также

𝑚

векторов ***b*** и ***b* –** 𝒃∗***,*** где 𝒃∗ = (𝑏∗, 𝑏∗, … , 𝑏∗ )т – вектор приближенно

1 2 𝑚

заданных правых частей. Удобной для этой цели количественной характеристикой является широко используемое понятие нормы вектора.

Говорят, что в пространстве 𝑹𝒎 задана норма, если каждому вектору ***x*** из 𝑹𝒎 сопоставлено вещественное число ǁ ***x*** ǁ, называемое нормой вектора ***x*** и обладающее следующими свойствами:

1. ǁ ***x*** ǁ ≥ 0, причем ǁ ***x*** ǁ = 0 тогда и только тогда, когда ***x*** = 0;
2. ǁα ***x*** ǁ = |α| ǁ ***x*** ǁ для любого вектора ***x*** и любого числа α;
3. ǁ ***x + y*** ǁ ≤ ǁ ***x*** ǁ + ǁ ***y*** ǁ для любых векторов ***x*** и ***y;***

последнее неравенство принято называть *неравенством треугольника*.

3

Заметим, что такими же свойствами обладает обычная геометрическая длина вектора в трехмерном пространстве. Свойство 3) в этом случае следует из правила сложения векторов и из того известного факта, что сумма длин двух сторон треугольника всегда больше длины третьей стороны.

Существуют различные способы введения норм. В вычислительной математике наиболее употребительными являются следующие три нормы:

ǁ***x***ǁ1= ∑𝑚

𝑖=1

|𝑥𝑖|, ǁ***x***ǁ2 =( ∑𝑚

|𝑥𝑖|2 )1/2, ǁ***x***ǁ∞= max1≤𝑖≤𝑚 |𝑥𝑖| (5.4)

Первые две из них являются частными случаями более общей нормы:

𝑖=1

ǁ ***x*** ǁ𝑝 = (∑𝑚 |𝑥𝑖|𝑝 )1/𝑝, *p* ≥ 1 (5.5)

𝑖=1

(при *p=1* и *p=2),* а последняя получается из нормы (5.5) предельным переходом при *p* → ∞.

З а м е ч а н и е 1. Норма ǁ ***x*** ǁ2 является естественным обощением на случай *m-*мерного пространства понятия длины вектора в двух- и трехмерных геометрических пространствах. Поэтому ее называют *евклидовой* нормой.

З а м е ч а н и е 2. Справедливы неравенства

ǁ ***x*** ǁ∞ ≤ ǁ ***x*** ǁ2 ≤ ǁ 𝒙 ǁ1 ≤ *m* ǁ ***x*** ǁ∞, (5.6)

Указывающие на то, что в определенном смысле все три введенные нормы эквивалентны: каждая из них оценивается любой из двух других с точностью до множителя, зависящего от *m.*

1. **Скалярное произведение.** Напомним, что скалярным произведением векторов ***x =*** *(*𝑥1, 𝑥2, … , 𝑥𝑚)т и ***y =*** *(*𝑦1, 𝑦2, … , 𝑦𝑚)т называется величина

(***x, y) =*** 𝑥1𝑦1*+*𝑥2𝑦2*+ … +*𝑥𝑚𝑦𝑚 *=* ∑𝑚 𝑥𝑖𝑦𝑖*.* (5.7)

𝑖=1

Нетрудно установить, что ǁ ***x*** ǁ2 = (***x, x***)1/2***.***

4

В случае, когда векторы ***x, y*** имеют комплексные компоненты, скалярное произведение понимают так:

(***x, y) =*** ∑𝑚

𝑖=1

𝑥𝑖

𝑦̅𝑖*.*

1. **Абсолютная и относительная погрешности вектора.** Далее будем считать, что в пространстве *m*-мерных векторов 𝑅𝑚 введена и фиксирована некоторая норма ǁ ***x*** ǁ (например, одна из норм ǁ ***x*** ǁ𝑝, 1 ≤ *p* ≤ ∞). В этом случае в качестве меры близости векторов ***x*** и 𝒙∗ естественно использовать величину ǁ ***x*** − 𝒙∗ǁ , являющуюся аналогом расстояния между точками ***x*** и 𝒙∗**.** Введем абсолютную и относительную погрешности вектора 𝒙∗ с помощью формул

**∆(**𝒙∗) **=** ǁ ***x*** − 𝒙∗ǁ, δ(𝒙∗) = ǁ 𝒙 − 𝒙∗ǁ **.** (5.8)

ǁ 𝒙 ǁ

Выбор той или иной конкретной нормы в практических задачах диктуется тем, какие требования предъявляются к точности решения.

1. **Сходимость по норме.** Пусть {𝒙(𝑛)}∞ − последовательность векторов 𝒙(𝑛) = (𝑥(𝑛), 𝑥 (𝑛), … , 𝑥(𝑛)). Говорят, что последовательность

𝑛=1

1 2 𝑚

векторов 𝒙(𝑛) сходится к вектору ***x*** при *n → ∞* (𝒙(𝑛) → ***x*** при *n → ∞),*

если **∆(**𝒙(𝑛)) **=** ǁ ***x*** − 𝒙(𝑛)ǁ → 0 при *n → ∞* .

З а м е ч а н и е. Сам факт наличия или отсутствия сходимости

𝒙(𝑛) → ***x*** при *n → ∞* в конечномерных пространствах не зависит от выбора нормы (см. неравенство (5.6)). Более того, 𝒙(𝑛) → ***x*** при *n →*

*∞)* тогда и только тогда, когда для всех *i=1,2,…,m* имеем 𝑥 (𝑛) → 𝑥𝑖 при *n → ∞ ,* т.е. сходимость по норме в 𝑅𝑚 эквивалентна покомпонентной (покоординатной) сходимости.

𝑖

1. **Норма матрицы.** Величина ǁAǁ = max ǁAxǁ

x≠0

ǁxǁ

5

(5.9)

называется *нормой матрицы* А, подчиненной норме векторов, введенной в 𝑅𝑚.

Заметим, что множество всех квадратных матриц размером *m˟ m* является векторным пространством. Можно показать, что введенная в этом пространстве формулой (5.9) норма обладает следующими свойствами, аналогичными свойствам нормы вектора:

1. ǁ ***A*** ǁ ≥ 0, причем ǁ ***A*** ǁ = 0 тогда и только тогда, когда ***A*** = 0;
2. ǁα ***A*** ǁ = |α| ǁ ***A*** ǁ для любой матрицы ***А*** и любого числа α;
3. ǁ ***A + B*** ǁ ≤ ǁ ***A*** ǁ + ǁ B ǁ для любых матриц ***A*** и ***B.***

Дополнительно к этому верны следующие свойства:

1. ǁ***A·B*** ǁ ≤ ǁ ***A*** ǁ·ǁ B ǁ для любых матриц ***A*** и ***B;***
2. Для любой матрицы ***A*** и любого вектора ***x*** справедливо неравенство

ǁ***A x***ǁ ≤ ǁ ***A*** ǁ·ǁ***x***ǁ. (5.10)

Как следует из неравенства (5.9), каждой из векторных норм ǁ***x***ǁ соответствует своя подчиненная норма матрицы ***A.*** Известно, в частности, что нормам ǁ ***x*** ǁ1, ǁ ***x*** ǁ2 и ǁ ***x*** ǁ∞ подчинены нормы ǁ***А***ǁ1, ǁ***А***ǁ2 и ǁ***А***ǁ∞, вычисляемые по формулам

ǁ***А***ǁ1= max1≤𝑗≤𝑚 ∑𝑚 |𝑎𝑖𝑗 |, (5.11)

𝑖=1

ǁ***А***ǁ = max

2

1≤𝑗≤𝑚

√𝜆𝑗(АтА) , (5.12)

где 𝜆𝑗(АтА) – собственные числа матрицы АтА;

ǁ***А***ǁ∞= max1≤𝑖≤𝑚 ∑𝑚 |𝑎𝑖𝑗 |. (5.13)

𝑗=1

Нормы ǁ***А***ǁ1 и ǁ***А***ǁ∞ вычисляются просто, а для вычисления ǁ***А***ǁ2 Необходимо найти собственные числа 𝜆𝑗, а это иногда бывает непросто сделать.

6

Поэтому используют оценку

ǁ***А***ǁ2*≤* ǁ***А***ǁЕ. (5.14)

Здесь ǁ***А***ǁЕ = √∑𝑚

𝑖,𝑗=1

|𝑎𝑖𝑗|2

*-* величина, называемая *евклидовой нормой*

*матрицы* ***А.***

Норма (5.9) имеет простую геометрическую интерпретацию. Для того, чтобы ее привести, заметим, что операцию умножения матрицы на вектор можно рассматривать как преобразование, которое переводит вектор ***x*** в новый вектор ***y*** = ***Ax***. Если значение ǁ***x***ǁ интерпретируется как длина вектора ***x,*** то величина ǁ***Ax***ǁ/ǁ***x***ǁ есть коэффициент растяжения вектора под действием матрицы ***А***. Таким образом, величина

𝑘𝑚𝑎𝑥 *=* ǁ***A***ǁ *=* maxx≠0 ǁAxǁ

ǁxǁ

(5.15)

представляет собой максимальный коэффициент растяжения вектора ***x*** под действием матрицы ***A.*** Полезно отметить, что для невырожденной матрицы ***A*** минимальный коэффициент растяжения 𝑘𝑚𝑖𝑛 отвечает норме обратной матрицы и вычисляется по формуле

𝑘𝑚𝑖𝑛 = ǁ 𝐴−1ǁ−1 = min𝑥≠0 ǁ𝑨 𝒙ǁ. (5.16)

ǁ𝒙ǁ

Заметим, что в случае ǁ***A***ǁ < 1 происходит сжатие вектора под действием матрицы.

# § 5.3. Обусловленность задачи

**решения системы линейных алгебраических уравнений**

Оказывается, что решения различных систем линейных алгебраических уравнений обладают разной чувствительностью к погрешностям входных данных. Так же как и другие задачи, задача вычисления решения ***x*** системы уравнений

7

***Ax = b*** (5.17)

может быть как хорошо, так и плохо обусловленной.

Рассмотрим случай, когда элементы матрицы ***A*** считаются заданными точно, а вектор-столбец правой части – приближенно.

Л е м м а 5.1. *Для погрешности приближенного решения системы (5.17) справедлива оценка*

**∆(**𝒙∗) **≤** ǁ𝑨−𝟏ǁ · ǁ***r***ǁ*,* (5.18)

*где* ***r = b* – *A***𝒙∗ – *невязка, отвечающая* 𝒙∗*.*

Для доказательства достаточно взять норму левой и правой частей равенства (5.3) ( ***x* –** 𝒙∗***=*** 𝑨−𝟏***r )*** и воспользоваться свойством (5.10) ( ǁ***Ax***ǁ ≤ ǁ***A***ǁ·ǁ***x***ǁ ).

Т е о р е м а 5.1. *Пусть* 𝒙∗ **–** *точное решение системы* ***A***𝒙∗***=*** 𝒃∗, *в которой правая часть* 𝒃∗*является приближением к* ***b.*** *Тогда верны следующие оценки абсолютной и относительной погрешностей:*

**∆(**𝒙∗) **≤** 𝑣𝛥 **∆(**𝒃∗)**,** (5.19)

δ**(**𝒙∗) **≤** 𝑣𝛿 **δ(**𝒃∗)**,** (5.20)

*где* 𝑣 **=** ǁ𝑨−𝟏ǁ, 𝑣 **=** ǁ𝑨−𝟏ǁ ǁ𝒃ǁ **.**

𝛥 𝛿

ǁ𝒙ǁ

**□** В рассматриваемом случае ***r = b* – *A***𝒙∗***= b -*** 𝒃∗ и неравенство (5.18) принимает вид (5.19). Разделив обе части неравенства (5.19) на ǁ 𝒙ǁ и записав его в виде

∆(𝒙∗) **≤** ǁ𝑨−𝟏ǁ ǁ 𝒃ǁ **·**∆(𝒃∗),

ǁ 𝒙ǁ

ǁ 𝒙ǁ

ǁ𝒃ǁ

приходим к оценке (5.20). **■**

З а м е ч а н и е 1. Величина 𝑣𝛥 **=** ǁ𝑨−𝟏ǁ для задачи (5.17) играет роль абсолютного числа обусловленности.

З а м е ч а н и е 2. Величина 𝑣 **=** ǁ𝑨−𝟏ǁ ǁ𝒃ǁ

𝛿

ǁ𝒙ǁ

называется *естественным*

*числом обусловленности.* Она зависит от конкретного решения ***x*** и характеризует коэффициент возможного возрастания относительной

8

погрешности этого решения, вызванной погрешностью задания правой части. Это означает, что 𝑣𝛿(𝒙) для задачи вычисления решения ***x*** системы (5.17) играет роль относительного числа обусловленности.

Вычислим максимальное значение естественного числа обусловленности, используя определение (5.9) нормы матрицы:

max

𝑥≠0

𝑣𝛿

(𝒙) = max

𝑥≠0

ǁ𝑨−𝟏ǁ·ǁ𝑨 𝒙ǁ ǁ 𝒙ǁ

= ǁ𝑨−𝟏ǁ · ǁ𝑨ǁ. (5.21)

Полученную величину принято называть *стандартным числом обусловленности* (или просто *числом обусловленности*) матрицы ***A*** и обозначать через ν(***A*** ) или cond (***A)***. Таким образом,

ν(***A***) = cond (***A***) = ǁ𝑨−𝟏ǁ · ǁ𝑨ǁ. (5.22) Сформулируем важное следствие из теоремы 5.1.

С л е д с т в и е. *В условиях теоремы 5.1 справедлива оценка*

*δ(*𝑥∗*) ≤* cond (***A***) *δ(*𝑏∗*).* (5.23)

Для ее доказательства достаточно воспользоваться оценкой (5.20) и заметить, что в силу определения (5.21) верно неравенство

𝑣𝛿 **≤** cond (***A***).

Величина cond(***A***) является широко используемой количественной мерой обусловленности системы ***Ax = b.*** В частности, систему и матрицу ***A*** принято называть *плохо обусловленными*, если cond(***A***) >> 1. В силу оценки (5.23) для такой системы существуют решения, обладающие чрезвычайно высокой чувствительностью к малым погрешностям входных данных – вектора ***b.*** Тем не менее, заметим, что не для всякого решения ***x*** коэффициент 𝑣𝛿(𝒙) роста относительной ошибки достигает значений, близких к максимально возможному значению cond(***A***).

Отметим следующие свойства числа обусловленности. 1). Для единичной матрицы cond(***Е***) = 1.

□ Пользуясь тем, что 𝑬−𝟏 = ***E*** и ǁ𝑬ǁ = 1, получим cond(***Е***) =

ǁ𝑬−𝟏ǁ · ǁ𝑬ǁ = 1. ■

9

1. Справедливо неравенство cond(***A***) ≥ 1.

□ Из равенства ***Е = A*** 𝑨−1, свойства 4) норм матриц и равенства

ǁ 𝑬ǁ = 1 следует, что 1 = ǁ 𝑬ǁ ≤ ǁ𝑨−𝟏ǁ · ǁ𝑨ǁ = cond(***A***). ■

1. Число обусловленности матрицы ***A*** не меняется при умножении матрицы ***A*** на произвольное число α ≠ 0.

□ Заметим, что (α ***A***)−𝟏***=*** 𝜶−𝟏 𝑨−1*.* Поэтому cond(α***A***) = ǁ𝜶−𝟏𝑨−𝟏ǁ · ǁα𝑨ǁ = |𝜶−𝟏| ǁ𝑨−𝟏ǁ · |α | ǁ𝑨ǁ = cond(***A***). ■

З а м е ч а н и е. Пользуясь приведенной в **§** 5.2 геометрической интерпретацией норм матриц 𝑨 и 𝑨−1 (формулы (5.15) и (5.16)), число обусловленности можно интерпретировать как отношение максимального коэффициента растяжения векторов под действием матрицы 𝑨 к минимальному коэффициенту: cond(A) = 𝑘𝑚𝑎𝑥 / 𝑘𝑚𝑖𝑛.

Выше рассмотрен случай, когда элементы матрицы ***A*** считаются заданными точно, а вектор-столбец правой части – приближенно. Однако на практике это часто не так. Установлено, что введенная выше величина cond(A) характеризует также и чувствительность решений системы к малым погрешностям задания элементов матрицы ***A.*** В частности, справедлива следующая теорема.

Т е о р е м а 5.2. *Пусть* 𝒙∗ – *точное решение системы* 𝑨∗𝒙∗ = 𝒃 *с приближенно заданной матрицей* 𝑨∗*. Тогда верна следующая оценка относительной погрешности:*

𝛿∗(𝒙∗***)*** *≤* cond(A) · δ(𝑨∗*),*

*где* 𝛿∗(𝒙∗*) =* ǁ𝒙 − 𝒙∗ǁ / ǁ𝒙∗ǁ, δ(𝑨∗*) =* ǁ𝑨 − 𝑨∗ǁ / ǁ𝑨ǁ.

В случае, когда с погрешностью заданы как правая часть системы, так и матрица, доказано, что справедливо неравенство

δ(𝒙∗***)*** *≤* cond(A) · (δ(𝑨∗) + δ(𝑏∗))

З а м е ч а н и е. Вычисление чисел обусловленности непосредственно по представленным выше формулам предполагает предварительное вычисление обратной матрицы, что является достаточно трудоемкой операцией (порядка 2𝑚3 арифметических операций). На практике избегают этой операции. При этом исходят из того, что в большинстве случаев достаточно лишь грубой оценки числа обусловленности с точностью до порядка. Для этого существуют достаточно эффективные методы, которые мы оставим за рамками нашего курса.

10

# § 5.4. Типы используемых матриц

Эффективность вычислений в линейной алгебре существенно зависит от умения использовать специальную структуру и свойства используемых в расчетах матриц. Рассмотрим некоторые типы матриц.

Квадратная матрица называется *диагональной*, если ее элементы удовлетворяют условию 𝑎𝑖𝑗 = 0 для *i ≠ j* (т.е. все отличные от нуля элементы расположены на главной диагонали).

Диагональная матрица, у которой все элементы 𝑎𝑖𝑖 = 1,

называется *единичной* и обозначается буквой ***Е***.

Важную роль в численном анализе играют *треугольные матрицы.* Квадратная матрица ***А*** называется *нижней треугольной*, если все ее элементы, расположенные выше главной диагонали, равны нулю (𝑎𝑖𝑗 = 0 для *i > j)*. Если же равны нулю все элементы, расположенные ниже главной диагонали (𝑎𝑖𝑗 = 0 для *i < j* ), то такая матрица называется *верхней треугольной*.

Для треугольных матриц определитель вычисляется по формуле

det ***A*** = 𝑎11 𝑎22 … 𝑎𝑚𝑚. (5.24)

Будем называть симметричную матрицу ***A*** положительно определенной и писать ***A >*** 0, если для всех векторов ***x*** ≠ 0

квадратичная форма (***Ax, x***) = ∑𝑚 𝑎𝑖𝑗 𝑥𝑖𝑥𝑗 принимает

𝑖,𝑗=1

положительные значения.

На практике часто встречаются матрицы, в которых число ненулевых элементов много меньше общего числа элементов матрицы. Такие матрицы принято называть *разреженными*. Матрицы общего вида называют *плотными* (или *заполненными*).

Простой пример разреженной матрицы дает *трехдиагональная* матрица, все ненулевые элементы котрой расположены на главной и двух соседних диагоналях. Число ненулевых элементов такой матрицы равно (3*m* -2) << 𝑚2 при большом *m.*

# Лекция 8

**§ 5.5. Метод Гаусса. LU – разложение.**

Рассмотрим один из самых распространенных методов решения СЛАУ – метод Гаусса. Вычисления с помощью метода Гаусса состоят из двух этапов, называемых *прямым ходом* и *обратным ходом*. Прямой ход заключается в последовательном исключении неизвестных из системы (5.1) для преобразования ее к эквивалентной системе с верхней треугольной матрицей. Вычисления значений неизвестных производят на этапе обратного хода.

1. **Схема единственного деления.** Рассмотрим сначала простейший вариант метода Гаусса для решения системы линейных алгебраических уравнений ***Ax = b,*** называемый *схемой единственного деления.*

**П р я м о й х о д** состоит из *m* – 1 шагов исключения неизвестных из системы (5.1).

1. й ш а г. Целью этого шага является исключение неизвестного 𝑥1 из уравнений с номерами *i* = 2, 3, … , *m*. Предположим, что коэффициент 𝑎11≠ 0. Будем называть его *главным (или ведущим) элементом 1-го шага.*

Найдем величины

𝜇𝑖1 = 𝑎𝑖1/ 𝑎11 (*i =* 2, 3, … *,m),* (5.25) называемые *множителями 1-го шага.* Вычтем последовательно из второго, третьего, … , *m*-го уравнений системы (5.1) первое уравнение, умноженное соответственно на 𝜇21, 𝜇31, … 𝜇𝑚1. Это позволит обратить в нуль коэффициенты при 𝑥1 во всех уравнениях, кроме первого. В результате получим эквивалентную систему

𝑎11𝑥1*+* 𝑎12𝑥2*+*𝑎13𝑥3*+…*+ 𝑎1𝑚𝑥𝑚 *=* 𝑏1*,*

𝑎(1)𝑥2*+*𝑎(1)𝑥3*+…+*𝑎(1) 𝑥𝑚 *=*𝑏(1)*,*

22 23 2𝑚 2

𝑎(1)𝑥2*+*𝑎(1)𝑥3*+…+*𝑎(1) 𝑥𝑚 *=*𝑏(1)*,* (5.26)

32 33 3𝑚 3

*. . . . . . . . . . . . . . . . . . . . . .*

𝑎(1) 𝑥2*+*𝑎(1) 𝑥3*+…+*𝑎(1) 𝑥𝑚 *=*𝑏(1)*,*

𝑚2

𝑚3

𝑚𝑚 𝑚

в которой 𝑎(1) и 𝑏(1) (*i, j =* 2, 3*, … , m)* вычисляются по формулам

𝑖𝑗 𝑖

𝑎(1)*=* 𝑎𝑖𝑗 – 𝜇𝑖1𝑎1𝑗, 𝑏(1)*=* 𝑏𝑖 *-* 𝜇𝑖1𝑏1. (5.27)

𝑖𝑗 𝑖

Таким образом, в результате 1-го шага исключения по схеме единственного деления система уравнений (5.2) ***Ax = b*** приводится к виду

𝐴(1)***x =*** 𝒃(𝟏)***,*** (5.28)

где

𝑎11 𝑎12 𝑎13 … 𝑎1𝑚

𝖥 0 a(1)a(1) … a(1) 1

𝖥 𝑏1 1

𝑏(1)

I 22 23 2𝑚 I

I 2 I

𝑨(𝟏) =

I 0 a(1)a(1) … a(1) I, 𝒃(𝟏)*=* I

𝑏(1)I *,*

I 32 33 3𝑚 I

I 3 I

I I

I . . . I

[0 a(1) a(1) … a(1) ]

[ 𝑏(1) ]

𝑚2

𝑚3

𝑚𝑚 𝑚

а коэффициенты 𝑎(1)*,* 𝑏(1) вычисляются по формулам (5.25), (5.27).

𝑖𝑗 𝑖

Представим алгоритм вычисления 1-го шага в матричном виде.

Введем матрицу

𝖥 1 0 0 … 01

I−𝜇21 1 0 … 0

𝑴1 = I−𝜇31 0 1 … 0 I .

I

I I

[ −𝜇𝑚10 0 … 1 ]

Как нетрудно проверить, справедливы равенства

𝑨(𝟏) = 𝑴1𝑨***,*** 𝒃(𝟏)***=*** 𝑴1𝒃***,***

т.е. преобразование системы (5.2) к виду (5.28) эквивалентно умножению левой и правой частей системы (5.2) на матрицу 𝑴1.

1. й ш а г. Целью этого шага является исключение неизвестного 𝑥2

из уравнений с номерами *i* = 3, 4, … , *m*. Предположим, что коэффициент a(1) ≠ 0. Будем называть его *главным (или ведущим) элементом 2-го шага.* Вычислим множители 2-го шага

22

𝜇𝑖2 = a(1) / a(1) (*i =* 3, 4, … *,m)*

𝑖2 22

и вычтем последовательно из третьего, четвертого,… , *m*-го уравнений системы (5.26) второе уравнение, умноженное соответственно на 𝜇32,

𝜇42, … 𝜇𝑚2. Это позволит обратить в нуль коэффициенты при 𝑥2 во всех уравнениях, кроме первого и второго. В результате получим эквивалентную систему

𝑎11𝑥1*+* 𝑎12𝑥2*+*𝑎13𝑥3*+ …+* 𝑎1𝑚𝑥𝑚 *=* 𝑏1*,*

𝑎(1)𝑥2*+*𝑎(1)𝑥3*+…+* 𝑎(1) 𝑥𝑚 *=*𝑏(1)*,*

22 23 2𝑚 2

𝑎(2)𝑥3*+…+* 𝑎(2) 𝑥𝑚 *=*𝑏(2)*,* (5.29)

33 3𝑚 3

*. . . . . . . . . . . . . . . . . . . . . .*

𝑎(2) 𝑥3*+…+* 𝑎(2) 𝑥𝑚 *=*𝑏(2)*,*

𝑚3 𝑚𝑚 𝑚

в которой коэффициенты 𝑎(2) и 𝑏(2) (*i, j =* 3, 4*, … , m)* вычисляются по

формулам

𝑖𝑗 𝑖

𝑎(2)*=* 𝑎(1) – 𝜇𝑖2𝑎(1), 𝑏(2)*=* 𝑏(1) – 𝜇𝑖2𝑏(1). (5.30)

𝑖𝑗

𝑖𝑗

2𝑗 𝑖 𝑖 2

Таким образом, в результате 2-го шага исключения по схеме единственного деления система уравнений 𝐴(1)***x =*** 𝒃(𝟏)приводится к виду

где

𝐴(2)***x =*** 𝒃(𝟐)***,***

𝑎11 𝑎12 𝑎13 … 𝑎1𝑚

𝖥0 a(1)a(1) … a(1) 1

𝖥 𝑏1 1

𝑏(1)

I 22 23 2𝑚I

I 2 I

𝑨(𝟐) =

I0 0 a(2) … a(2) I, 𝒃(𝟐)*=* I

𝑏(2)I *,*

I 33 3𝑚 I

I 3 I

I I

I . . . I

[0 0 a(2) … a(2) ]

[ 𝑏(2) ]

𝑚3

𝑚𝑚 𝑚

а коэффициенты 𝑎(2)*,* 𝑏(2) вычисляются по формулам (5.30).

