В. Пикулев.

Методы компьютерных вычислений и их приложение к физическим задачам.

Методическое пособие (черновой вариант)

Петрозаводск, 2004

---

1. Всякий специалист стремится достичь своего уровня некомпетентности.
2. Компьютер – средство многократного увеличения некомпетентности человека.
3. Если две ошибки не принесли результата – испробуй третью.

Лоуренс Питер, «Иерархиология».

1. Введение в предмет

Численные методы – раздел математики, который со времен Ньютона и Эйлера до настоящего времени находит очень широкое применение в прикладной науке. Традиционно физика является основным источником задач построения математических моделей, описывающих явления окружающего мира, она же является основным потребителем алгоритмов и программ, позволяющих эти задачи с определенным успехом решать.

При этом задачей физика является не только правильный выбор программы, которая призвана решать физическую проблему, но и подробный анализ и корректировка используемых алгоритмов, в соответствии с реалиями поставленной задачи и теми математическими правилами, которые либо допускают существование решения с заданной точностью, либо говорят о невозможности такого решения.

Примеры современных физических задач, для решения которых используются численные методы – моделирование астрономических событий (рождение и развитие Вселенной), моделирование процессов в микромире (распад и синтез частиц), моделирование установок и процессов термоядерного синтеза. Более «прикладные» задачи – моделирование физических процессов в твердотельных структурах (широко используется в проектировании и изготовлении интегральных схем), моделирование процессов в газах и плазме. Учитывая большую сложность и дороговизну современных экспериментальных методик, и, с другой стороны, постоянный рост производительности вычислительных систем, нетрудно определить тенденцию к увеличению в настоящее время доли *модельных (вычислительных) экспериментов*. Большое количество численных методов разработано для решения задач математической физики, к которым, например, относятся задачи тепло- и массопереноса, исследования турбулентного движения.

К инженерным приложениям численных методов можно отнести расчеты магнитных и электростатических линз для заряженных частиц, различного рода радиотехнические расчеты, включая, например, проектирование СВЧ-волноводов. Любопытно, что как в теоретической физике, так и в инженерной практике решаются численными методами различные задачи теоретической механики, например, задачи столкновения (в том числе динамический хаос). Естественно, такое приложение вычислительной техники к физике, как управление экспериментом и сбор данных, в данном курсе не рассматривается.

План построения вычислительного эксперимента:

1) Создание модели, фиксирующей главные исследуемые факторы. Одновременно формулируются рамки применимости модели.
2) Предварительное исследование математической модели: поверка корректности постановки задачи, существования и единственности решения.
3) Разработка метода расчета сформулированной задачи, построение эффективных вычислительных алгоритмов.
4) Создание программы, осуществляющей моделирование физического объекта, включающей в себя реализации используемых численных методов, проверки корректности ввода исходных и вывода результирующих данных

5) Сравнение полученных результатов моделирования с тестовыми примерами и экспериментальными данными; решение вопроса о правильности практического моделирования (иначе повторяются пункты 3 и 4).
6) Решение вопроса о достоверности предложенной математической модели. Если модель не описывает экспериментальные данные, возврат на пункт 1.

Таким образом, численный эксперимент – это не однократное вычисление по некоторому набору формул, а многостадийный процесс программирования, анализа результатов и их погрешностей.

Задача $y = A(x)$ называется **корректно поставленной**, если для любых входных данных x из некоторого класса решение y **существует**, **единственно** и **устойчиво** по входным данным.

(Класс может представлять собой координатное бесконечномерное пространство, множество непрерывных функций и др.)

Численный алгоритм – однозначная последовательность действий, которые могут привести к одному решению.

Отсутствие устойчивости обычно означает, что сравнительно небольшой погрешности δx соответствует весьма большое δy, а значит получаемое решение будет далеко от истинного. К такой задаче численные методы применять бессмысленно, ибо погрешности численного расчета будут катастрофически нарастать. Устойчивость задачи определяется (1) математической формулировкой, (2) используемым алгоритмом расчета. Пример неустойчивой задачи в первом случае:

Система $\begin{cases} 300x_1 + 400x_2 = 700 \\ 100x_1 + 133x_2 = 233 \end{cases}$ имеет решение $\begin{cases} x_1 = 1 \\ x_2 = 1 \end{cases}$, однако

Система $\begin{cases} 300x_1 + 400x_2 = 700 \\ 100x_1 + 132x_2 = 233 \end{cases}$ имеет решение $\begin{cases} x_1 = -3 \\ x_2 = 4 \end{cases}$, то есть разница в коэффициенте менее 1% приводит к изменению решения в 300%.