𝑖𝑗 𝑖

Представим алгоритм вычислений 2-го шага в матричном виде.

Введем матрицу

𝖥 1 0 0 … 0 1

𝑴2 = I

I

I [

0 1 0 … 0 I

0 − 𝜇32 1 … 0 I .

. . . . . . . . . . . . . . . I

0 − 𝜇𝑚2 0 … 1]

Как нетрудно проверить, справедливы равенства

𝑨(𝟐) = 𝑴2𝑨(𝟏)***,*** 𝒃(𝟐)***=*** 𝑴2𝒃(𝟏)***,***

т.е. преобразование системы (5.26) к виду (5.29) эквивалентно умножению левой и правой частей системы (5.26) на матрицу 𝑴1.

Аналогично проводятся остальные шаги. Опишем очередной *k*-й

шаг.

*k*-й ш а г. В предположении, что *главный (ведущий) элемент k*-го

шага a(𝑘−1) ≠ 0, вычислим *множители k-го шага*

𝑘𝑘

𝜇𝑖𝑘 = a(𝑘−1) / a(𝑘−1) (*i = k+1*, *k+2*, … *,m)*

𝑖𝑘 𝑘𝑘

И вычтем последовательно из (*k+*1)-го,…, *m*-го уравнений полученной на предыдущем шаге системы *k*-е уравнение, умноженное соответственно на 𝜇𝑘+1,𝑘, 𝜇𝑘+2,𝑘, … 𝜇𝑚𝑘.

После (*m*−1)-го шага исключения получим эквивалентную систему уравнений

𝑎11𝑥1*+* 𝑎12𝑥2*+*𝑎13𝑥3*+ …+* 𝑎1𝑚𝑥𝑚 *=* 𝑏1*,*

𝑎(1)𝑥2*+*𝑎(1)𝑥3*+…+* 𝑎(1) 𝑥𝑚 *=*𝑏(1)*,*

22 23 2𝑚 2

𝑎(2)𝑥3*+…+* 𝑎(2) 𝑥𝑚 *=*𝑏(2)*,* (5.31)

33 3𝑚 3

*. . . . . . . . . . . . . . . . . . . . . .*

𝑎(𝑚−1)𝑥𝑚 *=*𝑏(𝑚−1)*,*

𝑚𝑚 𝑚

Матрица 𝑨(𝒎−𝟏) которой является верхней треугольной. То есть после (*m*−1)-го шага, завершающего прямой ход, система оказывается приведенной к виду

𝑨(𝒎−𝟏)𝒙 = 𝒃(𝒎−𝟏)**.** (5.32)

Здесь 𝑨(𝒎−𝟏) = 𝑴𝑚−1𝑨(𝒎−𝟐)***,*** 𝒃(𝒎−𝟏)***=*** 𝑴𝑚−1𝒃(𝒎−𝟐)***,***

𝑎11 𝑎12 𝑎13 … 𝑎1𝑚

𝖥 0 a(1)a(1) … a(1) 1

1 0 0 … 0 0

0 1 0 … 0 01

𝖥

I

(𝒎−𝟏)

22 23

2𝑚 I

I … … … … … … … . . I

𝑨 = I

I I

0 0 a(2) … a(2)

I , 𝑴𝑚−1= I I .

33 3𝑚

I I

I 0 0 0 … 1 0 I

I 0 0 0 − 𝜇𝑚,𝑚−1 1 I

[ 0 0 0 … a(𝑚−1) ] [ ]

𝑚𝑚

𝖥 𝑏1 1

I

𝒃(𝒎−𝟏)*=* I

I I

𝑏(1) I

𝑏(2) I*.*

2

3

. . . I

I

[ 𝑏(𝑚−1)]

𝑚

Заметим, что матрица 𝑨(𝒎−𝟏) получена из матрицы 𝑨

последовательным умножением на 𝑴1, 𝑴2, …, 𝑴𝑚−1:

𝑨(𝒎−𝟏) = 𝑴𝑚−1 … 𝑴2 𝑴1 𝑨 (5.33) Аналогично,

𝒃(𝒎−𝟏) = 𝑴𝑚−1 … 𝑴2 𝑴1 𝒃 (5.34) Из равенства (5.33) вытекает следующее представление:

***A*** = 𝑀−1𝑀−1 … 𝑀−1 𝐴(𝑚−1) (5.35)

1 2 𝑚−1

Как легко проверить,

𝖥 1 0 0 … 0

1 0 0 … 0

𝑀−1 =

I 𝜇21 1 0 … 0

𝜇31 0 1 … 0

1 𝖥

I , 𝑀−1 = I

0 1 0 … 0 1

0 − 𝜇32 1 … 0 I

, … ,

1 I I

I I

[ 𝜇𝑚10 0 … 1 ]

2 I I

I I

[ 0 − 𝜇𝑚2 0 … 1]

𝖥 1 0 0 … 0 0 1

I

𝑀−1 = I

0 1 0 … 0 0

… … … … … … … . . I

I.

𝑚−1

I 0 0 0 … 1 0 I

I 0 0 0 … 𝜇𝑚,𝑚−11I

[ ]

Для этого достаточно перемножить матрицы 𝑀−1 и 𝑀𝑘 (*k*=1, 2, …, *m*-1), в результате чего получится единичная матрица.

𝑘

Введем обозначения ***U =*** 𝐴(𝑚−1), ***L*** = 𝑀−1𝑀−1 … 𝑀−1 . Вычисляя

1 2 𝑚−1

матрицу ***L,*** убеждаемся, что она имеет следующий вид:

𝖥 1 0 0 … 0 1

I 𝜇21 1 0 … 0 I

***L =*** I 𝜇31 𝜇32 1 … 0 I. (5.36)

I I

[ 𝜇𝑚1𝜇𝑚2𝜇𝑚3 … 1]

Тогда равенство (5.35) в новых обозначениях примет вид

***A = LU.*** (5.37)

Это и есть *LU-разложение матрицы* ***A*** – представление матрицы ***A*** в виде произведения нижней треугольной матрицы ***L*** и верхней треугольной матрицы ***U***.

Таким образом, прямой ход метода Гаусса можно рассматривать как процесс вычисления *LU*- разложения матрицы системы, на *k*-м шаге которого определяются элементы *k-*го столбца матрицы ***L*** и *k*-й строки матрицы ***U***.

**О б р а т н ы й х о д.** Из последнего уравнения системы (5.31) находим 𝑥𝑚. Подставляя найденное значение 𝑥𝑚 в предпоследнее уравнение, получим 𝑥𝑚−1. Осуществляя обратную подстановку далее, находим последовательно 𝑥𝑚−2, 𝑥𝑚−3, … , 𝑥1. Вычисления неизвестных здесь проводятся по формулам

𝑥𝑚 *=* 𝑏(𝑚−1)*/*𝑎(𝑚−1), (5.38)

𝑚 𝑚𝑚

𝑥𝑘 *=* (𝑏(𝑘−1) − 𝑎(𝑘−1)𝑥𝑘+1 − ⋯ − 𝑎(𝑘−1)𝑥𝑚 )/ 𝑎(𝑘−1),

𝑘

(*k=m*−1, … ,1)

𝑘,𝑘+1

𝑘𝑚

𝑘𝑘

Т р у д о е м к о с т ь м е т о д а. Не вдаваясь в подробные вычисления, скажем лишь, что общее число операций прямого хода Q ≈ 2/3 𝑚3. Как нетрудно видеть, для реализации обратного хода по формулам (5.38) нужно всего 𝑚2 операций, что при больших *m* пренебрежимо мало по сравнению с числом операций прямого хода.

Таким образом, для реализации метода Гаусса по схеме единственного деления требуется примерно (2/3 𝑚3 + 𝑚2) операций, причем подавляющее число этих операций совершается на этапе прямого хода.

Н е д о с т а т о к с х е м ы е д и н с т в е н н о г о д е л е н и я. Заметим, что вычисление множителей 𝜇𝑖𝑗, а также обратная подстановка

требуют деления на главные (или ведущие) элементы 𝑎(𝑘−1). Поэтому, если один из главных элементов оказывается равным нулю, то схема единственного деления не может быть реализована. Даже если все главные элементы отличны от нуля, но среди них есть близкие к нулю, возможен неконтролируемый рост погрешности. Таким образом, метод может оказаться некорректным, т.е. привести к аварийному останову

𝑘𝑘

(если 𝑎(𝑘−1) = 0 при некотором *k*) и вычисления по нему могут оказаться неустойчивыми. Для преодоления указанного недостатка применяются другие варианты метода Гаусса. Рассмотрим два из них.

𝑘𝑘

# Метод Гаусса с выбором главного элемента по столбцу (схема частичного выбора).

О п и с а н и е м е т о д а. На *k-*м шаге прямого хода коэффициенты уравнений системы с номерами *i = k+1, … ,m* преобразуются по формулам

𝑎(𝑘) = 𝑎(𝑘−1)– 𝜇𝑖𝑘𝑎(𝑘−1), 𝑏(𝑘)*=* 𝑏(𝑘−1)– 𝜇𝑖𝑘𝑏(𝑘−1), *i = k+1, … ,m* (5.39)

𝑖𝑗

𝑖𝑗

𝑘𝑗 𝑖 𝑖 𝑘

Очевидно, что во избежание сильного роста коэффициентов системы и связанных с этим ошибок нельзя допускать появления больших множителей 𝜇𝑖𝑘*.*

В методе Гаусса с выбором главного элемента по столбцу гарантируется, что |𝜇𝑖𝑘| *≤* 1 для всех *k =1,2,…,m-1* и *i = k+1, …, m.*

Отличие этого варианта метода Гаусса от схемы единственного деления заключается в том, что на *k*-м шаге исключения в качестве главного элемента выбирают максимальный по модулю коэффициент

𝑎𝑖𝑘𝑘 при неизвестной 𝑥𝑘 в уравнениях с номерами *i = k, k+1, …, m*. Затем соответствующее выбранному коэффициенту уравнение с номером 𝑖𝑘 меняют местами с *k*-м уравнением системы для того, чтобы

главный элемент занял место коэффициента 𝑎(𝑘−1). После этой перестановки исключение неизвестного 𝑥𝑘 производят как в схеме единственного деления.

𝑘𝑘

Заметим, что дополнительная работа по выбору главных элементов в схеме частичного выбора требует порядка 𝑚2 операций, что практически не влияет на общую трудоемкость метода.

В ы ч и с л и т е л ь н а я у с т о й ч и во с т ь с х е м ы ч а с т и ч - н о г о в ы б о р а. Детальное исследование метода Гаусса показывает, что основной причиной неустойчивости схемы единственного деления является возможность неограниченного роста элементов промежуточных матриц 𝐴(1), 𝐴(2), … , 𝐴(𝑚−1) в процессе прямого хода. Так как на *k*-м шаге схемы частичного выбора |𝜇𝑖𝑘| *≤* 1, то для

вычисленных по формулам (5.39) элементов 𝑎(𝑘) справедлива оценка

𝑖𝑗

|𝑎(𝑘)| *≤* |𝑎(𝑘−1)| + |𝑎(𝑘−1)|*.* Следовательно, максимальное по

𝑖𝑗

𝑖𝑗

𝑘𝑗

модулю значение элементов матрицы возрастает на одном шаге не более чем в два раза и в самом неблагоприятном случае *m-1* шаг прямого хода даст коэффициент роста *φ(m)* = 2𝑚−1.

Гарантия ограниченности роста элементов матрицы делает схему частичного выбора вычислительно устойчивой. Более того, для нее оказывается справедливой следующая оценка погрешности:

δ(𝑥∗) < *f(m)·* con𝑑𝐸 (**A**)·𝜀𝑀. (5.40)

Здесь 𝑥∗ - вычисленное на ЭВМ решение системы; δ(𝑥∗) = ||𝑥−𝑥∗||

||𝑥||

- его

относительная погрешность; con𝑑𝐸(**A**)=||**A**||·||𝐴−1|| - число обусловленности матрицы **A;** 𝜀𝑀 – машинное эпсилон; наконец,

*f(m)= C(m) φ(m),* причем *C(m) –* некоторая медленно растущая функция, зависящая от порядка *m* системы, а *φ(m) –* коэффициент роста. Наличие

в оценке (5.40) множителя *φ(m)* = 2𝑚−1 указывает на то, что при большом *m* схема частичного выбора может оказаться плохо обусловленной и возможна ощутимая потеря точности. Однако практика матричных вычислений показывает, что существенный рост элементов матрицы происходит крайне редко. В подавляющем большинстве случаев действительное значение коэффициента роста не превышает 8 – 10. Если система хорошо обусловлена, то погрешность вычисленного значения оказывается, как правило, малой.

1. **Метод Гаусса с выбором главного элемента по всей матрице (схема полного выбора).** В этой схеме допускается нарушение естественного порядка исключения неизвестных.

На первом шаге среди элементов 𝑎𝑖𝑗 выбирается максимальный по

модулю элемент 𝑎𝑖1𝑗1 . Первое уравнение системы и уравнение с номером 𝑖1 меняются местами. Далее стандартным образом производят исключение неизвестного 𝑥𝑗1 из всех уравнений, кроме первого.

На *k*-м шаге среди элементов 𝑎(𝑘−1) при неизвестных в уравнениях системы с номерами *i* = *k, …, m* выбирают максимальный по модулю

𝑖𝑗

элемент 𝑎(𝑘−1). Затем *k*-е уравнение и уравнение, содержащее

𝑖 𝑗

𝑘 𝑘

найденный коэффициент, меняют местами и исключают неизвестное

𝑥𝑗𝑘 из уравнений с номерами *i* = *k+1, …, m.*

На этапе обратного хода неизвестные вычисляют в следующем порядке: 𝑥𝑗𝑚, 𝑥𝑗𝑚−1 , … , 𝑥𝑗1 .

Схема полного выбора по сравнению со схемой частичного

выбора дает существенное замедление роста элементов матрицы. Доказано, что для нее коэффициент роста *φ(m)*, входящий в оценку (5.40), не превышает величины 1.8 𝑚0.25 ln(𝑚), что значительно меньше соответствующего значения *φ(m)* = 2𝑚−1 для схемы частичного выбора. Подчеркнем, что до сих пор еще не найдено матрицы, для которой полный выбор дал бы значение *φ(m) > m.* Таким образом, для хорошо обусловленных систем этот вариант метода Гаусса является хорошо обусловленным.

Однако гарантия хорошей обусловленности достигается здесь ценой значительных затрат на выбор главных элементов. Общая трудоемкость этого варианта метода Гаусса составляет порядка 𝑚3 операций.

# Лекция 9

**§ 5.6. Применение метода Гаусса к решению задач линейной алгебры**

1. **Вычисление решений системы уравнений с несколькими правыми частями.** Довольно часто на практике встречается ситуация, когда нужно решить несколько систем линейных алгебраических уравнений

***Ax =*** 𝒅(𝟏)***, Ax =*** 𝒅(𝟐)***, … , Ax =*** 𝒅(𝒑) (5.41)

с одной матрицей ***A*** и различными правыми частями 𝒅(𝟏)***,*** 𝒅(𝟐)***, … ,***𝒅(𝒑). Конечно, применяя метод Гаусса к каждой из систем (5.41) независимо от других, можно найти соответствующие решения

𝒙(𝟏)***,*** 𝒙(𝟐)***, … ,***𝒙(𝒑)***,*** затратив примерно (2/3)*p*𝑚3 арифметических операций. Однако при одновременном решении систем (5.41) число операций можно существенно сократить. Как было отмечено выше, основные затраты в методе Гаусса связаны с преобразованием матрицы к треугольному виду. Преобразование же правой части производится параллельно и требует примерно 𝑚2 арифметических операций. Если параллельно с приведением матрицы ***A*** к треугольному виду выполнить преобразование всех *p* правых частей по однотипным формулам, то на прямой ход будет затрачено примерно (2/3)𝑚3 *+ p*𝑚2 операций. С учетом обратного хода, который в данном случае выполняется *p* раз, общие вычислительные затраты составят (2/3)𝑚3 *+* 2*p*𝑚2 операций.

1. **Вычисление обратной матрицы.** Вычисление обратной матрицы является довольно трудоемкой задачей, однако эта задача возникает не так часто, как это можно предполагать. К сожалению, зачастую обращение матрицы ***A*** производится с единственной целью вычислить по известному вектору ***b*** вектор ***x*** вида ***x =*** 𝑨−𝟏𝒃 (т.е. найти решение системы ***Ax = b***). Умножение матрицы 𝑨−𝟏 на вектор ***b*** требует примерно 2𝑚2 арифметических операций. Однако вычисление

𝑨−𝟏 обходится (как будет показано ниже) примерно в 2𝑚3 операций. Это означает, что на вычисление решения системы ***Ax = b*** по формуле ***x =*** 𝑨−𝟏𝒃 будет затрачено примерно 2𝑚3 + 2𝑚2 операций. В

2

данном случае вектор ***x*** можно найти в 3 раза быстрее методом Гаусса и вычисление обратной матрицы не требуется. Более того, вычисленное методом Гаусса решение окажется точнее, так как потребуется выполнение меньшего числа операций.

Может показаться выгодным предварительное вычисление обратной матрицы 𝑨−𝟏**,** если далее потребуется найти большое число векторов по формулам

𝒙(𝟏)***=*** 𝑨−𝟏 𝒅(𝟏)***,*** 𝒙(𝟐)***=*** 𝑨−𝟏 𝒅(𝟐)***,…,*** 𝒙(𝒑)***=*** 𝑨−𝟏 𝒅(𝒑)***.*** (5.42)

Однако суммарные затраты при таком подходе составят примерно 2𝑚3 + 2𝑝𝑚2 операций, в то время как при одновременном решении системы (5.41) методом Гаусса потребуется примерно (2/3)𝑚3 + 2𝑝𝑚2 операций. Следовательно, и в этом случае вычисление 𝑨−𝟏 нецелесообразно.

Довольно часто при решении различных задач средствами линейной алгебры возникают выражения типа

***v =*** 𝑩−𝟏***C***𝑨−𝟏***W***𝑫−𝟏***w .*** (5.43)

Если у исследователя нет достаточного опыта решения задач линейной алгебры на ЭВМ, то он может принять решение о необходимости вычислять матрицы 𝑩−𝟏***,***𝑨−𝟏, 𝑫−𝟏 с тем, чтобы действовать далее по формуле (5.43). Однако и в этом случае можно поступить иначе и найти вектор ***v*** с меньшими затратами. Решая систему ***Dx = w,*** найдем ***x =***

𝑫−𝟏***w.*** Затем вычислим ***y = Wx*** и, решая систему ***Az = y,*** найдем ***z=*** 𝑨−𝟏 ***y.***

Наконец, вычислим ***u = Cz*** и, решая систему ***Bv = u,*** найдем ***v =*** 𝑩−𝟏 ***u.***

Итак, во многих случаях вычисление обратной матрицы не требуется. Однако это вовсе не означает, что нет ситуаций, когда вычисление матрицы 𝑨−𝟏 необходимо и оправдано. В ряде технических приложений и статистических задач непосредственный интерес представляет анализ свойств именно обратной матрицы.

Покажем, как вычисление обратной матрицы можно свести к рассмотренной выше задаче решения системы уравнений с несколькими

3

правыми частями. Обозначим матрицу 𝑨−𝟏 через ***V***, ее столбцы через

𝒗1, 𝒗2, … 𝒗𝑚 и столбцы единичной матрицы через 𝒆1, 𝒆2, … 𝒆𝑚.

Согласно определению обратной матрицы верно равенство

***AV = E,*** эквивалентное совокупности равенств

***A***𝒗1 *=* 𝒆1, ***A***𝒗2 *=* 𝒆2, *…,* ***A***𝒗𝑚 *=* 𝒆𝑚. (5.44)

Таким образом, столбцы матрицы ***V =*** 𝑨−𝟏 (а следовательно, и саму матрицу) можно найти, решая *m* систем уравнений с общей матрицей ***A.*** Для этого потребовалось бы примерно (8/3) 𝑚3 арифметических операций, однако учет специального вида правых частей системы (5.44) позволяет вычислять матрицу 𝑨−𝟏 примерно за 2𝑚3 операций.

1. **Вычисление определителя.** Воспользуемся алгоритмом метода Гаусса с выбором главного элемента по столбцу и заметим, что искомый определитель и определитель полученной треугольной матрицы 𝑨(𝒎−𝟏) связаны равенством

det ***A*** = (-1)𝑠 det 𝑨(𝒎−𝟏)**,**

где *s* - число потребовавшихся перестановок строк. Остается воспользоваться формулой (5.24) и тогда получим

det ***A*** = (-1)𝑠 𝑎(0)𝑎(1)…𝑎(𝑚−1), (5.45)

11 22 𝑚𝑚

где 𝑎(0)= 𝑎11.

11

Рассмотрим далее другие методы решения систем линейных уравнений для других типов матриц.

4

# § 5.7. Метод Холецкого (метод квадратных корней).

1. **Описание метода.** Пусть требуется решить систему линейных алгебраических уравнений

***Ax = b*** (5.46)

С симметричной положительно определенной матрицей ***A.*** Линейные системы такого типа часто встречаются различных приложениях – в задачах оптимизации, при решении уравнений математической физики и т.п. Для их решения применяется *метод Холецкого (метод квадратных корней).*

В основе метода лежит алгоритм построения специального ***LU***- разложения матрицы ***A***, в результате чего она приводится к виду

***A = L***𝑳т***.*** (5.47)

В разложении (5.47) нижняя треугольная матрица

𝑙11 0 … 0

***L =*** [

𝑙21 𝑙22 … 0

. . . . . . . . . . . . . . . .

𝑙𝑚1 𝑙𝑚2 … 𝑙𝑚𝑚

] (5.48)

уже не обязательно должна иметь на главной диагонали единицы, как это было в методе Гаусса, а требуется только, чтобы диагональные элементы были положительными.

Если разложение (5.47) получено, то решение системы (5.46) сводится к последовательному решению двух систем с треугольными матрицами:

***Ly = b,*** 𝑳т***x = y.*** (5.49)

Для решения систем (5.49) требуется выполнение примерно 2𝑚2 арифметических операций.

Найдем элементы матрицы ***L.*** Для этого вычислим элементы матрицы ***L***𝑳т и приравняем их соответствующим элементам матрицы

***A***. В результате получим систему уравнений

5

𝑙2 = 𝑎11,

11

𝑙𝑖1𝑙11 = 𝑎𝑖1, *i = 2, 3, …,m,*

𝑙2 + 𝑙2 = 𝑎22,

21 22

𝑙𝑖1𝑙21 + 𝑙𝑖2𝑙22 = 𝑎𝑖2, *i = 3, 4, …, m,*

*. . . . . . . . . . . . . . . . . . . . . . . . . . . .* (5.50)

𝑙2 + 𝑙2 + . . . + 𝑙2 = 𝑎𝑘𝑘,

𝑘1

𝑘2

𝑘𝑘

𝑙𝑖1𝑙𝑘1 + 𝑙𝑖2𝑙𝑘2 + ⋯ + 𝑙𝑖𝑘𝑙𝑘𝑘 = 𝑎𝑖𝑘, *i = k+1, k+2, …, m,*

*. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .*

𝑙2 + 𝑙2 + . . . + 𝑙2 = 𝑎 .

𝑚1

𝑚2

𝑚𝑚

𝑚𝑚

Решая систему (5.50), последовательно находим

𝑙11 = √𝑎11 ,

𝑙𝑖1 = 𝑎𝑖1/ 𝑙11, *i = 2, 3, …,m,*

𝑙22 = √𝑎22 − 𝑙2 ,

𝑙𝑖2 = (𝑎𝑖2 − 𝑙𝑖1𝑙21)/ 𝑙22, *i = 3, 4, …, m,*

21

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . (5.51)

𝑙𝑘𝑘 = √𝑎𝑘𝑘 − 𝑙2 − 𝑙2 − . . . − 𝑙2

𝑘1

𝑘2

𝑘,𝑘−1

𝑙𝑖𝑘 = (𝑎𝑖𝑘 − 𝑙𝑖1𝑙𝑘1 − 𝑙𝑖2𝑙𝑘2 − ⋯ − 𝑙𝑖,𝑘−1𝑙𝑘,𝑘−1) / 𝑙𝑘𝑘, *i = k+1,…, m,*

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

𝑙𝑚𝑚 = √𝑎𝑚𝑚 − 𝑙2 − 𝑙2 − . . . − 𝑙2 .

𝑚1

𝑚2

𝑚,𝑚−1

Заметим, что для вычисления диагональных элементов используется операция извлечения квадратного корня. Поэтому метод Холецкого называют еще и методом квадратных корней. Доказано, что положительность соответствующих подкоренных выражений является следствием положительной определенности матрицы ***A.***

1. **Достоинства метода.** Метод Холецкого обладает рядом ценных качеств, которые позволяют предпочесть его методу Гаусса, если требуется решить систему линейных алгебраических уравнений с симметричной и положительно определенной матрицей.

Как нетрудно подсчитать, число операций, выполняемых в ходе вычисления разложения (5.47) по формулам (5.51), равно примерно

6

𝑚3/3. Учитывая, что для решения систем (5.49) требуется примерно 2𝑚2 арифметических операций, убеждаемся, что при больших *m* метод Холецкого требует вдвое меньше вычислительных затрат по сравнению с методом Гаусса.

Безусловным достоинством метода Холецкого является также его гарантированная устойчивость.

# § 5.8. Метод прогонки

Рассмотрим *метод прогонки* – простой и эффективный алгоритм решения систем линейных алгебраических уравнений с трехдиагональными матрицами:

𝑏1𝑥1 + 𝑐1𝑥2 = 𝑑1,

𝑎2𝑥1 + 𝑏2𝑥2 + 𝑐2𝑥3 = 𝑑2,

*………………….…………………………………..*

𝑎𝑖𝑥𝑖−1 + 𝑏𝑖𝑥𝑖 + 𝑐𝑖𝑥𝑖+1 = 𝑑𝑖*,* (5.52)

*………..……………………………………….*

𝑎𝑚−1𝑥𝑚−2 + 𝑏𝑚−1𝑥𝑚−1 + 𝑐𝑚−1𝑥𝑚 = 𝑑𝑚−1,

𝑎𝑚𝑥𝑚−1 + 𝑏𝑚𝑥𝑚 = 𝑑𝑚*.*

Системы такого вида часто возникают при решении различных задач математической физики, а также при решении других вычислительных задач (например, приближения функций сплайнами).

1. **Вывод расчетных формул.** Преобразуем первое уравнение системы (5.52) к виду

𝑥1 = 𝛼1𝑥2 + 𝛽1, где 𝛼1= − 𝑐1/ 𝑏1, 𝛽1 = 𝑑1/ 𝑏1. (5.53)

Подставим полученное для 𝑥1 выражение во второе уравнение системы:

𝑎2( 𝛼1𝑥2 + 𝛽1) + 𝑏2𝑥2 + 𝑐2𝑥3 = 𝑑2.

Преобразуем это уравнение к виду

7

𝑥2 = 𝛼2𝑥3 + 𝛽2, (5.54)

где 𝛼2= − 𝑐2/ (𝑏2 + 𝑎2 𝛼1), 𝛽2=(𝑑2 − 𝑎2 𝛽1) /(𝑏2 + 𝑎2 𝛼1).

Выражение (5.54) подставляем в третье уравнение системы и т.д.

На *i*-м шаге этого процесса (1< *i* <*m*) *i-*е уравнение системы преобразуется к виду

𝑥𝑖 = 𝛼𝑖𝑥𝑖+1 + 𝛽𝑖, (5.55)

где 𝛼𝑖= − 𝑐𝑖/ (𝑏𝑖 + 𝑎𝑖 𝛼𝑖−1), 𝛽𝑖=(𝑑𝑖 − 𝑎𝑖 𝛽𝑖−1) /(𝑏𝑖 + 𝑎𝑖 𝛼𝑖−1).

На *m*-м шаге подстановка в последнее уравнение выражения

𝑥𝑚−1 = 𝛼𝑚−1𝑥𝑚 + 𝛽𝑚−1 дает

𝑎𝑚( 𝛼𝑚−1𝑥𝑚 + 𝛽𝑚−1) + 𝑏𝑚𝑥𝑚 = 𝑑𝑚.

Отсюда можно определить значение 𝑥𝑚:

𝑥𝑚 = 𝛽𝑚 = (𝑑𝑚 − 𝑎𝑚 𝛽𝑚−1) /(𝑏𝑚 + 𝑎𝑚 𝛼𝑚−1).

Значения остальных неизвестных 𝑥𝑖 для *i = m-*1*, m-*2, …, 1 теперь легко вычисляются по формуле (5.55).

1. **Алгоритм прогонки.** Сделанные преобразования позволяют организовать вычисления метода прогонки в два этапа.

*Прямой ход* метода прогонки (*прямая прогонка*) состоит в вычислении *прогоночных коэффициентов*

𝛼𝑖 (1 ≤ 𝑖 < 𝑚) и 𝛽𝑖 (1 ≤ 𝑖 ≤ 𝑚).

При *i = 1* коэффициенты вычисляются по формулам

𝛼1= − 𝑐1/𝛾1, 𝛽1 = 𝑑1/𝛾1, 𝛾1= 𝑏1, (5.56) а при 𝑖 = 2, 3, … , 𝑚 − 1 – по рекуррентным формулам

𝛼𝑖= − 𝑐𝑖/ 𝛾𝑖 , 𝛽𝑖=(𝑑𝑖 − 𝑎𝑖 𝛽𝑖−1) / 𝛾𝑖 , 𝛾𝑖 = 𝑏𝑖 + 𝑎𝑖 𝛼𝑖−1. (5.57) При *i = m* прямая прогонка завершается вычислением

8

𝛽𝑚= (𝑑𝑚 − 𝑎𝑚 𝛽𝑚−1) / 𝛾𝑚 , 𝛾𝑚 = 𝑏𝑚 + 𝑎𝑚 𝛼𝑚−1. (5.58)

Обратный ход метода прогонки (*обратная прогонка*) дает значения неизвестных. Сначала полагают 𝑥𝑚 = 𝛽𝑚. Затем значения остальных неизвестных вычисляют по формуле

𝑥𝑖 = 𝛼𝑖𝑥𝑖+1+𝛽𝑖, *i = m*−1*, m*−2, …, 1*.* (5.59) Вычисления ведут в порядке убывания значений *i* от *m*−1 до 1*.*

1. **Свойства метода прогонки.** Непосредственный подсчет

показывает, что для реализации вычислений по формулам (5.56) – (5.59) для систем уравнений с трехдиагональной матрицей требуется примерно 8 *m* арифметических операций, тогда как в методе Гаусса (для систем с заполненной матрицей) это число составляет примерно (2/3)𝑚3.

Приведем достаточные условия на коэффициенты системы (5.52), при выполнении которых вычисления по формулам прямой прогонки могут быть доведены до конца (ни один из знаменателей 𝛾𝑖 не обратится в нуль). В частности, это гарантирует существование и единственность решения (5.52). При выполнении тех же условий коэффициенты 𝛼𝑖 при всех *i* удовлетворяют неравенству |𝛼𝑖| ≤ 1, а следовательно, обратная прогонка по формуле (5.59) устойчива по входным данным.

Т е о р е м а 5.2. *Пусть коэффициенты системы* (5.52)

*удовлетворяют следующим условиям диагонального преобладания:*

*|*𝑏𝑘 *| ≥ |*𝑎𝑘 *| + |*𝑐𝑘 *|, |*𝑏𝑘 *|>|*𝑎𝑘*| (1≤ k ≤ m).* (5.60)

*Тогда* 𝛾𝑖 ≠ 0 *и* |𝛼𝑖| ≤ 1 *для всех i =* 1*,* 2, …, *m.*

□ Теорема доказывается методом математической индукции ■

**Лекция 10**

**Тема 6. Приближение функций**

**§ 6.1. Постановка задачи приближения функций**

Вычисление значения функции *y = f(x)* – одна из тех задач, с которой на практике приходится сталкиваться. Естественно, что при решении на ЭВМ серьезных задач желательно иметь быстрые и надежные алгоритмы вычисления значений используемых функций. Для элементарных, а также основных специальных функций такие алгоритмы разработаны, реализованы в виде стандартных программ и включены в математическое обеспечение ЭВМ. Однако в расчетах нередко используются и другие функции, непосредственное вычисление которых затруднено либо приводит к слишком большим затратам машинного времени. Укажем на некоторые типичные ситуации.

* 1. Функция *f* задана таблицей своих значений:

𝑦𝑖 = *f* (𝑥𝑖), (*i* = 0, 1, 2, … , *n*), (6.1)

а вычисления проводятся в точках 𝑥, не совпадающих с табличными.

* 1. Непосредственное вычисление значения *y = f (x)* связано с проведением сложных расчетов и приводит к значительным затратам машинного времени, которые могут оказаться неприемлемыми, если функция *f* вычисляется многократно.
  2. При заданном значении *x* значение *f (x)* может быть найденным из эксперимента. В большинстве случаев нахождение значения функции из эксперимента в реальном масштабе времени невозможно. В этой ситуации экспериментальные данные получают до начала вычислений. Нередко они представляют таблицу вида (6.1) с тем отличием, что табличные значения 𝑦∗ отличаются от истинных значений 𝑦𝑖, так как заведомо содержат ошибки эксперимента.

𝑖

Возникающие проблемы нередко удается решить следующим образом. Функцию *f (x)* приближенно заменяют другой функцией *g (x),* вычисляемые значения которой и принимают за приближенное значение функции *f.* Конечно, такая замена оправдана лишь тогда, когда значения *g (x)* вычисляются быстро и надежно, а погрешность приближения *f (x)* – *g (x)* достаточно мала. Обсудим кратко некоторые вопросы, с которыми в каждом конкретном случае приходится сталкиваться при выборе постановки задачи приближения и метода ее решения.

1. Необходимо решить, какую информацию о функции *f* можно использовать как входные данные для вычисления приближения *g*. Например, часто известна или может быть получена таблица значений функции вида (6.1), а иногда – и таблица ее производных. В некоторых случаях можно использовать информацию о значениях функции на всем отрезке [*a, b* ].
2. Полезно иметь некоторую дополнительную априорную информацию об аппроксимируемой функции. Часто она бывает качественного характера, например, известно, что функция *f* «достаточно гладкая» («плавно меняющаяся»), периодическая, монотонная, четная и т.п. Иногда удается получить некоторые количественные характеристики функции *f*, например, бывают известны верхние оценки для максимума модуля некоторых ее производных, величина периода, оценка уровня погрешности в заданных значениях.
3. Знание свойств функции *f* позволяет осознанно выбирать класс *G* аппроксимирующих функций. Часто такой класс представляет собой параметрическое семейство функций вида *y =g (x,* ***a****) = g (x,*𝑎0*,* 𝑎1*, … ,* 𝑎𝑚) и выбор конкретной аппроксимирующей функции *g* осуществляется с помощью параметров 𝑎0*,* 𝑎1*, … ,* 𝑎𝑚. Широко используются классы функций вида

Ф𝑚(*x*) = 𝑎0𝜑0(*x*) + 𝑎1𝜑1(*x*) + … +𝑎𝑚𝜑𝑚(*x*), (6.2)

являющихся линейными комбинациями фиксированного набора некоторых базисных функций 𝜑0(*x*), 𝜑1(*x*), … , 𝜑𝑚(*x*). Функцию Ф𝑚(*x*) часто называют *обобщенным многочленом* по системе функций 𝜑0(*x*), 𝜑1(*x*), … , 𝜑𝑚(*x*), а число *m* – его *степенью*.

Если в качестве базисных функций берутся степенные функции

𝜑𝑘(*x*) = 𝑥𝑘 , то возникает задача приближения алгебраическими многочленами

𝑃𝑚(*x*) = 𝑎0 + 𝑎1*x* + … +𝑎𝑚𝑥𝑚. (6.3)

Отметим, что методы приближения функций алгебраическими многочленам играют важную роль в численном анализе и наиболее глубоко разработаны. Одна из причин этого состоит в том, что многочлены (6.3) легко вычисляются, без труда дифференцируются и интегрируются.

Тригонометрические многочлены

𝑆𝑚(*x*) = 𝑎0 + ∑1≤𝑘≤𝑚/2(𝛼𝑘 cos 2𝜋𝑘𝑥 + 𝛽𝑘 sin 2𝜋𝑘𝑥), (6.4)

часто используемые для аппроксимации периодических на отрезке [0, 1] функций, также могут быть записаны в виде (6.2), если в качестве базисных функций выбрать функции 𝜑0(*x*) = 1, 𝜑1(*x*) = cos 2𝜋𝑥, 𝜑2(*x*) = sin 2𝜋𝑥,

𝜑3(*x*) = cos 4𝜋𝑥, 𝜑4(*x*) = sin 4𝜋𝑥, … .

Используя формулу Эйлера exp {*iy*} = cos *y* + *i* sin *y*, можно записать тригонометрический многочлен (6.4) в виде

𝑆𝑚(*x*) = ∑−𝑚/2≤𝑘≤𝑚/2 exp{2𝜋𝑖𝑘𝑥}, (6.5) что соответствует выбору базисных функций 𝜑𝑘 (*x*) =exp {2𝜋𝑖𝑘𝑥},

−𝑚/2 ≤ 𝑘 ≤ 𝑚/2.

1. Необходим критерий выбора в классе *G* конкретной функции *g*, являющейся в этом классе наилучшим приближением к *f*. Например, требование совпадения функции *g* с функцией *f* в некоторых фиксированных точках приводит к задаче интерполяции. Другой распространенный критерий – требование минимизации среднеквадратического отклонения – лежит в основе метода наименьших квадратов.
2. Важно понимать, что решение указанных выше вопросов тесно связано с тем, как мы собираемся использовать приближение *g* и какая точность нам нужна.

**§ 6.2. Интерполяция обобщенными многочленами.**

1. **Постановка задачи интерполяции.** Пусть в точках 𝑥0, 𝑥1, … , 𝑥𝑛, расположенных на отрезке [*a, b*] и попарно различных, задана таблица (6.1) значений некоторой функции *f. Задача интерполяции* состоит в построении функции *g,* удовлетворяющей условию

*g* (𝑥𝑖) = 𝑦𝑖 (*i* = 0, 1, …, *n*). (6.6)

Другими словами, ставится задача построения функции *g,* график которой проходит через заданные точки (𝑥𝑖, 𝑦𝑖). Указанный способ приближения функций принято называть *интерполяцией* (или *интерполированием*), а точки

𝑥𝑖 – *узлами интерполяции*.

Нетрудно видеть, что выбор функции *g* неоднозначен, так как по заданной таблице можно построить бесконечно много интерполирующих функций. На практике, как правило, функцию *g* выбирают из достаточно узкого класса *G* функций, в котором единственность выбора гарантируется.

1. **Экстраполяция.** Пусть 𝑥𝑚𝑖𝑛 и 𝑥𝑚𝑎𝑥 – минимальный и максимальный из узлов интерполяции. В случае, когда интерполяция используется для вычисления приближенного значения функции *f* в точке *x,* не принадлежащей отрезку [𝑥𝑚𝑖𝑛, 𝑥𝑚𝑎𝑥] *(отрезку наблюдения),* принято говорить о том, что осуществляется *экстраполяция*. Этот метод приближения часто используют с целью прогнозирования характера протекания тех или иных процессов при значениях параметров *x*, выходящих за пределы отрезка наблюдения. Заметим, что надежность такого прогноза при значениях *x,* удаленных на значительное расстояние от отрезка [𝑥𝑚𝑖𝑛, 𝑥𝑚𝑎𝑥], как правило, невелика.
2. **Задача интерполяции обобщенными многочленами.** Рассмотрим более подробно задачу интерполяции обобщенными многочленами Ф𝑚(*x*) вида (6.2). Назовем обобщенный многочлен Ф𝑚(*x*) *интерполяционным*, если он удовлетворяет условию

Ф𝑚(𝑥𝑖) = 𝑦𝑖 (*i=* 0, 1, 2, … , *n*), (6.7) или, что то же самое, системе линейных алгебраических уравнений

𝜑0(𝑥0)𝑎0 + 𝜑1(𝑥0)𝑎1 + … +𝜑𝑚(𝑥0)𝑎𝑚 = 𝑦0,

𝜑0(𝑥1)𝑎0 + 𝜑1(𝑥1)𝑎1 + … +𝜑𝑚(𝑥1)𝑎𝑚 = 𝑦1,

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . (6.8)

𝜑0(𝑥𝑛)𝑎0 + 𝜑1(𝑥𝑛)𝑎1 + … +𝜑𝑚(𝑥𝑛)𝑎𝑚 = 𝑦𝑛

относительно коэффициентов 𝑎0, 𝑎1, … , 𝑎𝑚.

Заметим, что систему уравнений (6.7) можно записать в следующем виде:

***Pa = y,*** (6.9)

где

𝜑0(𝑥0) 𝜑1(𝑥0) … 𝜑𝑚(𝑥0)

𝑎0

𝑦0

***P =*** [ 𝜑0(𝑥1) 𝜑1(𝑥1) … 𝜑𝑚(𝑥1) ], ***a =***[ 𝑎1 ], ***y =***[ 𝑦1]. (6.10)

. . . . . . . . . . . . . . . . . . . . .

𝜑0(𝑥𝑛) 𝜑1(𝑥𝑛) … 𝜑𝑚(𝑥𝑛)

. . .

𝑎𝑚

. . .

𝑦𝑛

Введем векторы 𝝋𝒋 = (𝜑𝑗(𝑥0), 𝜑𝑗(𝑥1), … , 𝜑𝑗(𝑥𝑛))т, *j = 0, 1, … ,m*. Будем говорить, что *система функций* 𝜑0(𝑥), 𝜑1(𝑥), … 𝜑𝑚(𝑥) *линейно зависима в точках* 𝑥0, 𝑥1, … , 𝑥𝑛*,* если один из векторов

𝝋𝒋 системы 𝝋𝟎, 𝝋𝟏, … , 𝝋𝒎 может быть представлен в виде линейной

комбинации остальных векторов этой системы:

𝒎

𝝋𝒋 **=** ∑

𝑘=0,𝑘≠𝑗

𝑎𝒌 𝝋𝒌. (6.11)

В противном случае систему функций 𝜑0(𝑥), 𝜑1(𝑥), … 𝜑𝑚(𝑥) будем называть *линейно независимой в точках* 𝑥0, 𝑥1, … , 𝑥𝑛.

**У т в е р ж д е н и е 6.1.** При *m ≤ n* система функций 1, *x*, 𝑥2, …,

𝑥𝑚 линейно независима в точках 𝑥0, 𝑥1, … , 𝑥𝑛, если они попарно различны.

□ Допустим противное. Тогда справедливо равенство (6.11), которое в данном случае принимает вид

𝑥𝑗 **=** ∑𝑚 𝛼𝑘 𝑥𝑘, (*i* = 0, 1, … , *n*) (6.12)

𝑖 𝑘=0,𝑘≠𝑗 𝑖

Полагая 𝛼𝑗 = – 1, получаем, что многочлен 𝑃𝑚(𝑥) = ∑𝑚 𝛼𝑘 𝑥𝑘

𝑘=0

степени *m* обращается в нуль в точках 𝑥0, 𝑥1, … , 𝑥𝑛, число которых равно (*n* + 1) и, следовательно, больше *m.* Однако в силу основной теоремы алгебры многочлен степени *m*, тождественно не равный нулю, не может иметь более *m* корней. Полученное противоречие доказывает независимость рассматриваемой системы функций. ■

Рассмотрим *матрицу Грама* системы функций 𝜑0, 𝜑1, … , 𝜑𝑚, имеющую вид

*Г =* 𝑃∗𝑃 *=*[

(𝜑0 , 𝜑0) (𝜑1 , 𝜑0) … (𝜑𝑚 , 𝜑0)

(𝜑0 , 𝜑1) (𝜑1 , 𝜑1) … (𝜑𝑚 , 𝜑1)

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

]. (6.13)

(𝜑0 , 𝜑𝑚) (𝜑1 , 𝜑𝑚) … (𝜑𝑚 , 𝜑𝑚)

Здесь в случае, когда функции 𝜑𝑗(*x*) могут принимать комплексные значения, под 𝑃∗ понимается сопряженная к 𝑃 матрица, а элементы 𝛾𝑗𝑘 матрицы Грама вычисляются по формуле

𝛾𝑗𝑘 = (𝜑𝑘 , 𝜑𝑗) = ∑𝑛 𝜑𝑘 (𝑥𝑖)̅𝜑̅̅𝑦̅(̅𝑥̅̅𝑖̅), (6.14)

𝑖=0

Если же функции 𝜑𝑗(𝑥) принимают только вещественные значения, то

𝑃∗= 𝑃т и элементы матрицы Грама вычисляются по формуле

𝛾𝑗𝑘 = (𝜑𝑘 , 𝜑𝑗) = ∑𝑛 𝜑𝑘 (𝑥𝑖)𝜑𝑗(𝑥𝑖). (6.15)

𝑖=0

Определитель матрицы Грама det *Г* принято называть *определителем Грама*. Как следует из курса линейной алгебры, справедлив следующий результат.

Т е о р е м а 6.1. *Система функций* 𝜑0(𝑥), 𝜑1(𝑥), … 𝜑𝑛(𝑥) *является линейно независимой в точках* 𝑥0, 𝑥1, … , 𝑥𝑛 *тогда и только тогда, когда m ≤ n и определитель Грама* det *Г отличен от нуля.*

Известно, что при *m > n* система функций 𝜑0(𝑥), 𝜑1(𝑥), … 𝜑𝑚(𝑥) линейно зависима в точках 𝑥0, 𝑥1, … , 𝑥𝑛. Отсюда вытекает неединственность решения ***a*** системы (6.9) (если оно существует). Действительно, в этом случае справедливо представление (6.11) и вместе с вектором ***a*** решением системы (6.9) является вектор 𝒂′ = ***a +*** *t* Δ***a,*** где Δ***a =*** (𝛼0, 𝛼0, …,𝛼0, -1, 𝛼0, … 𝛼0)т, а *t –* любое число. Если же *m < n,* то решение системы (6.9) существует не для всякой правой части

1. В силу указанных причин при интерполяции обобщенными многочленами число параметров *m+1* обычно берут равным числу *n+1* заданных точек. В этом случае ***Р*** *–* квадратная матрица и для того, чтобы система (6.9) была однозначно разрешима при любой правой части ***y,*** необходимо и достаточно, чтобы определитель матрицы ***Р*** был отличен от нуля. В свою очередь при *m = n* это условие в силу равенства det *Г* = det 𝑷∗ det ***Р*** и теоремы 6.1 дает следующий результат. Т е о р е м а 6.2. *Если m = n, то решение задачи интерполяции обобщенным многочленом* (6.2) *существует и единственно при любом наборе данных* 𝑦0, 𝑦1, … , 𝑦𝑛 *тогда и только тогда, когда система*

*функций* 𝜑0, 𝜑1, … 𝜑𝑛 *линейно независима в точках* 𝑥0, 𝑥1, … , 𝑥𝑛*.*

Назовем систему функций 𝜑0(𝑥), 𝜑1(𝑥), … 𝜑𝑚(𝑥) *ортогональной на множестве точек* 𝑥0, 𝑥1, … , 𝑥𝑛, если (𝝋𝒌 , 𝝋𝒋) = 0 при *k ≠ j* и (𝝋𝒌 , 𝝋𝒋) *≠* 0 при *k* = *j* для всех *k = 0, 1, … ,m*; *j = 0, 1, … ,m*. Очевидно, что для ортогональной на множестве точек 𝑥0, 𝑥1, … , 𝑥𝑛 системы функций матрица Грама диагональна, а определитель Грама

отличен от нуля. Поэтому всякая ортогональная на множестве точек

𝑥0, 𝑥1, … , 𝑥𝑛 система функций заведомо является линейно независимой в этих точках.

**У т в е р ж д е н и е 6.2.** Система функций 𝜑0(𝑥), 𝜑1(𝑥), … 𝜑𝑁−1(𝑥), где 𝜑𝑘(𝑥) = exp{2π*ikx*}, ортогональна на множестве точек 𝑥𝑙 *=* 𝑙⁄𝑁*,* где

*l = 0, 1,…, N –1.* Здесь *i* – мнимая единица.

□ Для доказательства ортогональности рассматриваемой системы функций достаточно установить справедливость равенства

(𝜑𝑘, 𝜑𝑗) = *N*𝛿𝑘𝑗 (*k = 0, 1, …, N – 1; j = 0, 1, …, N – 1*), (6.16)

где 𝛿𝑘𝑗 *=* 0 при *k ≠ j* и 𝛿𝑘𝑗 *=* 1 при *k = j.* Введем обозначение ω = exp {2π*i/N*}. Тогда 𝜑𝑘(𝑥𝑙) = exp {2π*ikl/N*} = ω𝑘𝑙 и согласно формуле (6.14) имеем

(𝜑𝑘, 𝜑𝑗) = ∑𝑁−1 ω𝑘𝑙ω−𝑗𝑙 = ∑𝑁−1 ω(𝑘−𝑗)𝑙. (6.17)

𝑖=0 𝑖=0

При *k = j* правая часть равенства (6.17), очевидно, равна *N*. При *k ≠ j,* используя формулу суммы членов геометрической прогрессии и равенство ω(𝑘−𝑗)𝑁 *=* exp {2π*i(k-j)*} = 1, имеем

(𝜑𝑘, 𝜑𝑗) = (1 − ω(𝑘−𝑗)𝑁)/(1 − ω𝑘−𝑗) = 0.

Таким образом, равенство (6.16), а вместе с ним и ортогональность системы функций 𝜑𝑘(𝑥) = exp{2π*ikx*} доказаны. ■

В случае, когда система функций 𝜑0, 𝜑1, … 𝜑𝑛 ортогональна на множестве точек 𝑥0, 𝑥1, … , 𝑥𝑛*,* решение задачи интерполяции не представляет затруднений. Действительно, система уравнений (6.9) после умножения на матрицу 𝑷∗ преобразуется к виду

***Га = b, b =*** 𝑃∗***y.*** (6.18)

Заметим, что элементы вектора ***b*** = (𝑏0, 𝑏1, … , 𝑏𝑚)т вычисляются по формуле

𝑏𝑗 = (***y,*** 𝝋𝒋) *=* ∑𝑛 𝑦𝑙 𝜑̅̅𝑦̅(̅𝑥̅̅̅*, j = 0, 1, ... ,m.* (6.19)

𝑙=0

𝑙 )

Так как матрица ***Г*** диагональна, то решение системы (6.18) находится в явном виде:

𝑎𝑗

= (𝒚,𝝋𝒋) , *j* = 0, 1, … , *m.* (6.20)

(𝝋𝒋,𝝋𝒋)

# Лекция 11

**§ 6.3. Полиномиальная интерполяция. Многочлен Лагранжа**

* 1. **Интерполяционный многочлен.** Начнем с рассмотрения задачи интерполяции в наиболее простом и полно исследованном случае интерполирования алгебраическими многочленами. Для

заданной таблицы (6.1) многочлен 𝑃𝑛(*x*)*=* ∑𝑛 𝑎𝑘 𝑥𝑘 степени *n*

𝑘=0

называется *интерполяционным многочленом*, если он удовлетворяет условиям

𝑃𝑛(𝑥𝑖) = 𝑦𝑖 (*i = 0, 1, … , n*). (6.21)

Равенство (6.21) можно записать аналогично (6.8) в виде системы уравнений

𝑎0 *+*𝑎1𝑥0 *+*𝑎2𝑥2 *+ … +*𝑎𝑛𝑥𝑛 *=* 𝑦0

0 0

𝑎0 *+*𝑎1𝑥1 *+*𝑎2𝑥2 *+ … +*𝑎𝑛𝑥𝑛 *=* 𝑦1

1 1

*. . . . . . . . . . . . . . . . . . . . . . . . . . . .* (6.22)

𝑛 𝑛

𝑎0 *+*𝑎1𝑥𝑛 *+*𝑎2

𝑛

𝑥2 *+ … +*𝑎𝑛

𝑥𝑛 *=* 𝑦

относительно коэффициентов многочлена. Эта система однозначно разрешима, так как система функций 1, *x,* 𝑥2*, … ,* 𝑥𝑛 линейно независима в точках 𝑥0, 𝑥1, … , 𝑥𝑛 (см. утверждение 6.1 и теорему 6.1). Таким образом, верна следующая теорема.