Пример алгоритмической неустойчивости – вычисление производных численными методами: какой бы метод мы не использовали, приходится вычитать весьма мало различающиеся числа.

В настоящее время развиты методы решения многих некорректных задач, которые основаны на решении вспомогательной корректной задачи, близкой к исходной.

Если выполняется условие для норм (модулей) $\|\partial y\| = C\|\partial x\|$, то задача устойчива. Однако если константа *C* очень велика, то фактически наблюдается *слабая устойчивость*. Такую задачу называют *плохо обусловленной*. Пример – дифференциальное уравнение $y''(x) = y(x)$ с начальными условиями $y(0) = 1, y'(0) = -1$. Общее решение дифференциального уравнения есть $y(x) = \frac{1}{2}\left(y(0) + y'(0)\right)e^{x} + \frac{1}{2}\left(y(0) - y'(0)\right)e^{-x}$. Начальные условия приводят к обнулению первого слагаемого, но если из-за погрешности начальных данных это будет не так, то при возрастании *x* влияние первого слагаемого будет катастрофически нарастать.

2. Точность вычислений, классификация погрешностей.

Во всех случаях математическая точность решения должна быть в 2-4 раза выше, чем ожидаемая физическая точность модели. Более высокая математическая точность, как и более низкая, будут неадекватны данной модели.

Существуют четыре источника погрешности результата:

1) ***погрешность математической модели*** – связана с ее несоответствием физической реальности, так как абсолютная истина недостижима. Если математическая модель выбрана недостаточно тщательно, то, какие бы методы мы не применяли для расчета, все результаты будут недостаточно надежны, а в некоторых случаях и совершенно неправильны.

2) ***погрешность исходных данных***, принятых для расчета. Это ***неустранимая погрешность***, но это погрешность возможно и необходимо оценить для выбора алгоритма расчета и точности вычислений. Как известно, ошибки эксперимента условно делят на систематические, случайные и грубые, а идентификация таких ошибок возможно при статистическом анализа результатов эксперимента.

3) ***погрешность метода*** – основана на дискретном характере любого численного алгоритма. Это значит, что вместо точного решения исходной задачи метод находит решение другой задачи, близкого в каком-то смысле (например по норме банахова пространства) к искомому. Погрешность метода – основная характеристика любого численного алгоритма. Погрешность метода должна быть в 2-5 раз меньше неустранимой погрешности.

4) ***погрешность округления*** – связана с использованием в вычислительных машинах чисел с конечной точностью представления.

Вот иллюстрация этих определений. Пусть имеется реальный маятник, совершающий затухающие колебания, начинающий движение в момент $t = t_0$. Требуется найти угол отклонения φ от вертикали в момент t_1. Движение маятника мы можем описать следующим дифференциальным уравнением:

$$l\frac{d^2\varphi}{dt^2} + g\sin\varphi + \mu\frac{d\varphi}{dt} = 0,$$

где l – длина маятника, g – ускорение силы тяжести, μ – коэффициент трения.

Как только принимается такое описание задачи, решение уже приобретает неустранимую погрешность, в частности потому, что реальное трение зависит от скорости не совсем линейно (погрешность модели). Кроме того, воспроизведя реальный эксперимент, мы зададим l, g (в известной точке планеты), μ с некоторой точностью, и получим набор значений с погрешностью, которую можем оценить из анализа статистики некоторого числа однотипных опытов (погрешность исходных данных). Взятое в модели дифференциальное уравнение нельзя решить в явном виде, для его решения требуется применить какой-либо численный метод, имеющий заранее известную погрешность, которая должна быть меньше неустранимой погрешности. После совершения вычислений мы получим значения с погрешностью большей, нежели погрешность метода, так как к ней прибавится погрешность округления.