Т е о р е м а 6.2. *Существует единственный интерполяционный многочлен степени n, удовлетворяющий условиям* (6.21).

З а м е ч а н и е. На практике система (6.22) никогда не используется для вычисления коэффициентов интерполяционного многочлена. Дело в том, что она является плохо обусловленной. Кроме того существуют различные явные формы записи интерполяционного многочлена, которые применяются при интерполяции. Наконец, в большинстве приложений интерполяционного многочлена явное вычисление коэффициентов 𝑎𝑘 не нужно.

2

* 1. **Многочлен Лагранжа.** Приведем одну из форм записи интерполяционного многочлена – *многочлен Лагранжа*

𝐿𝑛*(x) =* ∑𝑛 𝑦𝑗 𝑙𝑛𝑗 (𝑥). (6.23)

𝑗=0

Здесь 𝑙

(𝑥) = ∏𝑛

𝑥−𝑥𝑘 = (𝑥− 𝑥0)(𝑥− 𝑥1)…(𝑥− 𝑥𝑗−1)(𝑥−𝑥𝑗+1)…(𝑥−𝑥𝑛) .

𝑛𝑗

𝑘=1,𝑘≠𝑗 𝑥𝑗−𝑥𝑘

(𝑥𝑗− 𝑥0)(𝑥𝑗− 𝑥1)…(𝑥𝑗− 𝑥𝑗−1)(𝑥𝑗−𝑥𝑗+1)…(𝑥𝑗−𝑥𝑛)

Как нетрудно видеть, 𝑙𝑛𝑗(𝑥) представляет собой многочлен степени *n*, удовлетворяющий условию

𝑙𝑛𝑗(𝑥𝑖) = {1, при 𝑖 = 𝑗

0, при 𝑖 ≠ 𝑗

Таким образом, степень многочлена 𝐿𝑛 равна *n* и при *x =* 𝑥𝑖 в сумме (6.22) обращаются в ноль все слагаемые, кроме слагаемого с номером *j = i*, равного 𝑦𝑖. Поэтому многочлен Лагранжа (6.23) действительно является интерполяционным.

Заметим, что на практике интерполяционный многочлен Лагранжа используется так, что нет необходимости его преобразования к

каноническому виду 𝐿𝑛*(x) =* ∑𝑛 𝑎𝑘 𝑥𝑘.

𝑗=0

Приведем формулы для записи многочленов Лагранжа первой и второй степени, которые часто используются на практике:

𝐿 *(x) =* 𝑦 𝑥− 𝑥1

1 0

𝑥0−𝑥1

*+*𝑦 𝑥− 𝑥0 *,* (6.24)

𝑥1−𝑥0

1

𝐿2*(x) =* 𝑦0 (𝑥 − 𝑥1)(𝑥 − 𝑥2) *+*𝑦1 (𝑥 − 𝑥0)(𝑥 − 𝑥2) *+*𝑦2 (𝑥 − 𝑥0)(𝑥 − 𝑥1) *.* (6.25)

(𝑥0−𝑥1)(𝑥0−𝑥2)

(𝑥1−𝑥0)(𝑥1−𝑥2)

(𝑥2−𝑥0)(𝑥2−𝑥1)

3

# § 6.4. Погрешность интерполяции

Приведем без доказательства наиболее известную теорему о погрешности интерполяции.

**Т е о р е м а 6.3.** *Пусть функция f дифференцируема n + 1 разна отрезке* [*a, b*], *содержащем узлы интерполяции* 𝑥𝑖*, i = 0, 1, … , n. Тогда для погрешности интерполяции в точке x ϵ* [*a, b*] *справедливо равенство*

*f ( x )* − 𝑃 *( x ) =* 𝑓(𝑛+1)( 𝜉) 𝜔

( 𝑥), (6.26)

𝑛 (𝑛+1)!

𝑛+1

*в котором* 𝜔𝑛+1( 𝑥) *= (x* − 𝑥0) *(x* − 𝑥1) … (𝑥 − 𝑥𝑛)*, а* 𝜉 *– некоторая точка, принадлежащая интервалу (a, b ).*

Основное неудобство в использовании этой теоремы состоит в том, что входящая в формулу (6.26) для погрешности точка 𝜉 неизвестна. Поэтому чаще используется не сама теорема, а ее следствие.

С л е д с т в и е. *В условиях теоремы (6.3) справедлива оценка погрешности интерполяции в точке x ϵ* [*a, b*], *имеющая вид*

| *f ( x )* − 𝑃 *( x )* | *≤* 𝑀𝑛+1

𝑛

(𝑛+1)!

|𝜔𝑛+1( 𝑥)|, (6.27)

*а также оценка максимума модуля погрешности интерполяции на отрезке* [*a, b*], *имеющая вид*

max| *f ( x )* − 𝑃𝑛*( x )* | *≤*𝑀𝑛+1

max |𝜔𝑛+1( 𝑥)|, (6.28)

[𝑎,𝑏]

*Здесь* 𝑀 = max

𝑛+1

[𝑎,𝑏]

(𝑛+1)!

|𝑓(𝑛+1)( 𝑥)|.

[𝑎,𝑏]

Пусть теперь 𝑥0 < 𝑥1 < ⋯ < 𝑥𝑛 и пусть ℎ𝑖 = 𝑥𝑖 − 𝑥𝑖−1 – *i*-й шаг таблицы, а ℎ𝑚𝑎𝑥 = max ℎ𝑖. Несколько огрубляя оценку (6.28),

1≤𝑖≤𝑛

можно получить следующее неравенство:

max | *f ( x)* − 𝑃𝑛*( x )* | *≤*𝑀𝑛+1

ℎ𝑛+1 . (6.29)

[𝑥0,𝑥𝑛]

4(𝑛+1)

𝑚𝑎𝑥

4

Оно позволяет утверждать, что для достаточно гладкой функции *f* при фиксированной степени интерполяционного многочлена погрешность интерполяции на отрезке [𝑥0, 𝑥𝑛] при ℎ𝑚𝑎𝑥 → 0 стремится к нулю не медленнее, чем некоторая величина, пропорциональная ℎ𝑛+1 . Этот факт принято формулировать так: интерполяция многочленом степени *n* имеет (*n+1*)−й порядок точности относительно ℎ𝑚𝑎𝑥. В частности, линейная и квадратичная интерполяции имеют второй и третий порядок точности соответственно.

𝑚𝑎𝑥

# § 6.5. Интерполяция с кратными узлами

1. **Интерполяционный многочлен с кратными узлами.** Иногда в узлах 𝑥𝑖 (*i = 0, 1, … , m*) бывают заданы не только значения 𝑦𝑖 = *f* (𝑥𝑖)

функции *f,* но и значения ее производных 𝑦′ = 𝑓′(𝑥𝑖), 𝑦′′ = 𝑓′′(𝑥𝑖), … ,

𝑖 𝑖

𝑦(𝑘𝑖−1) = 𝑓(𝑘𝑖−1)(𝑥𝑖) до некоторого порядка 𝑘𝑖 − 1. В этом случае узел

𝑖

𝑥𝑖 называют *кратным*, а число 𝑘𝑖, равное количеству заданных значений, − *кратностью узла*. Пусть *n* = 𝑘0 + 𝑘1 + ⋯ + 𝑘𝑚 − 1. Можно доказать, что существует единственный многочлен 𝑃𝑛*(x)* степени *n*, удовлетворяющий условиям

𝑃𝑛*(*𝑥𝑖*) =* 𝑦𝑖*,* 𝑃𝑛′(𝑥𝑖) = 𝑦′, … , 𝑃𝑛(𝑘𝑖−1)(𝑥𝑖) = 𝑦(𝑘𝑖−1) для *i=0, 1, … , m.*

𝑖 𝑖

Этот многочлен называют *интерполяционным многочленом с кратными узлами*. Можно указать и явную формулу его записи, аналогичную форме Лагранжа. Мы этого делать не будем, а отметим лишь два важных частных случая.

1). Пусть на концах отрезка [𝑥0, 𝑥1] заданы значения 𝑦0, 𝑦1, 𝑦′ , 𝑦′.

0 1

Тогда *m = 1,* 𝑘0 *= 2,* 𝑘1 *= 2, n = 3* и интерполяционный многочлен 𝑃3*(x),* удовлетворяющий условиям 𝑃3(𝑥0) = 𝑦0, 𝑃3(𝑥1) = 𝑦1, 𝑃3′(𝑥0) = 𝑦′ ,

0

𝑃3′(𝑥1) = 𝑦′, может быть представлен в следующем виде:

1

𝑃 *(x) =* 𝑦

(𝑥1−𝑥)2(2(𝑥− 𝑥0)+ℎ) *+* 𝑦′ (𝑥1−𝑥)2(𝑥− 𝑥0) *+*

3 0 ℎ3 0 ℎ2

+ 𝑦

(𝑥−𝑥0 )2(2(𝑥1−𝑥)+ℎ)

*+* 𝑦′ (𝑥−𝑥0)2(𝑥− 𝑥1)*,* где *h* = 𝑥

− 𝑥

. (6.30)

1 ℎ3 1 ℎ2 1 0

5

Многочлен (6.30) принято называть *кубическим интерполяционным многочленом Эрмита.*

2). Пусть в точке 𝑥0 заданы значения 𝑦0, 𝑦′ , … , 𝑦𝑛. Тогда

0 0

многочлен 𝑃𝑛*(x),* удовлетворяющий условиям

𝑃𝑛*(*𝑥0*) =* 𝑦0, 𝑃𝑛′(𝑥0) = 𝑦′ , … , 𝑃𝑛(𝑛)(𝑥0) = 𝑦(𝑛),

0 0

представляется в виде:

𝑃 *(x) =* ∑𝑛 𝑦(𝑘) (𝑥−𝑥0 )𝑘*.* (6.31)

𝑛 𝑘=0 0

𝑘!

Как нетрудно видеть, многочлен 𝑃𝑛*(x)* представляет собой отрезок ряда Тейлора. Таким образом, формула Тейлора дает решение задачи интерполяции с одним узлом кратности (*n* + 1). Заметим, что в действительности с ее помощью осуществляется экстраполяция.

# Погрешность интерполяции с кратными узлами.

**Т е о р е м а 6.4.** *Пусть функция f дифференцируема n + 1 раз на отрезке* [*a, b*], *содержащем узлы интерполяции* 𝑥𝑖 (*i = 0, 1, … , m*). *Тогда для погрешности интерполяции с кратными узлами в точке x ϵ* [*a, b*] *справедливы равенство* (6.26) *и неравенства* (6.27) *и* (6.28), *в которых* 𝜔𝑛+1( 𝑥) *= (x* − 𝑥0)𝑘0 *(x* − 𝑥1)𝑘1 … (𝑥 − 𝑥𝑚)𝑘𝑚 *, а* 𝜉 *– некоторая точка, принадлежащая интервалу (a, b ).*

Для формулы Тейлора (*m = 0,* 𝑘0 *= n + 1*) теорема 6.4 дает известную формулу остаточного члена в форме Лагранжа. Для кубического многочлена Эрмита (*m = 0,* 𝑘0 *= 2,* 𝑘1 *= 2*) неравенство (6.29) приводит к следующей оценке погрешности:

max | *f ( x)* − 𝑃3*( x )* | *≤*𝑀4

ℎ4. (6.32)

[𝑥0,𝑥1]

384

Здесь учтено то, что максимум функции 𝜔4( 𝑥) *= (x* − 𝑥0)2 *(x* − 𝑥1)2

на отрезке [𝑥0, 𝑥1] достигается в точке *x* = ( 𝑥0 + 𝑥1) / 2 и равен ℎ4/16.

6

# § 6.6. Минимизация оценки погрешности интерполяции. Многочлены Чебышева.

1. **Постановка задачи минимизации оценки погрешности.** Предположим, что значение заданной на отрезке [𝑎, 𝑏] функции *f* можно вычислить в произвольной точке *x.* Однако по некоторым причинам (например, вычисление значений *f(x) –* трудоемкая операция) целесообразнее заменить прямое вычисление функции *f* вычислением значений ее интерполяционного многочлена 𝑃𝑛. Для такой замены необходимо один раз получить таблицу значений функции *f* в выбранных на отрезке [𝑎, 𝑏] точках 𝑥𝑖*, i = 0, 1, … , n.* При этом естественно стремиться к такому выбору узлов интерполяции, который

позволит сделать минимальной величину Δ(𝑃 ) = max| *f ( x )* − 𝑃 *( x )*| −

𝑛 𝑛

[𝑎,𝑏]

погрешность интерполяции на отрезке [𝑎, 𝑏].

Пусть о функции *f* известно лишь то, что она непрерывно дифференцируема *n* + 1 раз на отрезке [𝑎, 𝑏]. Тогда неравенство (6.28) дает верхнюю границу погрешности интерполяции:

𝛥̅(𝑃 ) = 𝑀𝑛+1

𝑛

(𝑛+1)!

max |𝜔 ( 𝑥)|. (6.33)

[𝑎,𝑏]

𝑛+1

Поставим теперь задачу определить набор узлов 𝑥0, 𝑥1, *…,* 𝑥𝑛*,* при котором величина 𝛥̅(𝑃𝑛) минимальна. Для решения этой задачи нам потребуются некоторые сведения о многочленах Чебышева.

1. **Многочлены Чебышева.** Введенные П.Л.Чебышевым многочлены 𝑇𝑛(𝑥) широко используются в вычислительной математике. При *n* = 0 и *n* = 1они определяются явными формулами

𝑇0(𝑥) = 1, 𝑇1(𝑥) = *x,* (6.34)

а при *n* ≥ 2 рекуррентной формулой

𝑇𝑛(𝑥) = 2𝑥𝑇𝑛−1(𝑥) – 𝑇𝑛−2(𝑥). (6.35) Запишем явные формулы для многочленов Чебышева 𝑇𝑛(𝑥)

7

для *n* = 2, 3, 4, 5:

𝑇2(𝑥) = 2𝑥𝑇1(𝑥) − 𝑇0(𝑥) = 2𝑥2 – 1,

𝑇3(𝑥) = 2𝑥𝑇2(𝑥) − 𝑇1(𝑥) = 4𝑥3 − 3𝑥,

𝑇4(𝑥) = 2𝑥𝑇3(𝑥) − 𝑇2(𝑥) = 8𝑥4 − 8𝑥2 + 1

𝑇5(𝑥) = 2𝑥𝑇4(𝑥) − 𝑇3(𝑥) = 16𝑥5 – 20 𝑥3 + 5𝑥.

Аналогично можно записать явные формулы и при *n ≥* 6. Приведем некоторые свойства многочленов Чебышева.

* 1. *При четном n многочлен* 𝑇𝑛(𝑥) *содержит только четные степени x и является четной функцией, а при нечетном n многочлен*

𝑇𝑛(𝑥) *содержит только нечетные степени x и является нечетной функцией*.

* 1. *При n ≥ 1 старший коэффициент многочлена* 𝑇𝑛(𝑥)

*равен* 2𝑛−1*, т.е.* 𝑇𝑛(𝑥) = 2𝑛−1𝑥𝑛 *+ … .*

Справедливость свойств 1) и 2) следует непосредственно из определения (6.34), (6.35).

* 1. *Для x ϵ* [–1*,* 1] *справедлива формула*

𝑇𝑛(𝑥) = cos (*n* · arcсos *x*). (6.36)

□ При *n =* 0 и *n* = 1 формула (6.36) верна, так как cos (0 · arcсos *x)=1,* cos (1 · arcсos *x)= x.* Для того, чтобы

доказать справедливость формулы для всех *n ≥* 0, достаточно показать, что функции С𝑛(𝑥) = cos (*n* · arcсos *x*) удовлетворяют такому же, как и многочлены Чебышева, рекуррентному соотношению

С𝑛(𝑥) = 2𝑥С𝑛−1(𝑥) – С𝑛−2(𝑥). (6.37)

Соотношение (6.37) получится, если в легко проверяемом тригонометрическом тождестве

cos [( *m* + 1)φ] + cos [( *m* – 1)φ] = 2cos φ cos *mφ*

положить *m = n – 1* и φ = arcсos *x.* ■

8

* 1. *При n ≥* 1 *многочлен* 𝑇𝑛(𝑥) *имеет ровно n действительных корней, расположенных на отрезке* [–1*,* 1] *и вычисляемых по формуле*

𝑥 = cos (2𝑘+1)𝜋 , *k* = 0, 1, … , *n* –1. (6.38)

𝑘

2𝑛

* 1. *При n ≥* 0 *справедливо равенство* max[−1,1] | 𝑇𝑛(𝑥)| = 1. *Если n ≥ 1, то этот максимум достигается ровно в n + 1 точках, которые находятся по формуле*

𝑥 = cos 𝜋𝑚 , *m* = 0, 1, … , *n*. (6.39)

𝑚

𝑛

*При этом* 𝑇𝑛 (𝑥𝑚)*=* (–*1*)𝑚*, т.е. максимумы и минимумы многочлена Чебышева чередуются.*

Доказательство свойств 4) и 5) основано на применении формулы (6.36). Например, в силу этой формулы корни многочлена 𝑇𝑛(𝑥), расположенные на отрезке [–1*,* 1], совпадают с корнями уравнения cos (*n* · arcсos *x*) = 0. Эквивалентное преобразование этого уравнения дает *n* · arcсos *x = π/2 + πk, k = 0,* ±1, ±2, … . Так как 0 ≤ arccos *x* ≤ π, то заключаем, что имеется ровно *n* корней 𝑥𝑘, отвечающих значениям

*k* = 0, 1, 2, … , *n* – 1 и удовлетворяющих равенствам arccos 𝑥 = (2𝑘+1)𝜋,

𝑘

2𝑛

эквивалентным формуле (6.38).

Назовем величину max[−1,1] | 𝑃𝑛(𝑥)| *уклонением многочлена* 𝑃𝑛(𝑥) *от нуля.* Эта величина характеризует максимальное отклонение (уклонение) графика многочлена 𝑃𝑛 от графика функции *y = 0* на отрезке [–1*,* 1]. Можно доказать следующее утверждение.

* 1. *Среди всех многочленов фиксированной степени n со старшим*

*коэффициентом* 𝑎𝑛*, равным 1, наименьшее уклонение от нуля (равное*

21−𝑛*) имеет многочлен* 𝑇̅̅𝑛̅(𝑥) = 21−𝑛 𝑇𝑛(𝑥)*.*

Благодаря этому свойству, имеющему особую ценность для приложений, многочлены Чебышева иногда называют *наименее уклоняющимися от нуля.* Свойство 6) можно сформулировать так: для любого многочлена вида 𝑃𝑛(𝑥) = 𝑥𝑛 + 𝑎𝑛−1𝑥𝑛−1 + … + 𝑎0, отличного

от ̅𝑇̅𝑛̅(𝑥), справедливо неравенство

21−𝑛 = max |̅𝑇̅̅𝑛̅(𝑥) | < max |𝑃𝑛(𝑥)|.

[−1,1] [−1,1]

9

З а м е ч а н и е. Из свойства 6) следует, что среди всех многочленов 𝑃𝑛(𝑥) фиксированной степени *n ≥ 1* со старшим коэффициентом 𝑎𝑛 ≠ 0 наименьшее уклонение от нуля (равное

|𝑎𝑛| 21−𝑛) имеет многочлен 𝑎𝑛̅𝑇̅̅𝑛̅(𝑥).

1. **Решение задачи минимизации оценки погрешности.** Найдем сначала решение задачи в предположении, что отрезок интерполяции [*a, b*] совпадает с отрезком [–1*,* 1]. В этом случае величина (6.33) будет минимальной при таком выборе узлов 𝑥0, 𝑥1, *…,* 𝑥𝑛*,* при котором

минимальна величина max

[−1,1]

|𝜔𝑛+1( 𝑥)|, т.е. минимально уклонение

многочлена 𝜔𝑛+1(𝑥) = (*x* – 𝑥0) (*x* – 𝑥1) … (*x* – 𝑥𝑛) от нуля. В силу свойств 4) и 6) многочленов Чебышева решение задачи дает набор узлов

𝑥 = cos (2𝑘+1 𝜋), *k* = 0, 1, … , *n,*

𝑘

2𝑛+2

являющихся нулями многочлена 𝑇𝑛+1, так как в этом случае 𝜔𝑛+1 =

̅𝑇̅̅𝑛̅̅+̅1̅ .

Заметим, что при таком выборе

𝛥̅(𝑃𝑛) = 𝑀𝑛+1 , (6.40)

(𝑛+1)!2𝑛

Причем в силу свойства 6) любой другой выбор узлов дает большее значение верхней границы погрешности. Для сравнения укажем, что при использовании для приближения функции *f* отрезка ряда Тейлора

𝑃𝑛(𝑥) = ∑𝑛 0

𝑘=

𝑓(𝑘)(0)

𝑘!

𝑥𝑘 верхняя граница оценки погрешности такова:

𝛥̅(𝑃 ) = 𝑀𝑛+1 .

𝑛

(𝑛+1)!

Следовательно, она в 2𝑛 раз хуже, чем при интерполяции с оптимальным выбором узлов.

Пусть теперь отрезок интерполяции [*a, b*] произволен. Приведем его к стандартному отрезку [–1*,* 1] заменой

*x* = a+b

2

10

+ b−a *t,* где *t ϵ* [–1*,* 1]. (6.41)

2

Как нетрудно видеть, в этом случае

𝜔 (𝑥) = ( 𝑏−𝑎

𝑛+1

𝜔̃

(𝑡), где 𝜔̃

(𝑡) =

𝑛+1

2 ) 𝑛+1

𝑛+1

*(t*−𝑡0)*(t*−𝑡1) … (𝑡 − 𝑡𝑛)*,* и 𝑥𝑘 = a+b + b−a 𝑡𝑘 *, k = 0, 1, … , n.*

2 2

Следовательно,

𝛥̅(𝑃 ) =  𝑀𝑛+1 ( 𝑏−𝑎

𝑛+1

max 𝜔̃

(𝑡)

𝑛 (𝑛+1)! 2 )

[−1,1]

𝑛+1

и минимум этой величины достигается при значениях 𝑡0, 𝑡1, … , 𝑡𝑛, совпадающих с нулями многочлена 𝑇𝑛+1. Значит, решение поставленной задачи дает выбор узлов

𝑥𝑘 = 𝑎+𝑏

+ 𝑏−𝑎 cos (2𝑘+1 𝜋), *k* = 0, 1, … , *n,* (6.42)

2 2 2𝑛+2

которому отвечает минимальное значение верхней границы погрешности интерполяции, равное

𝛥̅(𝑃 ) =  𝑀𝑛+1 ( 𝑏−𝑎

𝑛+1

.

𝑛 (𝑛+1)!2𝑛 2 )

# Лекция 12

**§ 6.6. Интерполяционная формула Ньютона для неравных промежутков**

Продолжим рассмотрение вопроса интерполирования при помощи алгебраических многочленов. В этом параграфе мы получим формулу Ньютона, являющуюся видоизменением формулы Лагранжа. Она интересна сама по себе и послужит нам источником получения ряда новых формул.

1. **Разделенные разности и их свойства.** Предварительно введем новое понятие – *разделенные разности*. Рассмотрим некоторую функцию *f* и систему узлов интерполирования 𝑥0, 𝑥1, … , 𝑥𝑛, 𝑥𝑖 ≠ 𝑥𝑗, при *i ≠ j,* 𝑥𝑖 𝜖 [*a, b*]. Для этой функции и узлов образуем всевозможные отношения

𝑓(𝑥1)−𝑓(𝑥0) = *f(*𝑥 ; 𝑥 )*;*

0 1

𝑥1− 𝑥0

𝑓(𝑥2)−𝑓(𝑥1) = *f(*𝑥1; 𝑥2)*, … ,* 𝑓(𝑥𝑛)−𝑓(𝑥𝑛−1) = *f(*𝑥𝑛−1; 𝑥𝑛). (6.42)

𝑥2− 𝑥1

𝑥𝑛− 𝑥𝑛−1

Такие отношения называют *разделенными разностями первого порядка.*

Получив разделенные разности первого порядка, мы можем образовать отношения

𝑓(𝑥1;𝑥2)−𝑓(𝑥0; 𝑥1) = *f(*𝑥 ; 𝑥 ; 𝑥 )*;*

0 1 2

𝑥2− 𝑥0

𝑓(𝑥2;𝑥3)−𝑓(𝑥1; 𝑥2) = *f(*𝑥1; 𝑥2; 𝑥3), *… ,* 𝑓(𝑥𝑛−1;𝑥𝑛)−𝑓(𝑥𝑛−2; 𝑥𝑛−1) =

𝑥3− 𝑥1

𝑥𝑛− 𝑥𝑛−2

*= f(*𝑥𝑛−2; 𝑥𝑛−1; 𝑥𝑛)*.* (6.43)

Эти отношения называют *разделенными разностями второго порядка.* Вообще, если мы уже определили разделенные разности *k*-го порядка *f(*𝑥𝑖; 𝑥𝑖+1; … ; 𝑥𝑖+𝑘)*,* то разделенные разности *(k+1)* -го порядка находятся при помощи формулы

𝑓(𝑥𝑖;𝑥𝑖+1;… ,𝑥𝑖+𝑘)−𝑓(𝑥𝑖−1;𝑥𝑖;… ,𝑥𝑖+𝑘−1) = *f(*𝑥

; 𝑥 ; … , 𝑥

). (6.44)

𝑥𝑖+𝑘− 𝑥𝑖−1

𝑖−1 𝑖

𝑖+𝑘

Условимся располагать таблицу разделенных разностей следующим образом:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***x*** | ***f(x)*** | ***f*(**𝒙𝒊; 𝒙𝒊+𝟏) | ***f(***𝒙𝒊; 𝒙𝒊+𝟏; 𝒙𝒊+𝟐) | ***f(***𝒙𝒊; 𝒙𝒊+𝟏; 𝒙𝒊+𝟐; 𝒙𝒊+𝟑) |
| 𝑥0 | *f(*𝑥0) |  |  |  |
|  |  | *f(*𝑥0; 𝑥1) |  |  |
| 𝑥1 | *f(*𝑥1) |  | *f(*𝑥0; 𝑥1; 𝑥2) |  |
|  |  | *f(*𝑥1; 𝑥2) |  | *f(*𝑥0; 𝑥1; 𝑥2; 𝑥3) |
| 𝑥2 | *f(*𝑥2) |  | *f(*𝑥1; 𝑥2; 𝑥3) |  |
|  |  | *f(*𝑥2; 𝑥3) |  | *f(*𝑥1; 𝑥2; 𝑥3; 𝑥4) |
| 𝑥3 | *f(*𝑥3) |  | *f(*𝑥2; 𝑥3; 𝑥4) |  |
|  |  | *f(*𝑥3; 𝑥4) |  |  |
| 𝑥4 | *f(*𝑥4) |  |  |  |

Для вывода формулы Ньютона нам потребуется использовать некоторые свойства разделенных разностей.

Прежде всего докажем, *что разделенная разность k-го порядка f(*𝑥𝑖; 𝑥𝑖+1; … ; 𝑥𝑖+𝑘) *равна*

*f(*𝑥 ; 𝑥 ; … ; 𝑥 ) *=*  𝑓(𝑥𝑖) *+*

𝑖 𝑖+1 𝑖+𝑘

(𝑥𝑖− 𝑥𝑖+1)(𝑥𝑖− 𝑥𝑖+2)…(𝑥𝑖− 𝑥𝑖+𝑘)

+ 𝑓(𝑥𝑖+1) *+ …*

(𝑥𝑖+1− 𝑥𝑖)(𝑥𝑖+1− 𝑥𝑖+2)…(𝑥𝑖+1− 𝑥𝑖+𝑘)

*… +*  𝑓(𝑥𝑖+𝑘) *=* ∑𝑖+𝑘  𝑓(𝑥𝑗) *,* (6.45)

(𝑥𝑖+𝑘− 𝑥𝑖)(𝑥𝑖+𝑘− 𝑥𝑖+1)…(𝑥𝑖+𝑘− 𝑥𝑖+𝑘−1)

𝑗=𝑖 𝜔ʹ(𝑥𝑗)

где ω(*x) = (x* − 𝑥𝑖) *(x* − 𝑥𝑖+1) *… (x* − 𝑥𝑖+𝑘)*.*

□ Доказательство будем вести по индукции. Для *k* = 1 это утверждение справедливо, так как

*f(*𝑥𝑖; 𝑥𝑖+1) = 𝑓(𝑥𝑖+1)−𝑓(𝑥𝑖) = 𝑓(𝑥𝑖) *+* 𝑓(𝑥𝑖+1) *.*

𝑥𝑖+1− 𝑥𝑖

(𝑥𝑖 − 𝑥𝑖+1)

(𝑥𝑖+1 − 𝑥𝑖)

Предположим, что оно справедливо для *k = l–1*, и докажем его справедливость для *k = l*. В самом деле,

*f(*𝑥 ; 𝑥

; … ; 𝑥

) *=* 𝑓(𝑥𝑖+1;… ,𝑥𝑖+𝑙)−𝑓(𝑥𝑖;𝑥𝑖+1;… ,𝑥𝑖+𝑙−1) *=*

𝑖 𝑖+1

𝑖+𝑙

𝑥𝑖+𝑙− 𝑥𝑖

*=* 1

(𝑥𝑖+𝑙 − 𝑥𝑖)

{ 𝑓(𝑥𝑖+1) *+*

(𝑥𝑖+1− 𝑥𝑖+2)(𝑥𝑖+1− 𝑥𝑖+3)…(𝑥𝑖+1− 𝑥𝑖+𝑙)

+ 𝑓(𝑥𝑖+2) *+ …*

(𝑥𝑖+2− 𝑥𝑖+1)(𝑥𝑖+2− 𝑥𝑖+3)…(𝑥𝑖+2− 𝑥𝑖+𝑙)

*… +* 𝑓(𝑥𝑖+𝑙)

(𝑥𝑖+𝑙− 𝑥𝑖+1)(𝑥𝑖+𝑙− 𝑥𝑖+2)…(𝑥𝑖+𝑙− 𝑥𝑖+𝑙−1)

*+*

*–* [ 𝑓(𝑥𝑖)

(𝑥𝑖− 𝑥𝑖+1)(𝑥𝑖− 𝑥𝑖+2)…(𝑥𝑖− 𝑥𝑖+𝑙−1)

+ 𝑓(𝑥𝑖+1) *+ …*

(𝑥𝑖+1− 𝑥𝑖)(𝑥𝑖+1− 𝑥𝑖+2)…(𝑥𝑖+2− 𝑥𝑖+𝑙−1)

*… +* 𝑓(𝑥𝑖+𝑙−1) ]}.

(𝑥𝑖+𝑙−1− 𝑥𝑖)(𝑥𝑖+𝑙−1− 𝑥𝑖+1)…(𝑥𝑖+𝑙−1− 𝑥𝑖+𝑙−2)

В полученном выражении 𝑓(𝑥𝑖) и 𝑓(𝑥𝑖+𝑙) встречаются по одному разу и

притом в виде (с учетом множителя 1

(𝑥𝑖+𝑙 − 𝑥𝑖)

, стоящего перед

фигурными скобками):

𝑓(𝑥𝑖) , 𝑓(𝑥𝑖+𝑙) ,

(𝑥𝑖− 𝑥𝑖+1)(𝑥𝑖− 𝑥𝑖+2)…(𝑥𝑖− 𝑥𝑖+𝑙)

(𝑥𝑖+𝑙− 𝑥𝑖)(𝑥𝑖+𝑙− 𝑥𝑖+1)(𝑥𝑖+𝑙− 𝑥𝑖+2)…(𝑥𝑖+𝑙− 𝑥𝑖+𝑙−1)

т.е. так, как они должны входить в доказываемое равенство (6.45). Все остальные 𝑓(𝑥𝑗) входят дважды. Объединяя эти члены попарно, получим:

1

(𝑥𝑖+𝑙 − 𝑥𝑖)

[ 𝑓(𝑥𝑗) *–*

(𝑥𝑗− 𝑥𝑖+1)… (𝑥𝑗− 𝑥𝑗−1)(𝑥𝑗− 𝑥𝑗+1)…(𝑥𝑗− 𝑥𝑖+𝑙)

*–* 𝑓(𝑥𝑗) ] =

(𝑥𝑗− 𝑥𝑖)… (𝑥𝑗− 𝑥𝑗−1)(𝑥𝑗− 𝑥𝑗+1)…(𝑥𝑗− 𝑥𝑖+𝑙−1)

*=* 𝑓(𝑥𝑗) *∙* 1

* [ 1

− 1 ]=

(𝑥𝑗− 𝑥𝑖+1)… (𝑥𝑗− 𝑥𝑗−1)(𝑥𝑗− 𝑥𝑗+1)…(𝑥𝑗−𝑥𝑖+𝑙−1) (𝑥𝑖+𝑙 − 𝑥𝑖)

(𝑥𝑗−𝑥𝑖+𝑙)

(𝑥𝑗− 𝑥𝑖)

= 𝑓(𝑥𝑗) ,

(𝑥𝑗− 𝑥𝑖)(𝑥𝑗− 𝑥𝑖+1)… (𝑥𝑗− 𝑥𝑗−1)(𝑥𝑗− 𝑥𝑗+1)…(𝑥𝑗−𝑥𝑖+𝑙)

что нам и требуется.

■

Из доказанного вытекает ряд следствий. Приведем некоторые из них.

С л е д с т в и е 1. *Разделенная разность суммы или разности функций равна сумме или разности разделенных разностей слагаемых, соответственно уменьшаемого и вычитаемого.*

С л е д с т в и е 2. *Постоянный множитель можно выносить за знак разделенной разности.*

С л е д с т в и е 3. *Разделенная разность есть симметричная функция своих аргументов, т.е.*

*f(*𝑥𝑖; 𝑥𝑖+1; … ; 𝑥𝑖+𝑘) *= f(*𝑥𝑖+1; 𝑥𝑖; … ; 𝑥𝑖+𝑘) *= f(*𝑥𝑖+2, 𝑥𝑖+1; 𝑥𝑖; 𝑥𝑖+3; … ; 𝑥𝑖+𝑘) *=*

*…*

# Вывод формулы Ньютона для неравных промежутков.

Перейдем к выводу формулы Ньютона. Пусть *f (x)* - заданная функция,

𝑥0, 𝑥1, … , 𝑥𝑛 - узлы интерполирования, 𝑥𝑖 ≠ 𝑥𝑗 при *i ≠ j,* 𝑥𝑖 𝜖 [*a, b*] и

𝐿𝑘(*x*) - интерполяционный многочлен Лагранжа, построенный для этой функции по узлам 𝑥0, 𝑥1, … , 𝑥𝑛. Тогда

𝐿𝑛(*x*) = 𝐿0(*x*) + [𝐿1(*x*) −𝐿0(*x*)] + [𝐿2(*x*) − 𝐿1(*x*)] + …

… + [𝐿𝑛(*x*) − 𝐿𝑛−1(*x*)]. (6.46)

Рассмотрим отдельную разность, стоящую в правой части,

𝐿𝑘(*x*) − 𝐿𝑘−1(*x*). Это многочлен степени *k.* Он обращается в нуль в точках 𝑥0, 𝑥1, … , 𝑥𝑘−1, т.к. в этих точках 𝐿𝑘(𝑥𝑖) = 𝑓(𝑥𝑖) и 𝐿𝑘−1(𝑥𝑖) =

𝑓(𝑥𝑖). Поэтому 𝐿𝑘(*x*) − 𝐿𝑘−1(*x*) = 𝐴𝑘 (*x* − 𝑥0) (*x* − 𝑥1) … (𝑥 − 𝑥𝑘−1) (𝐴𝑘 − постоянная). Для определения величины 𝐴𝑘 положим 𝑥 = 𝑥𝑘. При этом получим:

𝑓(𝑥𝑘) − 𝐿𝑘−1(𝑥𝑘) = 𝐴𝑘 (𝑥𝑘 − 𝑥0) (𝑥𝑘 − 𝑥1) … (𝑥𝑘 − 𝑥𝑘−1).

Откуда следует:

𝐴 = 𝑓(𝑥𝑘) –

𝑘

(𝑥𝑘 − 𝑥0) (𝑥𝑘 − 𝑥1)… (𝑥𝑘− 𝑥𝑘−1)

∑𝑘−1 𝑓(𝑥 ) (𝑥𝑘 − 𝑥0)… (𝑥𝑘 − 𝑥𝑗−1)(𝑥𝑘 − 𝑥𝑗+1)… (𝑥𝑘− 𝑥𝑘−1)

𝑗=0

−

𝑗

(𝑥𝑗 − 𝑥0)… (𝑥𝑗 − 𝑥𝑗−1)(𝑥𝑗 − 𝑥𝑗+1)… (𝑥𝑗− 𝑥𝑘−1) =

(𝑥𝑘 − 𝑥0) (𝑥𝑘 − 𝑥1)… (𝑥𝑘− 𝑥𝑘−1)

= ∑𝑘

𝑓(𝑥𝑗) = *f* (𝑥

; 𝑥

;…; 𝑥 ).

𝑗=0 (𝑥𝑗 − 𝑥0)… (𝑥𝑗 − 𝑥𝑗−1)(𝑥𝑗 − 𝑥𝑗+1)… (𝑥𝑗 − 𝑥𝑘)

0 1 𝑘

Отсюда

𝐿𝑛(*x*) = 𝑓(𝑥0) + (*x*−𝑥0) *f* (𝑥0; 𝑥1) + (𝑥 – 𝑥0)(𝑥 – 𝑥1)𝑓(𝑥0; 𝑥1; 𝑥2) +

+ ⋯ + (*x* − 𝑥0) (*x* − 𝑥1) … (𝑥 − 𝑥𝑛−1) *f* (𝑥0; 𝑥1;…; 𝑥𝑛). (6.47)

Эта форма записи интерполяционного многочлена Лагранжа и носит название *интерполяционного многочлена Ньютона для неравных промежутков.* Она более удобна для вычислений, чем формула Лагранжа. Добавление одного или нескольких узлов не приводит к повторению всей проделанной работы заново, как это было при вычислениях по формуле Лагранжа.

1. **Остаточный член формулы Ньютона.** Остаточный член формулы Ньютона точно такой же, как и у формулы Лагранжа. Но его можно записать и в другой форме. Для этого рассмотрим

*f* (*x*; 𝑥 ; 𝑥 ; … ; 𝑥 ) = 𝑓(𝑥) +

0 1 𝑛

(𝑥 − 𝑥0) (𝑥 − 𝑥1)… (𝑥 − 𝑥𝑛)

+ 𝑓(𝑥0) (𝑥0 − 𝑥) (𝑥0 − 𝑥1)… (𝑥0− 𝑥𝑛)

+ … + 𝑓(𝑥𝑛) .

(𝑥𝑛 − 𝑥0) (𝑥𝑛 − 𝑥1)… (𝑥𝑛− 𝑥𝑛−1)

(6.48)

Отсюда

*f* (*x*) = 𝑓(𝑥 ) (𝑥 − 𝑥1) (𝑥 − 𝑥2)… (𝑥 − 𝑥𝑛) + …

0

(𝑥0− 𝑥1) (𝑥0 − 𝑥2)… (𝑥0 − 𝑥𝑛)

… + 𝑓(𝑥 ) (𝑥 − 𝑥1) (𝑥 − 𝑥2)… (𝑥 − 𝑥𝑛−1) +

𝑛

(𝑥𝑛− 𝑥0) (𝑥𝑛 − 𝑥1)… (𝑥𝑛 − 𝑥𝑛−1)

+(𝑥 − 𝑥0)(𝑥 − 𝑥1) … (𝑥 − 𝑥𝑛) 𝑓 (𝑥; 𝑥0; 𝑥1; … ; 𝑥𝑛). (6.49)

Итак,

*f* (*x*) = 𝐿𝑛(*x*) + (𝑥 − 𝑥0)(𝑥 − 𝑥1) … (𝑥 − 𝑥𝑛) 𝑓 (𝑥; 𝑥0; 𝑥1; … ; 𝑥𝑛). (6.50) Таким образом,

𝑅𝑛(*x*) = *f* (*x*) − 𝐿𝑛(*x*) =

= (𝑥 − 𝑥0)(𝑥 − 𝑥1) … (𝑥 − 𝑥𝑛) 𝑓 (𝑥; 𝑥0; 𝑥1; … ; 𝑥𝑛). (6.51) В частности, если *f* (*x*) имеет производную порядка *n+1*, то получим:

𝑓 (𝑥; 𝑥0

; 𝑥1

; … ; 𝑥𝑛

) = 𝑓(𝑛+1)(𝜉) .

(𝑛+1)!

Заметим, что так как величина *f* (*x*) нам неизвестна, формула (6.51) имеет только теоретическое значение, а именно, она может быть использована для оценки погрешности, даваемой интерполяционной формулой Ньютона для заданной функции *f* (*x*).

# § 6.7. Интерполяционные формулы Ньютона для равных промежутков

Естественно ожидать, что если промежутки между последовательными узлами интерполирования равны, т.е. 𝑥𝑖 − 𝑥𝑖−1 – постоянная величина, то предыдущая *формула* упростится. Так оно и есть на самом деле. Прежде чем приступить к выводу формул для этого случая введем понятие о конечных разностях.

* + 1. **Конечные разности и их свойства.** Пусть для значений *x:*

𝑥0, 𝑥0 + ℎ, 𝑥0 + 2ℎ, … , 𝑥0 + 𝑛ℎ (*h –* шаг таблицы)*,* нам известны значения функции *f (x):* 𝑦0, 𝑦1, *… ,* 𝑦𝑛.

Назовем разности

𝑦1 − 𝑦0, 𝑦2 − 𝑦1, … , 𝑦𝑛 − 𝑦𝑛−1

*конечными разностями первого порядка.*

Будем обозначать их следующим образом

𝑦𝑖+1 − 𝑦𝑖 = 𝑓1 1

. (6.52)

𝑖+ ⁄2

Из разностей первого порядка можно образовать конечные разности второго порядка:

𝑓1 − 𝑓1 = 𝑓2; 𝑓1 − 𝑓1 = 𝑓2; … ; 𝑓1 − 𝑓1 = 𝑓2; … (6.53)

3⁄2

1⁄2

1 5⁄2

3⁄2 2

2𝑖+1)/2

(2𝑖−1)/2 𝑖

Аналогично можно образовать разности третьего порядка, четвертого и т.д. Таблицу разностей обычно располагают следующим образом:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***x*** | ***f*** | 𝒇𝟏 | 𝒇𝟐 | 𝒇𝟑 |
| 𝑥0 | 𝑓0 |  |  |  |
|  |  | 𝑓1  1⁄2 |  |  |
| 𝑥1 | 𝑓1 |  | 𝑓2  1 |  |
|  |  | 𝑓1  3⁄2 |  | 𝑓3  3⁄2 |
| 𝑥2 | 𝑓2 |  | 𝑓2  2 |  |
|  |  | 𝑓1  5⁄2 |  | 𝑓3  5⁄2 |
| 𝑥3 | 𝑓3 |  | 𝑓2  3 |  |
|  |  | 𝑓1  7⁄2 |  |  |
| 𝑥4 | 𝑓4 |  |  |  |

Установим связь между конечными и разделенными разностями для случая, когда 𝑥𝑖 − 𝑥𝑖−1 постоянны. Будем иметь:

*f* (𝑥𝑖; 𝑥𝑖+1) = 𝑓𝑖+1− 𝑓𝑖 =

𝑥 − 𝑥

𝑖+1 𝑖

𝑓 (𝑥

𝑓1 1

𝑖+ ⁄2,

ℎ

; 𝑥

)− 𝑓 (𝑥 ;𝑥

) 𝑓1 3

− 𝑓1 1

𝑓2

*f* (𝑥 ; 𝑥

; 𝑥

) = 𝑖+1 𝑖+2 𝑖 𝑖+1 =

𝑖+ ⁄2

𝑖+ ⁄2 = 𝑖+1 .

𝑖 𝑖+1

𝑖+2

𝑥𝑖+2− 𝑥𝑖

2ℎ∙ ℎ

2ℎ2

Вообще,

𝑓𝑘 𝑘

𝑓 (𝑥 ; 𝑥

; … ; 𝑥

) = 𝑖+

⁄2 . (6.54)

𝑖 𝑖+1

𝑖+𝑘

𝑘! ℎ𝑘

Доказательство будем вести методом индукции. Предполагая формулу справедливой для *k ≤ l,* докажем ее справедливость для *k = l+1.*

Действительно,

𝑓 (𝑥𝑖; 𝑥𝑖+1; … ; 𝑥𝑖+𝑙+1) = 𝑓 (𝑥𝑖+1;…;𝑥𝑖+𝑙+1) − 𝑓 (𝑥𝑖; 𝑥𝑖+1;…;𝑥𝑖+𝑙) =

𝑥 − 𝑥

𝑖+𝑙+1 𝑖

𝑓𝑙

𝑙 − 𝑓𝑙 𝑙

𝑓𝑙 (𝑙+1)

= 𝑖+1+ ⁄2 𝑖+ ⁄2 = 𝑖+ 2 .

⁄

(𝑙+1)ℎ∙ 𝑙! ℎ𝑙 (𝑙+1)! ℎ𝑙+1

* + 1. **Вывод интерполяционных формул Ньютона.** Перейдем теперь к выводу интерполяционных формул Ньютона. Для этого рассмотрим интерполяционную формулу для неравных промежутков, взяв в ней в качестве узлов интерполирования 𝑥0, 𝑥1, … , 𝑥𝑛 точки

𝑥0, 𝑥0 + ℎ, 𝑥0 + 2ℎ, … , 𝑥0 + 𝑛ℎ. При этом, заменяя разделенные разности их выражениями через конечные разности, получим:

𝐿𝑛(*x*) = 𝑓0+ (𝑥−𝑥0) 𝑓1 + (𝑥 – 𝑥0)(𝑥 – 𝑥1) 𝑓2 +

1

ℎ 2 2ℎ2 1

+ ⋯ + (𝑥 − 𝑥0) (𝑥 − 𝑥1)… (𝑥 − 𝑥𝑛−1) 𝑓𝑛 . (6.55)

𝑛! ℎ𝑛

𝑛/2

Обозначим (𝑥−𝑥0) = *t,* тогда наша формула (6.55) примет вид

ℎ

𝐿𝑛(𝑥0 + ℎ𝑡) = 𝑓0+ *t*𝑓1 + 𝑡(𝑡−1) 𝑓2 +

2

1 2! 1

+ ⋯ + 𝑡(𝑡−1)…(𝑡−(𝑛−1)) 𝑓𝑛 . (6.56)

𝑛!

𝑛/2

Полученную формулу называют *интерполяционной формулой Ньютона для интерполирования вперед*.

Выведем еще одну интерполяционную формулу Ньютона. Опять будем использовать интерполяционную формулу Ньютона для неравных промежутков, взяв в ней в качестве узлов интерполирования

𝑥0, 𝑥1, … , 𝑥𝑛 точки 𝑥0, 𝑥0 − ℎ, 𝑥0 − 2ℎ, … , 𝑥0 − 𝑛ℎ. При этом получим:

𝐿𝑛(*x*) = 𝑓(𝑥0) + (*x*−𝑥0) *f* (𝑥0; 𝑥0 − ℎ) +

+ (𝑥 – 𝑥0)(𝑥 – 𝑥0 + ℎ)𝑓(𝑥0; 𝑥0 − ℎ; 𝑥0 − 2ℎ) + … +

+ (*x* − 𝑥0) (*x* − 𝑥0 + ℎ) … (𝑥 − 𝑥0 + (𝑛 − 1)ℎ) *f* (𝑥0; 𝑥0 − ℎ;…;𝑥0 − 𝑛ℎ).

Но в силу симметрии разделенных разностей относительно своих аргументов будем иметь:

*f* (𝑥0; 𝑥0 − ℎ;…; 𝑥0 − 𝑖ℎ) = *f* (𝑥0 − 𝑖ℎ; 𝑥0 − 𝑖ℎ + ℎ;…; 𝑥0 − ℎ; 𝑥0).

Снова заменим разделенные разности конечными

𝑖

𝑓

*f* (𝑥0 − 𝑖ℎ; 𝑥0 − 𝑖ℎ + ℎ;…; 𝑥0 − ℎ; 𝑥0) = −𝑖/2.

𝑖!ℎ𝑖

Отсюда

𝐿𝑛(*x*) = 𝑓0+ (𝑥−𝑥0) 𝑓1 + (𝑥 – 𝑥0)(𝑥 – 𝑥0+ℎ) 𝑓2 +

ℎ −1/2

2!ℎ2 −1

+ ⋯ + (𝑥 − 𝑥0) (𝑥 − 𝑥0+ℎ)… (𝑥 − 𝑥0+(𝑛−1)ℎ) 𝑓𝑛 . (6.57)

𝑛! ℎ𝑛

Заменяя, как и прежде, (𝑥−𝑥0)

ℎ

−𝑛/2

на *t,* получим:

𝐿𝑛(𝑥0 + ℎ𝑡) = 𝑓0+ *t*𝑓1 + 𝑡(𝑡+1) 𝑓2 +

−1/2

2! −1

+ ⋯ + 𝑡(𝑡+1)…(𝑡+(𝑛−1)) 𝑓𝑛 . (6.58)

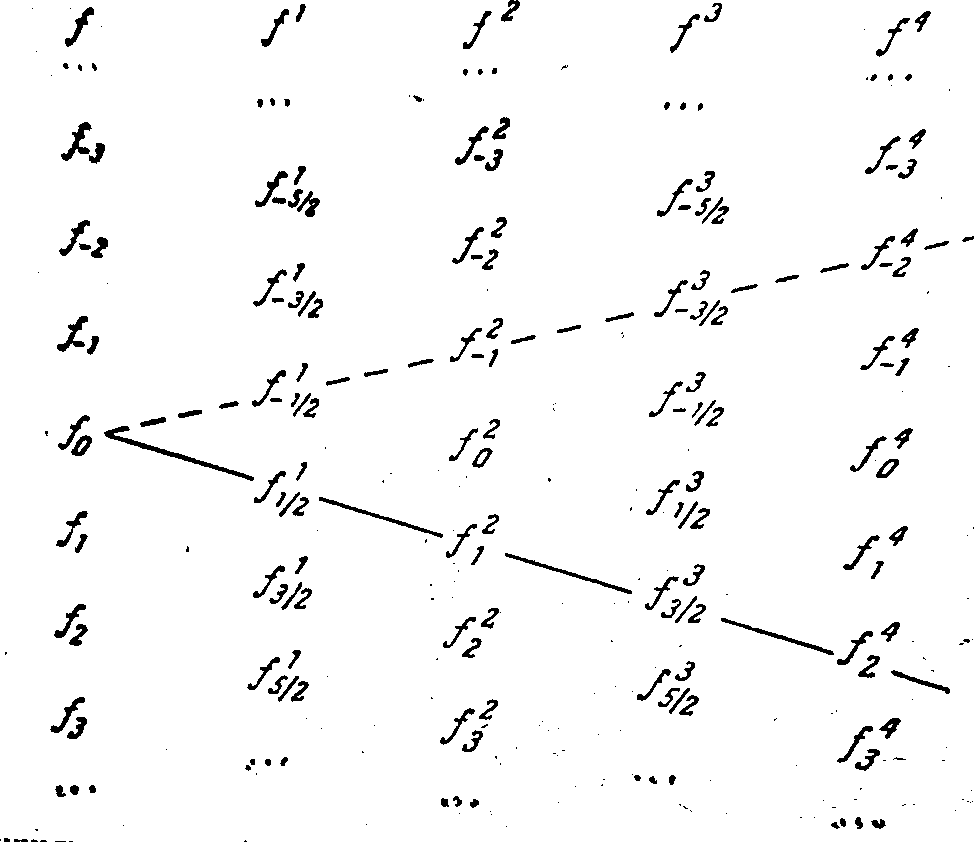
𝑛!