Рассмотрим правила расчета погрешности округления:

1) **Сложение и вычитание приближенных чисел**

Введем в рассмотрение два числа a и b, называемых *приближенными*, то есть это есть оценка точных значений A и B, известных с *абсолютными погрешностями* $\pm\varepsilon_a$ и $\pm\varepsilon_b$. Знаки этих погрешностей нам неизвестны, следовательно для обеспечения достоверности конечного результата мы должны взять наихудший случай, когда погрешности складываются. Таким образом формулируются следующие правила:

1. Абсолютная погрешность **суммы** приближенных чисел равна **сумме** абсолютных погрешностей слагаемых.

2. Абсолютная погрешность **разности** приближенных чисел равна **сумме** абсолютных погрешностей слагаемых.

Относительной погрешностью приближенного числа *a* будет являться величина $\delta_a \le \left|\frac{\varepsilon_a}{a}\right|$. По этому же правилу определим относительную погрешность **суммы** приближенных чисел *a* и *b* как $\partial \le \left|\frac{\varepsilon_a + \varepsilon_b}{a+b}\right|$. При этом можно показать, что

3. Относительная погрешность суммы слагаемых одного знака заключена между наименьшей и наибольшей относительными погрешностями слагаемых: $\partial_{\min} \le \delta \le \delta_{\max}$.
4. Для разности двух приближенных чисел одного знака величина относительной погрешности $\partial \le \left|\frac{\varepsilon_a + \varepsilon_b}{a-b}\right|$ может быть сколь угодно большой.

2) **Умножение и деление приближенных чисел**

Очевидно, что приближенное число $a = A \pm \varepsilon_a = A\left(1 \pm \frac{\varepsilon}{A}\right) = A(1 \pm \partial_a)$. Тогда для произведения $c = C(1 \pm \partial_{ab}) = ab = AB(1 \pm \partial_a) \cdot (1 \pm \partial_b) = AB(1 \pm (\partial_a + \partial_b) + \partial_a \partial_b)$. Если пренебречь последним малым слагаемым в скобках, то можно сформулировать следующее правило:

1. Относительная погрешность **произведения** приближенных чисел равна **сумме** относительных погрешностей множителей $\partial_{ab} = \partial_a + \partial_b$.

Так как деление на число b равнозначно умножению на 1/b, то справедливо утверждение:

2. Относительная погрешность **частного** приближенных чисел равна **сумме** относительных погрешностей делимого и делителя.

Следовательно, при умножении и делении приближенных чисел необходимо принимать во внимание количество значащих цифр, характеризующих относительную точность числа, а не количество десятичных знаков, обуславливающих его абсолютную погрешность.

Совершенно очевидно, что при большом количестве действий такого сорта правила нельзя считать удовлетворительными, так как погрешности будут иметь разные знаки и компенсировать друг друга. *Статистическая оценка* показывает, что при *N* одинаковых действиях среднее значение суммарной ошибки больше единичной в $\sqrt{N}$ раз, если нет систематических причин для накопления погрешности. Систематические причины возникают, если, например в алгоритме вычитаются близкие по величине числа.

При любых расчетах надо устанавливать такую точность вычислений, чтобы погрешность округления была существенно меньше всех остальных погрешностей.

3. Краткое введение в используемые программные средства

Традиционные языки высокоуровневого программирования. В большинстве практических случаев моделирование или численный расчет предполагает использование готового алгоритма, который необходимым образом модифицируется для конкретной задачи. В настоящее время существуют обширные фонды алгоритмов и программ, ориентированных на классические языки программирования, такие как Фортран, Си и Паскаль. Наиболее мощные математические библиотеки были разработаны для самого старого языка программирования из вышеперечисленных – Фортрана. Однако во многих случаях, обладая соответствующей квалификацией, легче написать программу, отталкиваясь от первых принципов, нежели тестировать и исправлять, или переводить на другой язык программирования незнакомый текст.

Другое направление, которое в настоящее время особенно популярно – использование мощных математических пакетов для численных и аналитических расчетов.

Mathcad.

Разрабатывается компанией MathSoft Inc. Является наиболее легкой для освоения системой математических расчетов. Принята концепция «активного документа», то есть все вычисления записываются в традиционной математической нотации (с использованием значков интеграла, суммы и др.), а после введения знака равенства или другого запускающего символа появляется рассчитанное значение. Основной недостаток – слишком мал набор основных функций и очень низкое быстродействие.

MathLab.

Система MATLAB (MATrix LABoratory) разрабатывается фирмой MathWorks. Эта система создана для работы в среде Windows и представляет собой интерактивную среду для вычислений и моделирования, причем она может работать как в режиме непосредственных вычислений (очень напоминает режим «командной строки»), так и в режиме интерпретации написанных программ. Сильная сторона системы – виртуозная работа с матрицами и векторами. Численное значение или аналитическая формула, а также сообщения системы выводится на экран в виде списка. Помимо обычных алгебраических вычислений система имеет огромный набор встроенных функций, а также имеется возможность создавать пользовательские функции. В системе очень качественно реализовано построение двух и трехмерных изображений, в том числе динамически изменяющихся. Кроме того, имеется библиотека, которая обеспечивает удобное управление исполнением программ. И это только базовый набор, который обычно расширяется многочисленными дополнениями – например языком Simulink моделирования нелинейных динамических систем. Основное назначение – технические расчеты.

Mathematica.

Разрабатывается фирмой Wolfram Research. Одна из наиболее сложных для освоения систем. Предназначена в основном для решения теоретических задач, в связи с чем весьма популярна в научных кругах. Предуставляет широкие возможности в проведении символических (аналитических) преобразований, красотой интерфейса не отличается.

Maple.

Разработка компании Waterloo Maple Inc. Также очень популярный в научных кругах пакет аналитических преобразований и численных методов. Символический процессор Maple поставляется отдельно, в связи с чем производители других программных средств могут интегрировать его в свои разработки.

4. Численное интегрирование.

Задача численного интегрирования состоит в замене исходной подинтегральной функции *f(x)*, для которой трудно или невозможно записать первообразную в аналитике, некоторой аппроксимирующей функцией *φ(x)*. Такой функцией обычно является полином (кусочный полином) $\phi(x) = \sum_{i=1}^{n} c_i \varphi_i(x)$. То есть:

$$I = \int_a^b f(x)dx = \int_a^b \varphi(x)dx + R,$$

где $R = \int_a^b r(x)dx$ – *априорная погрешность метода* на интервале интегрирования,

а *r(x)* – априорная погрешность метода на отдельном шаге интегрирования.

Обзор методов интегрирования.

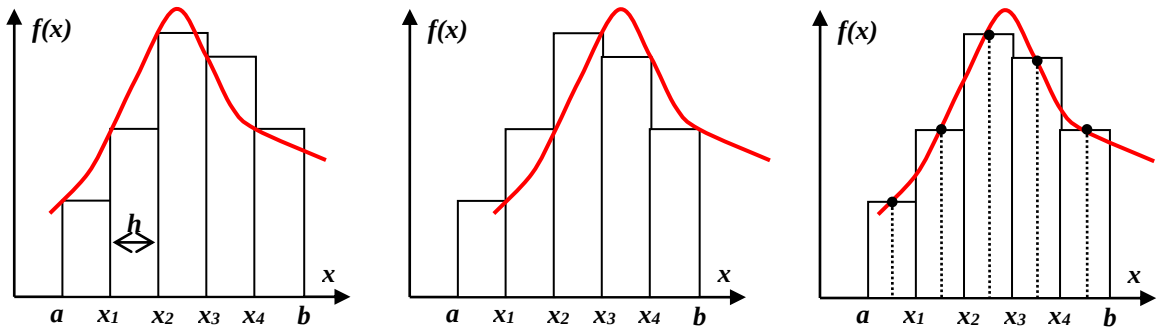
Методы вычисления однократных интегралов называются *квадратурными* (для кратных интегралов – *кубатурными*).