−𝑛/2

Это есть *интерполяционная формула Ньютона для интерполирования назад*.

Как видно из приведенной ниже таблицы разностей, в формулу Ньютона для интерполирования вперед входят разности, расположенные на диагонали, начинающейся в 𝑓0 и идущей вниз, а в

формуле Ньютона для интерполирования назад используются разности, расположенные на диагонали, начинающейся тоже в 𝑓0, но идущей вверх:



* + 1. **Остаточные члены интерполяционных формул Ньютона.** Сейчас мы приведем выражения для остаточных членов интерполяционных формул Ньютона для интерполирования вперед и назад, следующие из формулы (6.51) для остаточного члена интерполяционной формулы для неравноотстоящих узлов.

Для первой формулы получим:

𝑅𝑛(*x*) = *f* (*x*) − 𝐿𝑛(*x*) =

= (𝑥 − 𝑥0

)(𝑥 − 𝑥0

− ℎ) … (𝑥 − 𝑥0

− 𝑛ℎ) 𝑓(𝑛+1)(𝜉) =

(𝑛+1)!

= ℎ𝑛+1𝑓(𝑛+1)(𝜉) *t* (*t*−1) … (*t*−*n*). (6.59)

(𝑛+1)!

Для второй

𝑅𝑛(*x*) = *f* (*x*) − 𝐿𝑛(*x*) =

= (𝑥 − 𝑥0

)(𝑥 − 𝑥0

+ ℎ) … (𝑥 − 𝑥0

+ 𝑛ℎ) 𝑓(𝑛+1)(𝜉) =

(𝑛+1)!

= ℎ𝑛+1𝑓(𝑛+1)(𝜉) *t* (*t*+1) … (*t*+*n*). (6.60)

(𝑛+1)!

# Лекция 13

**§ 6.8. Обсуждение глобальной полиномиальной интерполяции. Понятие о кусочно-полиномиальной интерполяции**

Пусть функция интерполируется на отрезке [*a, b*]. Метод решения этой задачи единым для всего отрезка многочленом 𝑃𝑛(𝑥) называют *глобальной полиномиальной интерполяцией*.

Cуществуют весьма веские причины, по которым глобальная интерполяция многочленами высокой степени в вычислительной практике, как правило, не используется. Обсудим некоторые из этих причин.

1. **Сходимость при увеличении числа узлов.** Всегда ли можно добиться повышения точности интерполяции благодаря увеличению числа узлов (и соответственно степени интерполяционного многочлена)? Хотя положительный ответ на этот вопрос напрашивается сам собой, не будем торопиться с выводами.

Уточним постановку задачи. Для того чтобы реализовать процесс интерполяции многочленами возрастающей степени *n*, необходимо

указать стратегию выбора узлов интерполяции 𝑥(𝑛), 𝑥(𝑛), … , 𝑥(𝑛). Такая

0 1 𝑛

стратегия задается указанием *интерполяционного массива* – треугольной таблицы

𝑥(0)

0

𝑥(1), 𝑥(1)

0 1

…………

𝑥(𝑛), 𝑥(𝑛), … , 𝑥(𝑛),

0 1 𝑛

в каждой строке которой все 𝑥(𝑛) различны и 𝑥(𝑛)𝜖 [*a, b*]. Будем

𝑖 𝑖

говорить, что при заданной стратегии выбора узлов *метод интерполяции сходится*, если max[𝑎,𝑏] |𝑓(𝑥) − 𝑃𝑛(𝑥)| → 0 при *n* → ∞. Рассмотрим сначала простейшую стратегию, состоящую в равномерном распределении на отрезке [*a, b*] узлов интерполяции, т.е.

в выборе 𝑥(𝑛) = *a + ih* (*i = 0, 1,* … , *n*), где *h = (b – a) / n.*

𝑖

Можно привести пример, когда такая стратегия не может обеспечить сходимость даже для очень гладких функций:

**П р и м е р Р у н г е**. Используем глобальную полиномиальную интерполяцию с равномерным распределением узлов для приближения

на отрезке [-1, 1] следующей функции: *f(x)* = 1

1+25𝑥

. Вычисления

показывают, что при больших *n* интерполяция дает превосходные результаты в центральной части отрезка. В то же время вопреки ожиданиям последовательность 𝑃𝑛(𝑥) расходится для 0.73 < | *x* | ≤ 1. Соответствующая иллюстрация приведена на рис.1.

2

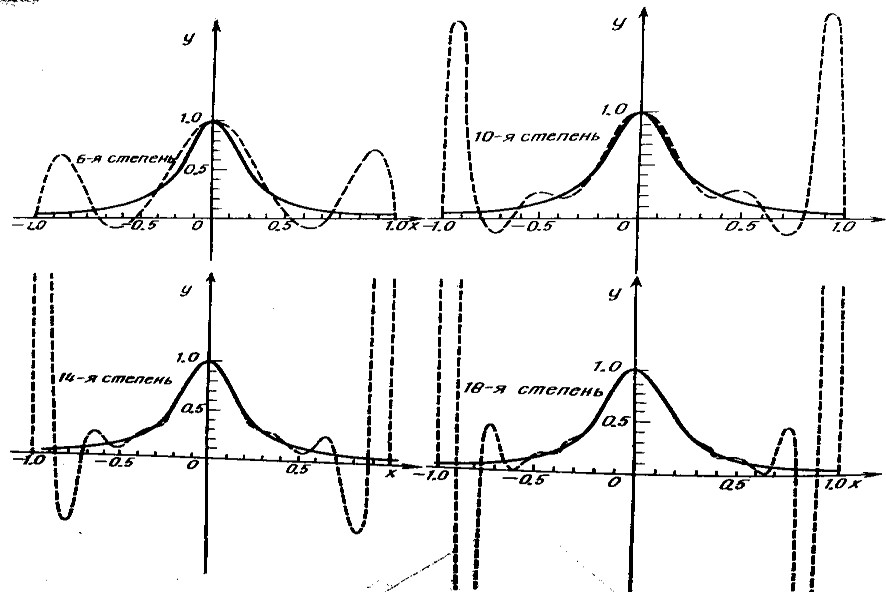


Рис.1

Таким образом, равномерное распределение узлов интерполяции для функции Рунге оказалось неудачным.

Однако проблема сходимости для этой функции исчезает, если в качестве узлов интерполяции брать корни многочлена Чебышева

𝑇𝑛+1(𝑥).

Существует ли единая для всех непрерывных на [*a, b*] функций *f* стратегия выбора узлов интерполяции, гарантирующая ее сходимость? Отрицательный ответ на этот вопрос дает следующая теорема.

**Т е о р е м а 6.6** (теорема Фабера). *Какова бы ни была стратегия выбора узлов интерполяции, найдется непрерывная на* [*a, b*] *функция f*

*, для которой* max[𝑎,𝑏] |𝑓(𝑥) − 𝑃𝑛(𝑥)| → ∞ при *n* → ∞.

Теорема Фабера отрицает существование единой для всех непрерывных функций стратегии выбора узлов интерполяции.

Однако для гладких функций (а именно такие функции чаще всего и интерполируются) такая стратегия существует, о чем говорит следующая теорема.

**Т е о р е м а 6.7.** *Пусть в качестве узлов интерполяции на отрезке* [*a, b*] *выбираются чебышевские узлы*

𝑥𝑘 = 𝑎+𝑏

+ 𝑏−𝑎 cos (2𝑘+1 𝜋), *k* = 0, 1, … , *n.*

2 2 2𝑛+2

*Тогда для любой непрерывно дифференцируемой на отрезке* [*a, b*] *функции f метод интерполяции сходится.*

З а м е ч а н и е. Практическая реализация выбора чебышевских узлов интерполяции возможна и оправданна в довольно редких случаях и просто невозможна тогда, когда приходится иметь дело с заданной таблицей значений функции.

1. **Чувствительность интерполяционного многочлена к погрешностям входных данных.** Помимо погрешности, которая возникает от приближенной замены функции *f* интерполяционным многочленом, возникает еще дополнительная погрешность, связанная с тем, что значения интерполируемой функции также задаются с погрешностью.

Пусть заданные в узлах 𝑥𝑖 значения 𝑦∗ содержат погрешности 𝜀𝑖. Тогда вычисляемый по этим значениям многочлен 𝑃∗(𝑥) =

𝑖

𝑛

𝑗

𝑛

∑

𝑗=0

𝑦∗ 𝑙𝑛𝑗(*x*) содержит погрешность

𝑃𝑛(𝑥) − 𝑃∗(𝑥) = ∑𝑛 𝜀 𝑙

(*x*). (6.61)

𝑛 𝑗=0 𝑗 𝑛𝑗

Пусть известно, что верхняя граница погрешности значений 𝑦∗ равна 𝛥̅(𝑦∗), т.е. |𝜀𝑖| ≤ 𝛥̅(𝑦∗) для всех *i* = 0, 1, …, *n*. Тогда для верхней границы соответствующей погрешности многочлена 𝛥̅(𝑃∗) = max[𝑎,𝑏] |𝑃𝑛(𝑥) − 𝑃∗(𝑥)| в силу равенства (6.61) справедлива оценка

𝑖

𝑛

𝑛

𝛥̅(𝑃∗) ≤ 𝛬𝑛

𝑛

𝛥̅(𝑦∗), (6.62)

здесь 𝛬𝑛 = max[𝑎,𝑏] ∑𝑛

𝑗=0

|𝑙𝑛𝑗(𝑥)|

− величина, которую называют

константой Лебега. В задаче интерполирования константа Лебега играет роль абсолютного числа обусловленности.

Величина 𝛬𝑛 не зависит от длины отрезка [*a, b*], а определяется лишь относительным расположением узлов на нем. В общем случае оптимальный выбор узлов неизвестен. Установлено, однако, что почти оптимальным является выбор в качестве узлов интерполяции корней

многочлена Чебышева. При таком выборе 𝛬 ≈ 2 ln (*n* + 1) + 1.

𝑛

𝜋

Известно, что крайне неудачным при больших *n* является выбор

равноотстоящих узлов интерполяции. При таком выборе 𝛬𝑛

> 2𝑛−1 (2𝑛−1)√𝑛

для *n ≥ 4* и обусловленность задачи резко ухудшается с ростом *n.*

Сказанное позволяет сделать важный вывод: в вычислениях не следует использовать интерполяционные многочлены высокой степени с равноотстоящими узлами.

Альтернативный подход состоит в *локальной интерполяции*, когда функция *f* аппроксимируется многочленом 𝑃𝑚(𝑥) невысокой степени *m* на содержащемся в [*a, b*] отрезке [α*, β*] малой длины. Естественно, что при этом используется лишь часть табличных значений. Рассмотрим два подхода к приближению функции, основанные на локальной интерполяции.

1. **Интерполирование с помощью «движущегося» полинома.** Строят набор полиномов 𝑃(0,1,…,𝑚), 𝑃(1,2,…,𝑚+1), …, 𝑃(𝑛−𝑚,𝑛−𝑚+1,…,𝑛) фиксированной степени *m*, каждый из которых совпадает с табличными значениями в *m* + 1 последовательных точках. Каждый такой полином используют для приближения функции в тех точках *x* из отрезка [*a, b*], для которых выбранные узлы таблицы являются ближайшими.

**Пример 1**. Пусть функция задана следующей таблицей:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *i* | 0 | 1 | 2 | 3 | 4 |
| 𝑥𝑖 | 0 | 1 | 2 | 3 | 4 |
| 𝑦𝑖 | 1.0 | 1.8 | 2.2 | 1.4 | 1.0 |

Для интерполяции этой функции воспользуемся «движущимся» полиномом второй степени. Заметим, что для *x ϵ* [0.0, 1.5] для приближения используется многочлен 𝑃(0,1,2)(𝑥), при *x ϵ* [1.5, 2.5]

− многочлен 𝑃(1,2,3)(𝑥), при *x ϵ* [2.5, 4.0] − многочлен 𝑃(2,3,4)(𝑥). Соответствующая геометрическая иллюстрация приведена на рис. 2.

Заметим, что полученная таким образом аппроксимирующая функция имеет разрывы в точках *x* = 1.5 и *x* = 2.5.

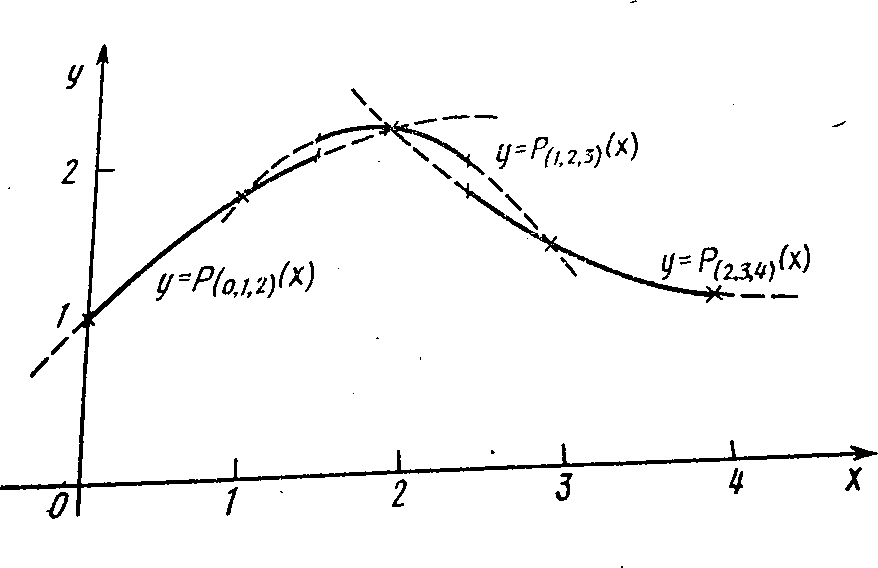


Рис. 2

**Пример 2**. Для интерполяции функции из примера 1 используем кусочно-полиномиальную интерполяцию. На отрезке [0, 2] аппроксимируем функцию многочленом 𝑃(0,1,2)(𝑥), а на отрезке [2, 4] – многочленом 𝑃(2,3,4)(𝑥). Соответствующая геометрическая иллюстрация приведена на рис. 3. Заметим, что результирующая аппроксимирующая функция непрерывна, но в точке 𝑥 = 2 график ее имеет излом, соответствующий разрыву первой производной.

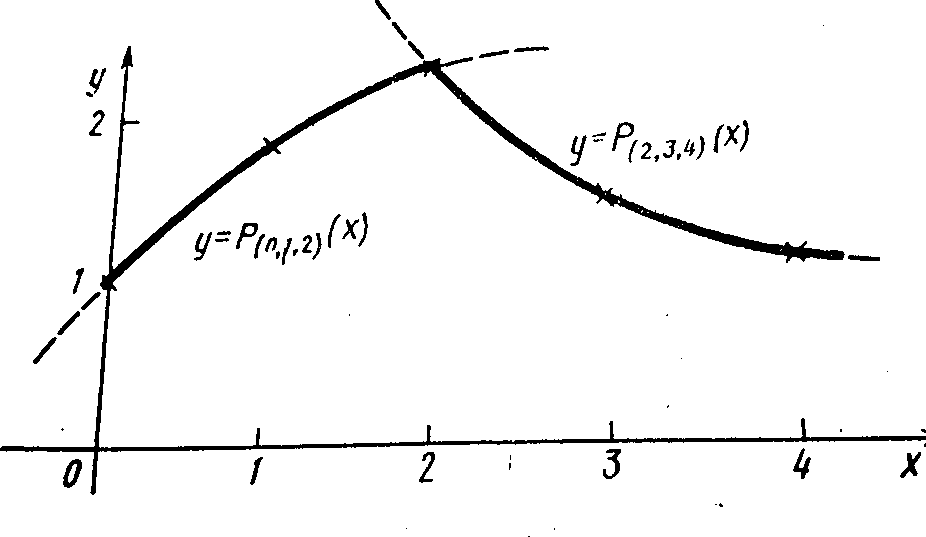


Рис. 3

Метод кусочно-полиномиальной интерполяции при использовании многочленов фиксированной степени *m* имеет (*m* + 1)- й порядок точности. Заметим, что интерполяцию «движущимся» полиномом можно рассматривать как частный случай кусочно- полиномиальной интерполяции, при этом интерполяционный многочлен является непрерывным, но в точках стыка не обеспечивается его гладкость.

# § 6.9. Интерполяция сплайнами

1. **Определение сплайна.** Итак, мы установили, что использование многочленов высокой степени может привести к неудаче или оказаться неэффективным. Альтернативный подход состоит в кусочно-полиномиальной интерполяции с использованием многочленов невысокой степени. Однако недостатком этого метода является разрыв производной в точках стыка двух соседних многочленов. Вместе с тем нередко требуется, чтобы аппроксимирующая функция была гладкой.

Потребность в наличии аппроксимирующих функций, которые бы сочетали в себе локальную простоту многочленов невысокой степени и глобальную на всем промежутке [*a, b*] гладкость, привела к появлению в 1946 году так называемых *сплайн-функций* или *сплайнов –* специальным образом построенных гладких кусочно-многочленных функций.

К настоящему времени сплайны стали составной частью самых различных вычислительных методов и нашли широкое применение в решении разнообразных научно-технических и инженерных задач.

Дадим строгое определение сплайна. Пусть отрезок [*a, b*] разбит точками *a =* 𝑥0 *<* 𝑥1 < *… <* 𝑥𝑛*= b* на *n* частичных отрезков [𝑥𝑖−1*,* 𝑥𝑖]. *Сплайном степени m* называется функция 𝑆𝑚(𝑥), обладающая следующими свойствами:

1. функция 𝑆𝑚(𝑥) непрерывна на отрезке [*a, b*] вместе со

всеми своими производными 𝑆(1)(𝑥), 𝑆(2)(𝑥), … , 𝑆(𝑝)(𝑥), до некоторого

порядка *p*;

𝑚 𝑚 𝑚

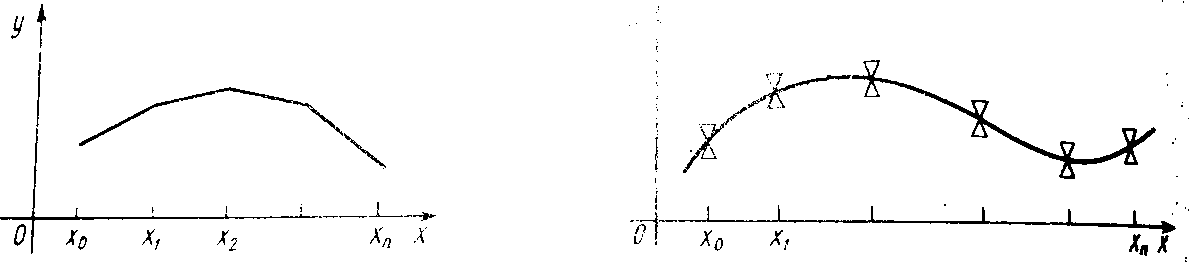
1. на каждом частичном отрезке [𝑥𝑖−1*,*𝑥𝑖] функция 𝑆𝑚(𝑥) совпадает с некоторым алгебраическим многочленом 𝑃𝑚,𝑖(𝑥) степени *m*.

Разность *m – p* между степенью сплайна и наивысшим порядком непрерывной на отрезке [*a, b*] производной называется *дефектом сплайна.*

Простейшим примером сплайна дает непрерывная кусочно- линейная функция (рис. 4), являющаяся сплайном первой степени (*линейным сплайном*) с дефектом, равным единице. Действительно, на отрезке [*a, b*] сама функция 𝑆1(𝑥) (нулевая производная) непрерывна. В то же время на каждом частичном отрезке 𝑆1(𝑥) совпадает с некоторым многочленом первой степени.

Наиболее широкое распространение на практике получили сплайны 𝑆3(𝑥) третьей степени (кубические сплайны) с дефектом, равным 1 или 2. Такие сплайны на каждом из частичных отрезков [𝑥𝑖−1*,* 𝑥𝑖] совпадают с кубическим многочленом:

𝑆3(𝑥) = 𝑃3,𝑖(𝑥) = 𝑎𝑖 + 𝑏𝑖(𝑥 − 𝑥𝑖) + 𝑐𝑖(𝑥 − 𝑥𝑖)2 + + 𝑑𝑖(𝑥 − 𝑥𝑖)3 (6.63)

и имеют на отрезке [*a, b*] по крайней мере, одну непрерывную производную 𝑆ʹ (𝑥).

3

Рис. 4 Рис. 5

Термин «сплайн» происходит от английского слова «spline» (гибкая линейка, стержень) – названия приспособления, использовавшегося чертежниками для проведения гладких кривых через заданные точки. Если гибкую стальную линейку поставить на ребро и, изогнув, зафиксировать ее положение в узловых точках (рис.5), то получится механический аналог кубического сплайна. В самом деле, из курса сопротивления материалов известно, что уравнение свободного равновесия профиля S(𝑥) линейки таково: 𝑆(4)(𝑥) = 0. Следовательно, в промежутке между двумя соседними узлами S(𝑥) представляет

собой многочлен третьей степени. В то же время отсутствие у линейки изломов свидетельствует о непрерывности касательной к графику функции S(𝑥) и кривизны, т.е. производных 𝑆ʹ(𝑥) и 𝑆ʹʹ(𝑥).

1. **Интерполяционный сплайн.** Пусть функция *y* = *f (x)* задана таблицей своих значений 𝑦𝑖 = *f* (𝑥𝑖), *i = 0, 1. …, n*. Сплайн 𝑆𝑚(𝑥) называется интерполяционным, если 𝑆𝑚(𝑥𝑖) = 𝑦𝑖 для всех *i = 0, 1. …, n*.

Значение 𝑠𝑖 = 𝑆(1)(𝑥𝑖) называется наклоном сплайна в точке 𝑥𝑖.

𝑚

Заметим, что на отрезке [𝑥𝑖−1*,* 𝑥𝑖] интерполяционный кубический сплайн однозначно определяется заданием значений 𝑦𝑖−1, 𝑦𝑖, 𝑠𝑖−1, 𝑠𝑖. В самом деле, из равенства (6.30) для интерполяционного многочлена Эрмита с кратными узлами

𝑃 *(x) =* 𝑦

(𝑥1−𝑥)2(2(𝑥− 𝑥0)+ℎ) *+* 𝑦′ (𝑥1−𝑥)2(𝑥− 𝑥0) *+*

3 0 ℎ3 0 ℎ2

+ 𝑦

(𝑥−𝑥0 )2(2(𝑥1−𝑥)+ℎ)

*+* 𝑦′ (𝑥−𝑥0)2(𝑥− 𝑥1) *,* где *h* = 𝑥

− 𝑥 ,

1 ℎ3 1 ℎ2 1 0

вытекает следующая формула:

𝑆 *(x) =* 𝑃

(𝑥) = (𝑥−𝑥𝑖)2(2(𝑥− 𝑥𝑖−1)+ℎ𝑖) 𝑦

*+*(𝑥−𝑥𝑖−1 )2(2(𝑥𝑖−𝑥)+ℎ𝑖) 𝑦 *+*

3 3,𝑖

ℎ

3 𝑖−1

𝑖

ℎ

3 𝑖

𝑖

*+*(𝑥−𝑥𝑖)2(𝑥− 𝑥𝑖−1) 𝑠 *+* (𝑥−𝑥𝑖−1)2(𝑥− 𝑥𝑖) 𝑠 *.* (6.64)

ℎ

ℎ

2 𝑖−1

𝑖

2 𝑖

𝑖

Здесь ℎ𝑖 = 𝑥𝑖 − 𝑥𝑖−1. Справедливость этой формулы проверяется непосредственно заменой соответствующих обозначений применительно к конкретному отрезку [𝑥𝑖−1*,* 𝑥𝑖].

Различные методы интерполяции кубическими сплайнами отличаются один от другого способом выбора наклонов 𝑠𝑖. Рассмотрим некоторые из них.

1. **Локальный сплайн.** Если в точках 𝑥𝑖 известны значения

производной 𝑦ʹ = 𝑓ʹ(𝑥𝑖) , то естественно в формуле положить 𝑠𝑖 = 𝑦ʹ

𝑖 𝑖

для всех *i = 0, 1. …, n*. Тогда на каждом частичном отрезке [𝑥𝑖−1*,* 𝑥𝑖] в соответствии с формулой (6.64) сплайн однозначно определяется

значениями 𝑦𝑖−1, 𝑦𝑖, 𝑦ʹ 𝑦ʹ (поэтому его и называют *локальным*

𝑖−1, 𝑖

*сплайном*). Заметим, что он совпадает с кубическим интерполяционным многочленом Эрмита (6.30) для отрезка [𝑥𝑖−1*,* 𝑥𝑖].

Из неравенства (6.32) для погрешности интерполяции с кратными узлами, которое имеет вид

max | *f ( x)* − 𝑃3*( x )* | *≤*𝑀4

ℎ4,

[𝑥0,𝑥1]

384

получается следующая оценка погрешности интерполяции локальным кубическим сплайном:

max| *f ( x)* − 𝑆3*( x )* | *≤*𝑀4

ℎ4 , (6.65)

[𝑎,𝑏]

384

𝑚𝑎𝑥

где ℎ𝑚𝑎𝑥 = max1≤𝑖≤𝑛 ℎ𝑖 – максимальная из длин частичных отрезков.