1) Методы Ньютона-Котеса. Здесь *φ(x)* – полином различных степеней. Сюда относятся метод прямоугольников, трапеций, Симпсона.
2) Методы статистических испытаний (методы Монте-Карло). Здесь узлы сетки для квадратурного или кубатурного интегрирования выбираются с помощью датчика случайных чисел, ответ носит вероятностный характер. В основном применяются для вычисления кратных интегралов.
3) Сплайновые методы. Здесь *φ(x)* – кусочный полином с условиями связи между отдельными полиномами посредством системы коэффициентов.
4) Методы наивысшей алгебраической точности. Обеспечивают оптимальную расстановку узлов сетки интегрирования и выбор весовых коэффициентов *ρ(x)* в задаче $\int_a^b \varphi(x)\rho(x)dx$.

   Сюда относится метод Гаусса-Кристоффеля (вычисление несобственных интегралов) и метод Маркова.

Метод прямоугольников.

Различают метод левых, правых и средних прямоугольников. Суть метода ясна из рисунка. На каждом шаге интегрирования функция аппроксимируется полиномом нулевой степени – отрезком, параллельным оси абсцисс.



Левые прямоугольники Правые прямоугольники Средние прямоугольники

Выведем формулу метода прямоугольников из анализа разложения функции *f(x)* в ряд Тейлора вблизи некоторой точки $x = x_i$.

$$f(x)\big|_{x=x_i} = f(x_i) + (x - x_i) f'(x_i) + \frac{(x - x_i)^2}{2!} f''(x_i) + \dots$$

Рассмотрим диапазон интегрирования от x_i до $x_i + h$, где h – шаг интегрирования.

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

$= f(x_i)h + \frac{h^2}{2} f'(x_i) + O(h^3) = f(x_i)h + r_i$. Получили формулу *правых (или левых) прямоугольников* и априорную оценку погрешности r на отдельном шаге интегрирования. Основной критерий, по которому судят о точности алгоритма – степень при величине шага в формуле априорной оценки погрешности.

В случае равного шага h на всем диапазоне интегрирования общая формула имеет вид

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

Здесь n – число разбиений интервала интегрирования, $R = \sum_{i=0}^{n-1} r_i = \frac{h}{2} \cdot h \sum_{i=0}^{n-1} f'(x_i) = \frac{h}{2} \int_a^b f'(x)dx$. Для справедливости существования этой оценки необходимо существование непрерывной $f'(x)$.

Метод средних прямоугольников. Здесь на каждом интервале значение функции считается в точке $\bar{x} = x_i + \frac{h}{2}$, то есть $\int_{x_i}^{x_i+h} f(x)dx = hf(\bar{x}) + r_i$. Разложение функции в ряд Тейлора показывает, что в случае средних прямоугольников точность метода существенно выше:

$$r = \frac{h^3}{24} f''(\bar{x}),\ R = \frac{h^2}{24} \int_a^b f''(x)dx.$$

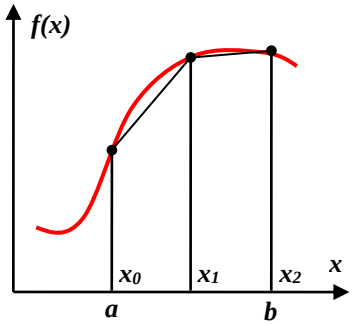
Метод трапеций.

Аппроксимация в этом методе осуществляется полиномом первой степени. Суть метода ясна из рисунка.

На единичном интервале $\int_{x_i}^{x_i+h} f(x)dx = \frac{h}{2}\left(f(x_i) + f(x_i + h)\right) + r_i$.

В случае равномерной сетки (h = const)

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

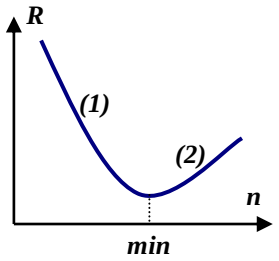


Метод трапеций

При этом $r_i = -\frac{h^3}{12} f''(x_i)$, а $R = -\frac{h^3}{12} \int_a^b f''(x)dx$. Погрешность метода трапеций в два раза выше, чем у метода средних прямоугольников! Однако на практике найти среднее значение на элементарном интервале можно только у функций, заданных аналитически (а не таблично), поэтому использовать метод средних прямоугольников удается далеко не всегда. В силу разных знаков погрешности в формулах трапеций и средних прямоугольников истинное значение интеграла обычно лежит между двумя этими оценками.

Особенности поведения погрешности.