Заметим, что для построенного указанным образом сплайна можно гарантировать непрерывность на отрезке [*a, b*] только функции 𝑆3(*x*) и ее первой производной 𝑆ʹ (*x*), т.е. его дефект равен 2.

3

Существуют и другие способы выбора коэффициентов 𝑠𝑖, приводящие к локальным сплайнам, например, кубический многочлен Бесселя и др.

1. **Глобальные способы построения кубических сплайнов.** Для того чтобы сплайн 𝑆3(*x*) имел непрерывную на [*a, b*] вторую производную 𝑆ʹʹ(*x*), необходимо выбирать наклоны 𝑠𝑖 так, чтобы в точках 𝑥𝑖 стыка многочленов 𝑃3,𝑖 и 𝑃3,𝑖+1 совпадали значения их вторых производных:

3

𝑃ʹʹ (𝑥𝑖) = 𝑃ʹʹ (𝑥𝑖), 𝑖 = 1, 2, … , 𝑛 − 1 (6.66)

3,𝑖 3,𝑖+1

Пользуясь формулой (6.64), найдем значение

𝑃ʹʹ (𝑥𝑖) = 2𝑠𝑖−1 + 4𝑠𝑖 – 6 𝑦𝑖−𝑦𝑖−1 . (6.67)

3,𝑖

ℎ

𝑖

ℎ𝑖

ℎ𝑖 2

Из подобной формулы, записанной для многочлена 𝑃3,𝑖+1, имеем

𝑃ʹʹ (𝑥𝑖) = −  4𝑠𝑖 − 2𝑠𝑖+1 + 6 𝑦𝑖+1−𝑦𝑖 . (6.68)

3,𝑖+1

ℎ

ℎ𝑖+1

ℎ𝑖+1

2

𝑖+1

Таким образом, равенства (6.66) приводят к следующей системе уравнений относительно коэффициентов 𝑠𝑖:

ℎ−1𝑠𝑖−1 + 2(ℎ−1 + ℎ−1 ) 𝑠𝑖 + ℎ−1 𝑠𝑖+1 =

𝑖 𝑖

𝑖+1

𝑖+1

=3[ℎ−2(𝑦𝑖 − 𝑦𝑖−1) + ℎ−2 (𝑦𝑖+1 − 𝑦𝑖)], *i=1,2,…,n*–*1.* (6.69)

𝑖 𝑖+1

Заметим, что эта система уравнений недоопределена, так как число уравнений системы (равное *n* – 1) меньше числа неизвестных (равного *n* + 1). Выбор двух оставшихся уравнений обычно связывают с некоторыми дополнительными условиями, накладываемыми на сплайн в граничных точках *a* и *b* (*граничными условиями*). Укажем на некоторые из наиболее известных граничных условий.

1. Если в граничных точках известны значения первой производной 𝑓ʹ(𝑎) и 𝑓ʹ(𝑏), то естественно положить

𝑠0 = 𝑓ʹ(𝑎), 𝑠𝑛 = 𝑓ʹ(𝑏). (6.70)

Дополняя систему (6.69) уравнениями (6.70), приходим к системе уравнений с трехдиагональной матрицей, которая легко решается методом прогонки. Полученный таким образом сплайн называют *фундаментальным кубическим сплайном*.

1. Если в граничных точках известны значения второй производной 𝑓ʹʹ(𝑎) и 𝑓ʹʹ(𝑏), то можно наложить на сплайн граничные

условия 𝑆ʹʹ(*a*) = 𝑃ʹʹ (𝑥0) = 𝑓ʹʹ(𝑎), 𝑆ʹʹ(*b*) = 𝑃ʹʹ (𝑥𝑛) = 𝑓ʹʹ(𝑏), что приводит

3 3,1 3 3,𝑛

к следующим уравнениям:

− 4𝑠0 − 2𝑠1 + 6 𝑦1−𝑦0

= 𝑓ʹʹ(𝑎), (6.71)

ℎ1 ℎ1 2

ℎ

1

2𝑠 𝑛−1 + 4𝑠𝑛 – 6 𝑦𝑛−𝑦𝑛−1

= 𝑓ʹʹ(𝑏) (6.72)

ℎ𝑛

ℎ

𝑛

ℎ𝑛 2

(достаточно в равенстве (6.68) взять *i* = 0, а в равенстве (6.69) *i* = *n* ).

3). Полагая в уравнениях (6.71), (6.72) 𝑓ʹʹ(𝑎) = 0, 𝑓ʹʹ(𝑏) = 0 (независимо от того, выполнены ли эти условия для интерполируемой функции), придем к системе уравнений, определяющих так называемый *естественный кубический сплайн*.

4). Если *f –* периодическая функция с периодом, равным *b – a,* то систему (6.69) следует дополнить условиями

𝑠0 = 𝑠𝑛,

ℎ−1(𝑠

+ 2𝑠 )+ ℎ−1(2𝑠 + 𝑠

) = 3[ℎ−2(𝑦

− 𝑦

) + ℎ−2(𝑦 − 𝑦

)].

𝑛 𝑛−1

𝑛 1 0 1

𝑛 𝑛

𝑛−1

1 1 0

Существуют и другие подходы к заданию граничных условий.

# Погрешность приближения кубическими сплайнами.

Доказано, что если функция *f* имеет на отрезке [*a, b*] непрерывную производную четвертого порядка и 𝑀4 = max[𝑎,𝑏] |𝑓(4)(𝑥)|, то для интерполяционного кубического сплайна 𝑆3(*x*) справедлива следующая оценка погрешности:

max |𝑓(𝑥) − 𝑆3(𝑥)| ≤ C𝑀 ℎ4 . (6.73)

[𝑎,𝑏]

4 𝑚𝑎𝑥

Заметим, что сплайн 𝑆3(*x*) не только сам аппроксимирует функцию

*f*(*x*), но и его производные 𝑆ʹ (*x*), 𝑆ʹʹ(*x*) и 𝑆(3)(*x*) приближают

3 3 3

соответствующие производные функции *f.* В частности, справедливы неравенства:

max |𝑓(𝑘)(𝑥) − 𝑆(𝑘)(𝑥)| ≤ 𝐶𝑘𝑀

ℎ4−𝑘 . (6.74)

[𝑎,𝑏]

3 4 𝑚𝑎𝑥

# Лекция 14 Тема 7. Численное дифференцирование

Численное дифференцирование применяется тогда, когда функцию трудно или невозможно продифференцировать аналитически. Например, необходимость в численном дифференцировании возникает в том случае, когда функция задана таблицей. Кроме того, формулы численного дифференцирования широко используются при разработке вычислительных методов решения многих задач (решение дифференциальных уравнений, поиск решений нелинейных уравнений, поиск точек экстремума функций и др.).

# § 7.1. Простейшие формулы численного дифференцирования

1. **Вычисление первой производной.** Предположим, что в окрестности точки *x* функция *f* дифференцируема достаточное число раз. Исходя из определения производной

𝑓ʹ(𝑥) = lim𝛥𝑥→0 𝑓(𝑥+𝛥𝑥)−𝑓(𝑥)*,*

𝛥𝑥

естественно попытаться использовать для ее вычисления простейшие приближенные формулы:

𝑓ʹ(𝑥) ≈ 𝑓(𝑥+ℎ)−𝑓(𝑥)*,* (7.1)

ℎ

𝑓ʹ(𝑥) ≈ 𝑓(𝑥)−𝑓(𝑥−ℎ)*,* (7.2)

ℎ

соответствующие выбору фиксированных значений 𝛥𝑥 = ℎ и 𝛥𝑥 = −ℎ. Здесь ℎ > 0 – малый параметр (шаг). Разностные отношения в правых частях формул (7.1) и (7.2) часто называют *правой* и *левой разностными производными*.

Для оценки погрешностей

𝑟+(𝑥, ℎ) = 𝑓ʹ(𝑥) − 𝑓(𝑥+ℎ)−𝑓(𝑥) , 𝑟−(𝑥, ℎ) = 𝑓ʹ(𝑥) − 𝑓(𝑥)−𝑓(𝑥−ℎ)

ℎ ℎ

введенных формул численного дифференцирования (*погрешностей аппроксимации*) воспользуемся формулами Тейлора:

*f* (*x ± h*) *= f*(*x*) *±* 𝑓ʹ*(x)h +* 𝑓ʹʹ(𝜉±) ℎ2. (7.3)

2

Здесь и ниже 𝜉+ и 𝜉− - некоторые точки, расположенные на интервалах (*x, x + h*) и (*x – h, x*) соответственно. Подставляя

разложения (7.3) в выражения для 𝑟±, получаем 𝑟+(𝑥, ℎ) = *–* 1 𝑓ʹʹ(𝜉+)*h,*

2

𝑟−(𝑥, ℎ) = 1 𝑓ʹʹ(𝜉−)*h.* Следовательно,

2

| 𝑟+(𝑥, ℎ) | ≤ 1 𝑀2ℎ, 𝑀2 = max[𝑥,𝑥+ℎ] |𝑓ʹʹ(𝜉)|, (7.4)

2

| 𝑟−(𝑥, ℎ) | ≤ 1 𝑀2ℎ, 𝑀2 = max[𝑥−ℎ,𝑥] |𝑓ʹʹ(𝜉)| (7.5)

2

Таким образом, формулы (7.1) и (7.2) имеют первый порядок точности по ℎ. Иначе говоря, правая и левая разностные производные аппроксимируют производную 𝑓ʹ(𝑥) с первым порядком точности.

Приведенные формулы численного дифференцирования имеют простую геометрическую интерпретацию (рис. 7.1):

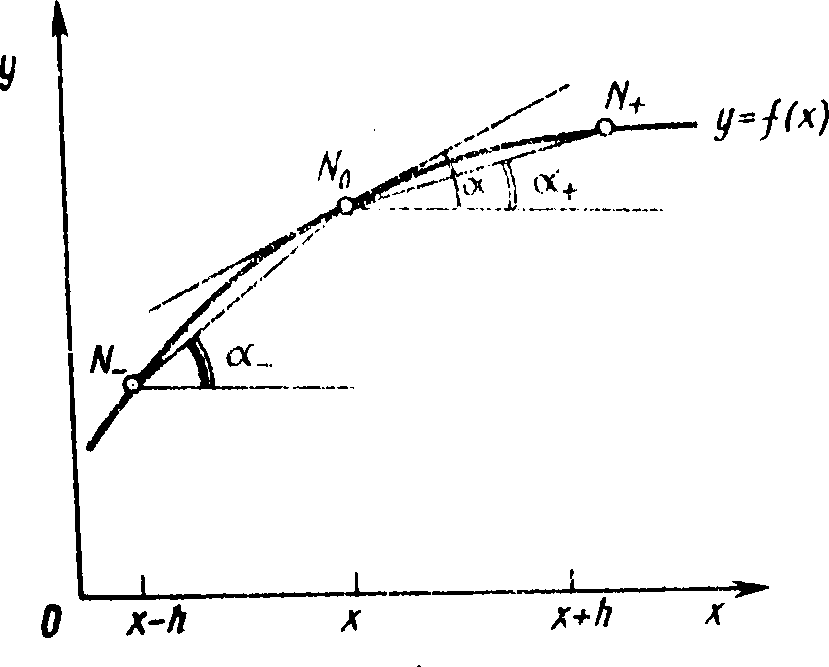


Рис 7.1

Пусть 𝑁0, 𝑁− и 𝑁+ *–* расположенные на графике функции *y* = *f* (*x*) точки с координатами *(x, f(x)), (x – h, f(x – h))* и *(x + h, f(x + h)).* Напомним, что производная 𝑓ʹ(*x*) равна тангенсу угла *α* наклона к оси *Ox* касательной, проведенной к графику функции в точке 𝑁0. Формула (7.1) соответствует приближенной замене производной 𝑓ʹ(*x*) = tg *α*

правой разностной производной 𝑓(𝑥+ℎ)−𝑓(𝑥)*,* равной тангенсу угла 𝛼

+

ℎ

наклона секущей, проведенной через точки 𝑁0 и 𝑁+. Формула (7.2) соответствует аналогичной замене производной 𝑓ʹ(*x*) левой разностной

производной 𝑓(𝑥)−𝑓(𝑥−ℎ)*,* равной тангенсу угла 𝛼 наклона секущей,

−

ℎ

проведенной через точки 𝑁0 и 𝑁−.

Естественно предположить (рис.7.2), что лучшим по сравнению с tg𝛼+ и tg𝛼− приближением к 𝑓ʹ(*x*) = tg *α* является тангенс угла наклона 𝛼0 секущей, проведенной через точки 𝑁− и 𝑁+.

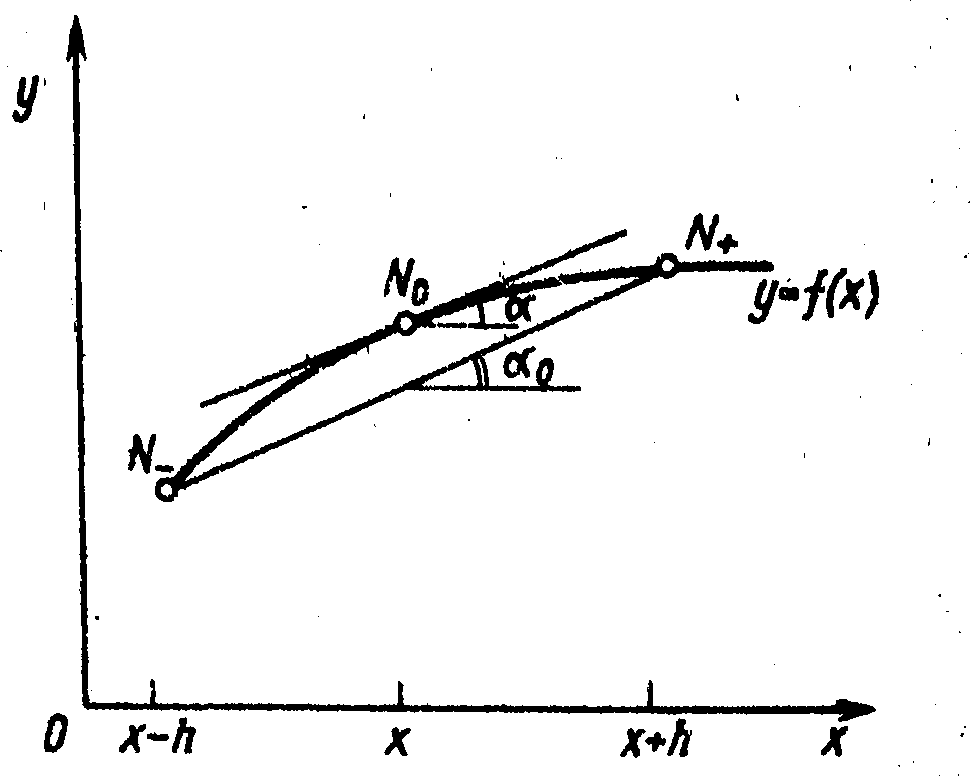


Рис. 7.2

Соответствующая приближенная формула имеет вид

𝑓ʹ(𝑥) ≈ 𝑓(𝑥+ℎ)−𝑓(𝑥−ℎ)*.* (7.6)

2ℎ

Величину в правой части этой формулы часто называют *центральной разностной производной*.

Подставляя в выражение для погрешности

𝑟 (𝑥, ℎ) = 𝑓ʹ(𝑥) − 𝑓(𝑥+ℎ)−𝑓(𝑥−ℎ)

0

2ℎ

соответствующие разложения по формуле Тейлора

*f* (*x ± h*) *= f*(*x*) *±* 𝑓ʹ*(x)h +* 𝑓ʹʹ(𝑥) ℎ2 ± 𝑓(3)(𝜉±) ℎ3,

получим 𝑟0

2

(𝑥, ℎ) = − 𝑓(3)(𝜉+)+ 𝑓(3)(𝜉−)

12

6

ℎ2. Следовательно, справедлива

оценка погрешности

| 𝑟0(𝑥, ℎ) | ≤ 1 𝑀3ℎ2, 𝑀3 = max[𝑥−ℎ, 𝑥+ℎ] |𝑓(3)(𝜉)|. (7.7)

6

Таким образом, центральная разностная производная аппроксимирует производную 𝑓ʹ(*x*) со вторым порядком точности относительно ℎ.

Для вычисления 𝑓ʹ(*x*) можно получить формулы любого порядка точности. Однако в таких формулах с ростом порядка точности возрастает и число используемых значений функции. В качестве примера приведем формулу

𝑓ʹ(𝑥) ≈ 𝑓(𝑥−2ℎ)−8𝑓(𝑥−ℎ)+8𝑓(𝑥+ℎ)− 𝑓(𝑥+2ℎ) *,* (7.8)

12ℎ

имеющую четвертый порядок точности.

1. **Вычисление второй производной.** Наиболее простой и широко применяемой для приближенного вычисления второй производной является следующая формула:

𝑓ʹʹ(𝑥) ≈ 𝑓(𝑥−ℎ)−2𝑓(𝑥)+𝑓(𝑥+ℎ) *.* (7.9)

2

ℎ

Величину в правой части этого приближенного равенства часто называют *второй разностной производной.*

Подставляя в выражение для погрешности

𝑟(𝑥, ℎ) = 𝑓ʹʹ(𝑥) − 𝑓(𝑥−ℎ)−2𝑓(𝑥)+𝑓(𝑥+ℎ)

2

ℎ

соответствующие разложения по формуле Тейлора

*f* (*x ± h*) *= f*(*x*) *±* 𝑓ʹ*(x)h +* 𝑓ʹʹ(𝑥) ℎ2 ± 𝑓(3)(𝑥) ℎ3*+* 𝑓(4)(𝜉±) ℎ4*,*

2 6 24

получим 𝑟(𝑥, ℎ) = − 𝑓(4)(𝜉+)+𝑓(4)(𝜉−) ℎ2. Следовательно,

24

| 𝑟(𝑥, ℎ)| ≤ 𝑀4 ℎ2, 𝑀4 = max[𝑥−ℎ, 𝑥+ℎ] |𝑓(4)(𝜉)|. (7.10) Таким образом, формула (7.9) имеет второй порядок точности.

12

Для вычисления 𝑓ʹʹ(*x*) можно получить формулы любого порядка

точности. Например, формула

𝑓ʹʹ(𝑥) ≈ −𝑓(𝑥−2ℎ)+16𝑓(𝑥−ℎ)−30𝑓(𝑥)+ 16𝑓(𝑥+ℎ) −𝑓(𝑥+2ℎ)

2

12ℎ

(7.11)

имеет четвертый порядок точности.

# § 7.2. Об общем подходе к выводу формул численного дифференцирования

Хотя простейшие формулы численного дифференцирования можно получить сравнительно элементарно, для вывода и анализа таких формул в более сложных случаях необходимо использовать значительно более серьезный математический аппарат. Основой для построения различных приближенных формул численного дифференцирования являются методы теории приближения функций, элементы которой были изложены в предыдущей главе.

Предположим, что в окрестности точки *x* функция *f* аппроксимируется некоторой другой функцией *g*, причем производная 𝑔(𝑘) в точке *x* легко вычисляется.

Естественно в такой ситуации воспользоваться приближенной

формулой

𝑓(𝑘)(𝑥) ≈ 𝑔(𝑘)(𝑥). (7.12)

Наиболее просто этот подход реализуется в случае, когда приближение осуществляется с помощью интерполяции.

1. **Формулы численного дифференцирования, основанные на интерполяции алгебраическими многочленами.** Пусть 𝑃𝑛(𝑥) - интерполяционный многочлен степени *n* с узлами интерполяции

𝑥0 < 𝑥1 < ⋯ < 𝑥𝑛 и *x* ϵ[𝑥0, 𝑥𝑛]. В этом случае формула (7.11) принимает вид

𝑓(𝑘)(𝑥) ≈ 𝑃(𝑘)(𝑥), 0 ≤ *k ≤ n.* (7.13) При этом справедлива следующая оценка погрешности формулы (7.13):

𝑛

| 𝑓(𝑘)(𝑥) − 𝑃(𝑘)(𝑥)| ≤ 𝐶𝑛,𝑘𝑀 ℎ𝑛+1−𝑘, 0 ≤ 𝑘 ≤ 𝑛. (7.14)

𝑛 𝑛+1 𝑚𝑎𝑥

Здесь 𝐶𝑛,𝑘 – положительные числа, а 𝑀𝑛+1=max[𝑥 ,𝑥 ] |𝑓(𝑛+1)(𝑥)|.

0 𝑛

З а м е ч а н и е 1. Порядок точности формулы (7.13) относительно ℎ𝑚𝑎𝑥 равен разности между числом узлов интерполяции и порядком вычисляемой производной.

З а м е ч а н и е 2. Если формула (7.13) применяется для вычисления производной в точке, относительно которой узлы таблицы расположены симметрично, и число 𝑛 − 𝑘 четно, то порядок точности формулы повышается на единицу по сравнению с порядком 𝑛 + 1 − 𝑘, гарантируемым оценкой (7.14). Таковы, например, формулы (7.6), (7,8),

(7.9), (7.11).

1. **Подход, основанный на использовании сплайнов.** Применение формулы (7.13) для вычисления производной 𝑓(𝑘) фактически основано на кусочно-полиномиальной интерполяции. Полученная таким образом производная в точке стыка двух соседних многочленов может иметь разрыв. Поэтому, если требуется глобально на отрезке [*a, b*] аппроксимировать производную гладкой функцией, то

целесообразно использовать сплайны. Производная 𝑆(𝑘)(𝑥) сплайна

𝑚

𝑆𝑚(𝑥) при *k ≤ m – r* ( где *r* – дефект сплайна) дает гладкую глобальную аппроксимацию для 𝑓(𝑘)(𝑥).

# §7.3. Обусловленность формул численного дифференцирования

Несмотря на внешнюю простоту формул численного дифференцирования, их применение требует особой осторожности. Отметим, что используемые при численном дифференцировании значения 𝑓∗(𝑥) функции *f* (*x*) непременно содержат ошибки. Поэтому к погрешности аппроксимации формул численного дифференцирования добавляется неустранимая погрешность, вызванная погрешностями вычисления функции *f.* Для того, чтобы погрешность аппроксимации была достаточно малой, требуется использование таблиц с малыми шагами *h*. Однако, к сожалению, при малых шагах формулы численного дифференцирования становятся плохо обусловленными и результат их применения может быть полностью искажен неустранимой ошибкой. Важно понимать, что действительная причина этого явления лежит не в несовершенстве используемых методов вычисления производных, а в заданной функции (см. пример 3.5).

Поясним сказанное на примере формулы (7.1). Полная

погрешность 𝑟∗(𝑥, ℎ) = 𝑓ʹ(𝑥) − 𝑓∗(𝑥+ℎ)−𝑓∗(𝑥)

ℎ

реально вычисляемого

значения правой разностной производной представляет собой сумму

погрешности аппроксимации 𝑟 (𝑥, ℎ) = 𝑓ʹ(𝑥) − 𝑓(𝑥+ℎ)−𝑓(𝑥) и

+

ℎ

неустранимой погрешности

𝑟 (𝑥, ℎ) = 1 ((*f(x+h) –* 𝑓∗(𝑥 + ℎ)) − ( 𝑓(𝑥) − 𝑓∗(𝑥))). ℎ

н

Пусть 𝛥̅ *–* верхняя граница абсолютной погрешности 𝛥(𝑓∗(𝑥)) =

= |𝑓(𝑥) − 𝑓∗(𝑥)| используемых значений функции. Тогда погрешность оценивается следующим образом:

|𝑟 | ≤ 2𝛥̅ . (7.15)

н ℎ

Оценка (7.15) означает, что чувствительность формулы (7.1) к погрешностям входных данных характеризуется абсолютным числом

обусловленности 𝜈𝛥 = 2 .

ℎ

Так как 𝜈𝛥 → ∞ при *h* → 0, то формула(7.1) при малых ℎ становится очень плохо обусловленной. Поэтому несмотря на то, что погрешность аппроксимации стремится к нулю при *h* → 0 (см. оценку (7.4)), следует ожидать, что полная погрешность будет неограниченно возрастать при *h* → 0*.* Во всяком случае так ведет себя верхняя граница полной

погрешности 𝑟̅(ℎ) = 1 𝑀 *h +* 2𝛥̅ .

2 2 ℎ

Выберем оптимальное значение шага *h*, при котором величина

𝑟̅(ℎ) достигает минимального значения.

*–*

Приравнивая производную 𝑟̅ʹ(*h*) = 1

2

𝑀2

2𝛥̅ ℎ2

к нулю, получаем

значение ℎопт

= 2√ 𝛥̅

𝑀2

, которому отвечает величина

𝑟𝑚

𝑖𝑛 = 𝑟̅(ℎопт) = 2 √𝛥̅ 𝑀2 .

Таким образом, при использовании формулы (7.1) для производной функции 𝑓, заданной с погрешностью, следует обратить внимание на выбор шага *h.* Однако даже при оптимальном выборе шага полная погрешность окажется величиной, пропорциональной лишь

√𝛥̅.

Формулы для вычисления производных порядка *k > 1* обладают еще большей чувствительностью к ошибкам задания функций. Поэтому значения производных высокого порядка, найденные с помощью таких формул, могут быть очень неточными.

# Лекция 15 Тема 8. Численное интегрирование

**§ 8.1. Простейшие квадратурные формулы**

1. **Постановка задачи.** В прикладных исследованиях часто возникает задача вычисления значения определенного интеграла

I = ∫𝑏 𝑓(𝑥)𝑑𝑥

𝑎

(8.1)

Этот интеграл может выражать площадь, объем, работу переменной силы и т.д.

Если функция 𝑓(𝑥) непрерывна на отрезке [a, b] и ее первообразную *F(x)* удается выразить через известные функции, то для вычисления интеграла (8.1) можно воспользоваться *формулой Ньютона-Лейбница:*

∫𝑏 𝑓(𝑥)𝑑𝑥 *= F(b) - F(a).* (8.2)

𝑎

К сожалению, в подавляющем большинстве случаев получить значение определенного интеграла с помощью формулы (8.2) или других аналитических методов не удается.

Пример 8.1. Интеграл ∫𝑥 𝑒−𝑡2 𝑑𝑡

0

широко используется при

исследовании процессов теплообмена и диффузии, в статистической физике и теории вероятностей. Однако его значение не может быть выражено в виде конечной комбинации элементарных функций.

Заметим, что даже в тех случаях, когда удается получить первообразную функцию *F(x)* в аналитической форме, значительные усилия, затраченные на это, часто оказываются чрезмерно высокой платой за окончательный результат. Добавим еще, что вычисление интеграла в этих случаях по формуле (8.2), как правило, приводят к громоздким (а часто – и приближенным) вычислениям. Следует отметить также, что зачастую найти точное значение интеграла (8.1)

просто невозможно. Например, это имеет место, когда функция 𝑓(𝑥)

задается таблицей своих значений.

Обычно для вычисления значения определенного интеграла применяют специальные численные методы. Наиболее широко используют на практике *квадратурные формулы* – приближенные равенства вида

∫𝑏 𝑓(𝑥)𝑑𝑥 ≈ ∑𝑁

𝐴𝑖 *f*(𝑥̅𝑖). (8.3)

𝑎 𝑖=0

Здесь 𝑥̅𝑖

– некоторые точки из отрезка [a, b] – *узлы квадратурной*

*формулы*; 𝐴𝑖 – числовые коэффициенты, называемые *весами*

*квадратурной формулы; N ≥ 0* – целое число. Сумма ∑𝑁 𝐴𝑖 *f*(𝑥̅𝑖),

𝑖=0

которая принимается за приближенное значение интеграла, называется

*квадратурной суммой*. Величина *R =* ∫𝑏 𝑓(𝑥)𝑑𝑥

𝑎

𝑁

𝑖=0

– ∑

𝐴𝑖 *f*(𝑥̅𝑖)

называется *погрешностью* (или *остаточным членом*) *квадратурной формулы*.

Будем говорить, что квадратурная формула (8.3) точна для многочленов степени *m*, если для любого многочлена степени не выше *m* эта формула дает точное значение интеграла, т.е.

∫𝑏 𝑃𝑚(𝑥)𝑑𝑥 ≈ ∑𝑁

𝐴𝑖

𝑃𝑚(𝑥̅𝑖).

𝑎 𝑖=0

При оценке эффективности квадратурных формул часто исходят из того, что наиболее трудоемкой операцией при вычислении по формуле (8.3) является нахождение значения функции 𝑓. Поэтому среди двух формул, позволяющих вычислить интеграл с заданной точностью ε, более эффективной считается та, в которой используется меньшее количество узлов.

Выведем простейшие квадратурные формулы, исходя из наглядных геометрических соображений. Будем интерпретировать интеграл (8.1) как площадь криволинейной трапеции, ограниченной графиком функции *y = f* (*x*) (при *f* (*x*) *≥ 0*)*,* осью абсцисс и прямыми

*x = а, x = b* (рис. 8.1,*а*).

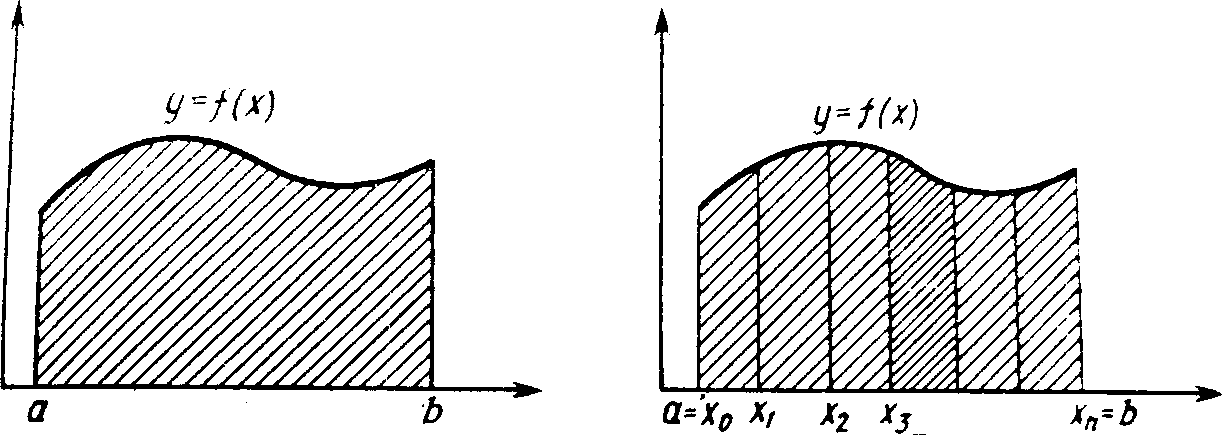


Рис.8.1,а Рис.8.1,б

Разобьем отрезок [*a, b*] на элементарные отрезки [𝑥𝑖−1, 𝑥𝑖] точками *a =* 𝑥0 < 𝑥1 < ⋯ < 𝑥𝑛 = 𝑏. Интеграл I разобьется при этом на сумму элементарных интегралов:

𝑛

I = ∑

𝑖=1

где 𝐼 = 𝑥𝑖

𝑖 ∫

𝑥𝑖−1

𝐼𝑖, (8.4)

𝑓(𝑥)𝑑𝑥, что соответствует разбиению площади исходной

криволинейной трапеции на сумму площадей элементарных криволинейных трапеций (рис. 8.1,б).

Введем обозначения: 𝑓𝑖 = *f*(𝑥𝑖), 𝑓𝑖−1 = *f* (𝑥𝑖−1), где 𝑥𝑖−1 =

2 2 2

= (𝑥𝑖−1 + 𝑥𝑖)/2 – середина элементарного отрезка. Для простоты шаг

*h* = 𝑥𝑖 − 𝑥𝑖−1 будем считать постоянным.

1. **Формула прямоугольников.** Заменим приближенно площадь элементарной криволинейной трапеции площадью прямоугольника, основанием которого является отрезок [𝑥𝑖−1, 𝑥𝑖], а

высота равна значению 𝑓𝑖−1 (на рис. 8.2, а через 𝑁𝑖−1 обозначена точка

2 2

с координатами (𝑥𝑖−1, 𝑓𝑖−1 )). Так мы приходим к *элементарной*

2 2

*квадратурной формуле прямоугольников*:

𝐼𝑖 ≈ *h*𝑓𝑖−1*.* (8.5)

2

Произведя такую замену для всех элементарных криволинейных трапеций, получаем *составную квадратурную формулу прямоугольников*:

*I ≈* 𝐼ℎ *= h* (𝑓

*+*𝑓

+ ⋯ + 𝑓

) = *h* ∑𝑛 𝑓

*.* (8.6)

пр 1/2

3/2

𝑛−1/2

𝑖=1 𝑖−1/2

Эта формула соответствует приближенной замене площади исходной криволинейной трапеции площадью ступенчатой фигуры, изображенной на рис. 8.2, б.

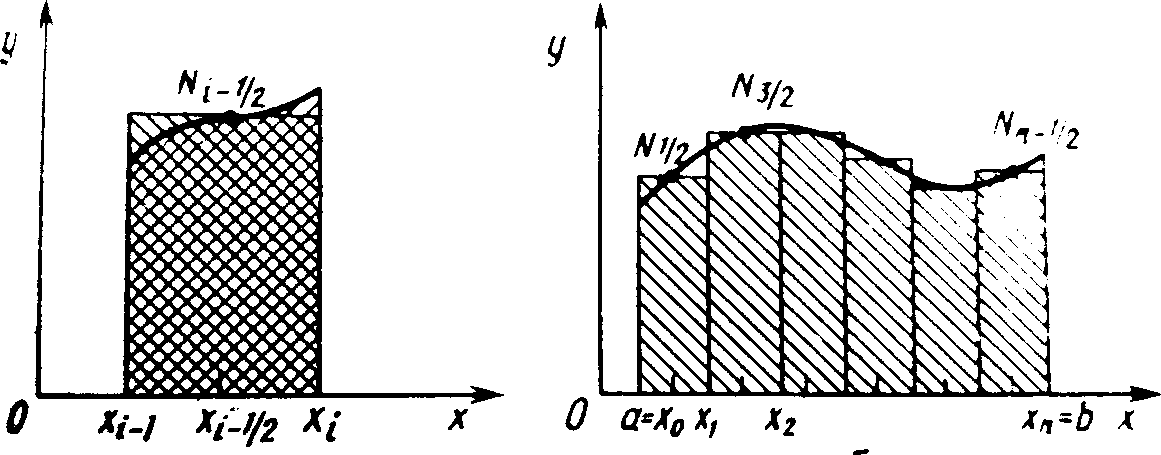


Рис.8.2,а Рис.8.2,б

1. **Формула трапеций.** Соединив отрезком точки 𝑁𝑖−1(𝑥𝑖−1, 𝑓𝑖−1) и

𝑁𝑖(𝑥𝑖, 𝑓𝑖) на графике функции *y = f* (*x*), получим трапецию (рис. 8.3,а). Заменим теперь приближенно площадь элементарной криволинейной трапеции площадью построенной фигуры. Тогда получим *элементарную квадратурную формулу трапеций*:

𝐼𝑖 ≈ ℎ

2

(𝑓𝑖−1 + 𝑓𝑖). (8.7)

Пользуясь этой формулой для *i = 1, 2, … , n* выводим *составную квадратурную формулу трапеций*:

𝐼 ≈ 𝐼ℎ = *h* (𝑓0 + 𝑓 + 𝑓 + ⋯ + 𝑓 + 𝑓𝑛) =

тр 2 1 2 𝑛−1 2

= *h* (𝑓0+𝑓𝑛 + ∑𝑛−1 𝑓𝑖). (8.8)

2 𝑖=1

Эта формула соответствует приближенной замене площади исходной криволинейной трапеции площадью фигуры, ограниченной ломаной линией, проходящей через точки 𝑁0, 𝑁1, … , 𝑁𝑛 (рис.8.3,б).

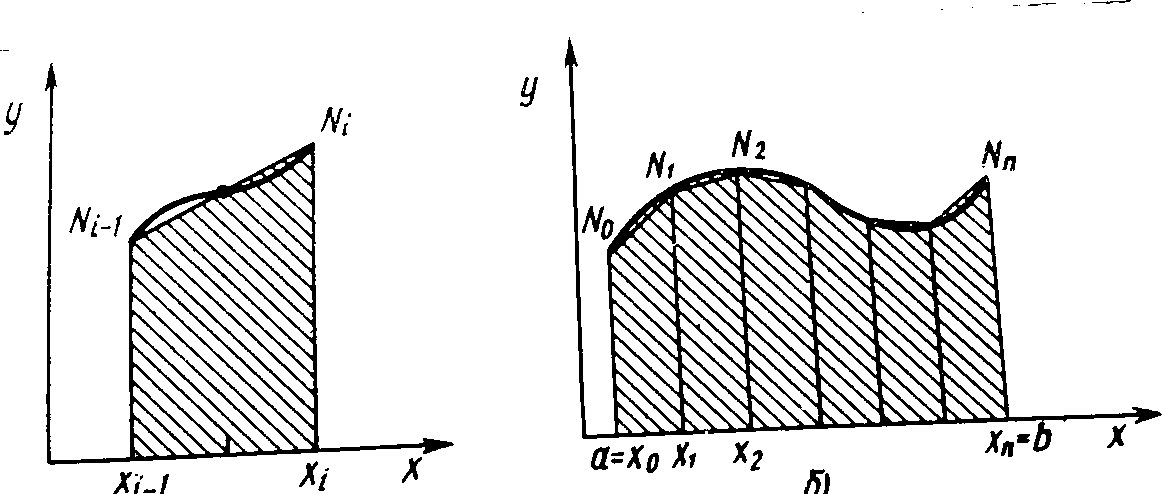


Рис. 8.3,а Рис. 8.3,б

1. **Формула Симпсона.** Если площадь элементарной криволинейной трапеции заменить площадью фигуры, расположенной под параболой, проходящей через точки 𝑁𝑖−1, 𝑁𝑖−1/2, 𝑁𝑖 (рис.8.4,а), то

получим приближенное равенство 𝐼 ≈ 𝑥𝑖

𝑖 ∫

𝑥𝑖−1

𝑃2(𝑥)𝑑𝑥.

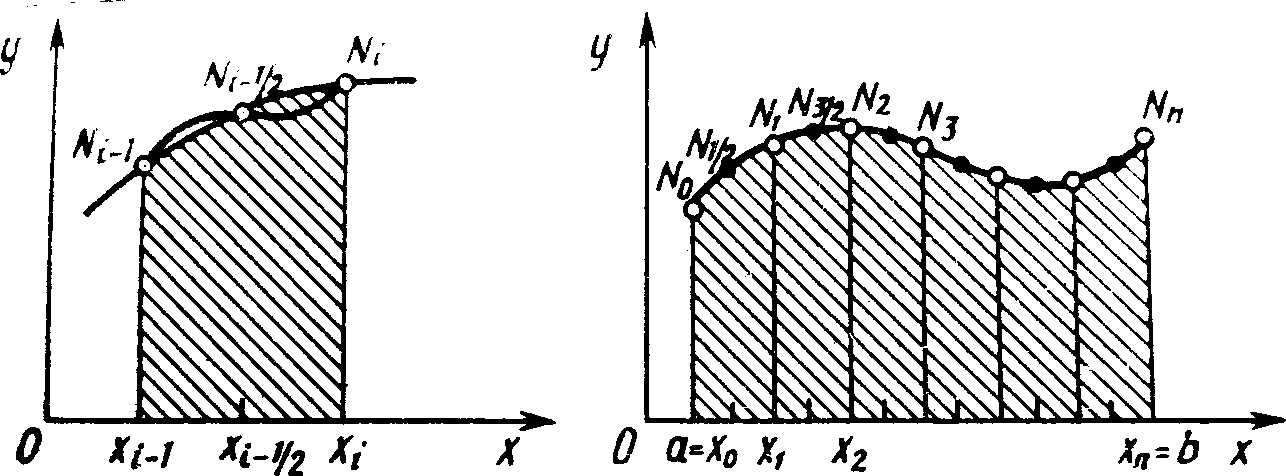


Рис.8.4,а Рис.8.4,б

Здесь 𝑃2(𝑥) – интерполяционный многочлен второй степени с узлами

𝑥𝑖−1, 𝑥𝑖−1/2, 𝑥𝑖 . Как нетрудно убедиться, верна формула

𝑓𝑖− 2𝑓

1 + 𝑓𝑖−1

𝑃 (𝑥) = 𝑓 + 𝑓𝑖− 𝑓𝑖−1 (*x* – 𝑥 ) +

𝑖−2

(*x* – 𝑥

)2.

2 𝑖−1/2 ℎ

𝑖−1/2

ℎ2 /2

𝑖−1/2

Ее интегрирование приводит к равенству

∫𝑥𝑖

𝑃2(𝑥)𝑑𝑥 = ℎ 𝑓𝑖−1/2 + 𝑓𝑖− 𝑓𝑖−1 ∫𝑥𝑖

(𝑥 – 𝑥

1) 𝑑𝑥 +

𝑥𝑖−1

ℎ 𝑥𝑖−1

𝑖−2

𝑓𝑖− 2𝑓

1 + 𝑓𝑖−1

𝑖−2

∫

2

+

ℎ2 /2

𝑥𝑖

𝑥𝑖−1

(𝑥 − 𝑥

𝑖−1)2 𝑑𝑥 =

ℎ

= ℎ 𝑓 + 6 (𝑓 − 2𝑓𝑖−

𝑖−1/2 𝑖 1

2

ℎ

𝑖−1 𝑖−1 1

+ 𝑓 ) = 6 (𝑓 + 4𝑓𝑖−

2

+ 𝑓𝑖)

Таким образом, выведена *элементарная квадратурная формула Симпсона:*

ℎ

𝐼 ≈ 6 (𝑓 + 4𝑓𝑖−

𝑖 𝑖−1 1

2

+ 𝑓𝑖). (8.9)

Применяя эту формулу на каждом элементарном отрезке, выводим

*составную квадратурную формулу Симпсона*:

*I* ≈ 𝐼ℎ = ℎ (𝑓0 + 4𝑓1 + 2𝑓1 + 4𝑓3

+ 2𝑓2 + ⋯ + 2𝑓𝑛−1 + 4𝑓

1 + 𝑓𝑛) =

𝐶 6 2 2

𝑛−2

= ℎ (𝑓0 + 𝑓𝑛 + 4 ∑𝑛

𝑓 1

+ 2 ∑𝑛−1 𝑓𝑖). (8.10)

6 𝑖=1

𝑖−2

𝑖=1

Учитывая геометрическую интерпретацию формулы Симпсона, ее иногда называют *формулой парабол*.

1. **Оценка погрешности.** Оценим погрешность выведенных формул в предположении, что подынтегральная функция *f* достаточно гладкая. Как и в предыдущих разделах, будем использовать обозначение 𝑀𝑘 = max[𝑎,𝑏] |𝑓(𝑘)(𝑥)|.

**Т е о р е м а 8.1**. *Пусть функция f дважды непрерывно дифференцируема на отрезке [a, b]. Тогда для составных квадратурных формул прямоугольников и трапеций справедливы следующие оценки погрешности:*

|*I* − 𝐼ℎ | ≤ 𝑀2 (𝑏−𝑎) ℎ2, (8.11)

пр 24

|*I* − 𝐼ℎ | ≤ 𝑀2 (𝑏−𝑎) ℎ2. (8.12)

тр 12

□ Выведем сначала оценку (8.11). Представим погрешность

*R = I* − 𝐼ℎ формулы прямоугольников в виде

пр

*R =* ∫𝑏 𝑓(𝑥)𝑑𝑥 − *h* ∑𝑛

𝑓𝑖−1/2 *=* ∑𝑛

∫𝑥𝑖

(𝑓(𝑥) − 𝑓(𝑥𝑖−1/2)d*x.*

𝑎 𝑖=1

𝑖=1

𝑥𝑖−1

Используя формулу Тейлора

*f(x) = f* (𝑥

) + 𝑓ʹ(𝑥

)(*x* − 𝑥 ) + 𝑓ʹʹ(𝜉) (*x* − 𝑥

)2,

𝑖−1/2

𝑖−1/2

𝑖−1/2 2

𝑖−1/2

где *x ϵ* [𝑥𝑖−1, 𝑥𝑖], ξ = ξ(*x*) ϵ [𝑥𝑖−1, 𝑥𝑖], имеем

𝑅𝑖 = ∫𝑥𝑖 (𝑓(𝑥) − 𝑓(𝑥𝑖−1/2) 𝑑𝑥 = 1 ∫𝑥𝑖

𝑓ʹʹ(ξ(*x*))( *x* − 𝑥𝑖−1/2)2d*x,*

𝑥𝑖−1 2 𝑥𝑖−1

|𝑅𝑖| ≤ 𝑀2 ∫𝑥𝑖 ( 𝑥 − 𝑥𝑖−1/2)2d*x =* 𝑀2 ℎ3*.*

2 𝑥𝑖−1 24

Так как *R* = ∑𝑛

𝑅𝑖, то |R| ≤ ∑𝑛 𝑀2 ℎ3 = 𝑀2

ℎ3𝑛. Замечая, что

𝑖=1

𝑖=1 24 24

*nh = b – a,* приходим к оценке (8.11).

Для вывода оценки (8.12) воспользуемся тем, что отрезок, соединяющий точки 𝑁𝑖−1 и 𝑁𝑖 , представляет собой график интерполяционного многочлена первой степени

*y =* 𝑃1(*x*) *=* 𝑓𝑖−1 𝑥𝑖−𝑥 *+* 𝑓𝑖 𝑥−𝑥𝑖−1*.*

ℎ ℎ

Поэтому для элементарной формулы трапеций верно равенство:

𝑅𝑖 = ∫𝑥𝑖 𝑓(𝑥) 𝑑𝑥 − ℎ (𝑓𝑖−1 + 𝑓𝑖) = ∫𝑥𝑖

(𝑓(𝑥) − 𝑃1(𝑥)) 𝑑𝑥

𝑥𝑖−1 2 𝑥𝑖−1

Используя оценку погрешности линейной интерполяции, имеем:

**|** 𝑅𝑖| ≤ ∫𝑥𝑖 𝑀2 (𝑥 − 𝑥𝑖−1)(𝑥𝑖 − 𝑥)𝑑𝑥 = 𝑀2

ℎ3.

𝑥𝑖−1 2 12

Следовательно, для *R = I* − 𝐼ℎ справедлива оценка

тр

|R| ≤ ∑𝑛 |𝑅𝑖| ≤ 𝑀2 ℎ3 = 𝑀2 (𝑏−𝑎) ℎ2. ■

𝑖=1 12 12

Приведем теперь без доказательства теорему об оценке погрешности формулы Симпсона.

**Т е о р е м а 8.2**. *Пусть функция f имеет на отрезке непрерывную производную четвертого порядка* 𝑓(4) . *Тогда для формулы Симпсона* (8.10) *справедлива оценка погрешности*

|*I* − 𝐼ℎ| ≤ 𝑀4 (𝑏−𝑎) ℎ4. (8.13)

С 2880

З а м е ч а н и е. Оценки (8.11), (8.12) и (8.13) означают, что формулы прямоугольников и трапеций имеют второй порядок точности относительно *h*, а формула Симпсона – четвертый порядок точности. Из тех же оценок следует, что формулы прямоугольников и трапеций точны для многочленов первой степени, а формула Симпсона – для многочленов третьей степени.

# § 8.2. Апостериорные оценки погрешности

Применение неравенств (8.11), (8.12) и (8.13) для априорной оценки погрешности квадратурных формул в большинстве случаев оказывается неэффективным или вообще невозможным. Это связано как с трудностями оценивания производных подынтегральной функции *f*, так и с тем, что получаемые оценки бывают сильно завышенными. Поэтому на практике обычно используются иные подходы к оценке

погрешности, позволяющие строить процедуры численного интегрирования с автоматическим выбором шага.

# Главный член погрешности.

Пусть 𝐼ℎ – приближенное значение интеграла *I* = ∫𝑏 𝑓(𝑥)𝑑𝑥, вычисленное по некоторой квадратурной формуле и использующее разбиение отрезка [*a, b*] на элементарные отрезки длины *h*. Предположим, что для погрешности этой формулы справедливо представление

𝑎

*I -* 𝐼ℎ*= C*ℎ𝑘 *+ o* (ℎ𝑘), (8.14)

где *C ≠ 0* и *k* > 0 – величины, не зависящие от *h*. Тогда величина *C*ℎ𝑘 называется *главным членом погрешности* квадратурной формулы, а число *k* является порядком точности квадратурной формулы.

В силу предположения (8.14) для погрешности квадратурной формулы при достаточно малом справедливо приближенное равенство

*I* – 𝐼ℎ *≈ C*ℎ𝑘. (8.15)

Несмотря на элементарный характер формулы (8.15), она позволяет сделать ряд важных выводов. Первый из них состоит в том, что уменьшение шага *h* в *M* раз приводит к уменьшению погрешности квадратурной формулы примерно в 𝑀𝑘 раз. Действительно, при ℎ1 = ℎ/𝑀 имеем

𝐼 − 𝐼ℎ1 ≈ *C* ℎ𝑘 = 1

*C* ℎ𝑘 ≈ 1

(*I* – 𝐼ℎ).

1 𝑀𝑘

𝑀𝑘

В частности, уменьшение шага *h* в два раза приводит к уменьшению погрешности примерно в 2𝑘 раз:

𝐼 − 𝐼ℎ/2 ≈ 1

𝑘

2

*C* ℎ𝑘 ≈ 1

2

𝑘

(*I* – 𝐼ℎ). (8.16)

1. **Правило Рунге практической оценки погрешности.** Вычитая из равенства (8.15) равенство (8.16), получим

𝐼ℎ/2 – 𝐼ℎ ≈ 1

𝑘

2

*C* ℎ𝑘 (2𝑘 – 1).

Учитывая приближенное равенство (8.16), приходим к следующей приближенной формуле:

𝐼 − 𝐼ℎ/2 ≈ 𝐼ℎ/2 – 𝐼ℎ. (8.17)

𝑘

2 – 1

Использование этой формулы для приближенной оценки погрешности значения 𝐼ℎ/2 принято называть *правилом Рунге* (или *правилом двойного пересчета*).

З а м е ч а н и е. Заменой *h* на 2*h* формула (8.17) приводится к следующему виду:

*I* – 𝐼ℎ *≈* 𝐼ℎ – 𝐼2ℎ*.* (8.18)

𝑘

2 – 1

Для формул прямоугольников и трапеций *k* = 2, а для формулы Симпсона *k* = 4. Поэтому для этих квадратурных формул равенство (8.18) принимает следующий вид:

*I* – 𝐼ℎ *≈* 1 (𝐼ℎ – 𝐼2ℎ), (8.19)

пр 3 пр пр

*I* – 𝐼ℎ *≈* 1 (𝐼ℎ – 𝐼2ℎ), (8.20)

тр 3 тр тр

*I* – 𝐼ℎ *≈* 1 (𝐼ℎ – 𝐼2ℎ). (8.21)

С 15 С С

Наличие правила Рунге получения апостериорной оценки погрешности позволяет строить процедуры вычисления интеграла *I* с заданной точностью ε, достигаемой последовательным дроблением шага интегрирования. Простейшая процедура такого типа состоит в последовательном вычислении значений 𝐼ℎ𝑖 и соответствующих апостериорных оценок погрешности 𝜀𝑖 для ℎ𝑖 = ℎ0/2𝑖, где ℎ0 – начальное значение шага, *i = 1, 2, 3, ….* Вычисления прекращаются тогда, когда при некотором *i* оказывается |𝜀𝑖| < ε (требуемая точность

достигнута) либо тогда, когда величина |𝜀𝑖| начинает возрастать (точность не может быть достигнута из-за влияния вычислительной погрешности).