Казалось бы, зачем анализировать разные методы интегрирования, если мы можем достичь высокой точности, просто уменьшая величину шага интегрирования. Однако рассмотрим график поведения апостериорной погрешности R результатов численного расчета в зависимости от числа n разбиений интервала (то есть при $n \to \infty$ шаг $h \to 0$). На участке (1) погрешность уменьшается в связи с уменьшением шага h. Но на участке (2) начинает доминировать вычислительная погрешность, накапливающаяся в результате многочисленных арифметических действий. Таким образом, для каждого метода существует своя R_{min}, которая зависит от многих факторов, но прежде всего от априорного значения погрешности метода R.



<u>Уточняющая формула Ромберга.</u>

Метод Ромберга заключается в последовательном уточнении значения интеграла при кратном увеличении числа разбиений. В качестве базовой может быть взята формула трапеций с равномерным шагом h.

Обозначим интеграл с числом разбиений $n = 1$ как $R(1;1) = \frac{h}{2}\left(f(a) - f(b)\right)$.

Уменьшив шаг в два раза, получим $R(2;1) = \frac{h}{2}\left(f(a) + f(b)\right) + hf(a+h)$.

Если последовательно уменьшать шаг в 2^n раз, получим рекуррентное соотношение для расчета

$$R(n+1;1) = \frac{1}{2}R(n;1) + h\sum_{i=1}^{2^n-1} f\left(a + (2i-1)h\right).$$

Пусть мы вычислили четыре раза интеграл с n от 1 до 4. Представим следующий треугольник:

R(1;1)

R(2;1) R(2;2)

R(3;1) R(3;2) R(3;3)

R(4;1) R(4;2) R(4;3) **R(4;4)**

В первом столбце стоят значения интеграла, полученные при последовательном удвоении числа интервалов. Следующие столбцы – результаты уточнения значения интеграла по следующей рекуррентной формуле:

$$R(n+1;m+1) = R(n+1;m) + \frac{R(n+1;m) - R(n;m)}{4^m - 1}.$$

Правое нижнее значение в треугольнике – искомое уточненное значение интеграла.

<u>Метод Симпсона</u>.

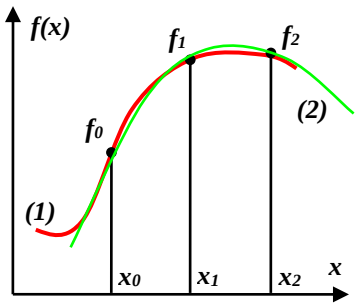
Подинтегральная функция $f(x)$ заменяется интерполяционным полиномом второй степени $P(x)$ – параболой, проходящей через три узла, например, как показано на рисунке ((1) – функция, (2) – полином).

Рассмотрим два шага интегрирования (h = const = $x_{i+1} - x_i$), то есть три узла x_0, x_1, x_2, через которые проведем параболу, воспользовавшись уравнением Ньютона:

$$P(x) = f_0 + \frac{x - x_0}{h}\left(f_1 - f_0\right) + \frac{(x - x_0)(x - x_1)}{2h^2}\left(f_0 - 2f_1 + f_2\right).$$

Пусть $z = x - x_0$,

тогда $P(z) = f_0 + \frac{z}{h}\left(f_1 - f_0\right) + \frac{z(z-h)}{2h^2}\left(f_0 - 2f_1 + f_2\right) =$



Метод Симпсона

$$= f_0 + \frac{z}{2h}\left(-3f_0 + 4f_1 - f_2\right) + \frac{z^2}{2h^2}\left(f_0 - 2f_1 + f_2\right)$$

Теперь, воспользовавшись полученным соотношением, сосчитаем интеграл по данному интервалу:

$$\int_{x_0}^{x_2} P(x)dx = \int_0^{2h} P(z)dz = 2hf_0 + \frac{(2h)^2}{4h}\left(-3f_0 + 4f_1 - f_2\right) + \frac{(2h)^3}{6h^2}\left(f_0 - 2f_1 + f_2\right) =$$

$$= 2hf_0 + h\left(-3f_0 + 4f_1 - f_2\right) + \frac{4h}{3}\left(f_0 - 2f_1 + f_2\right) = \frac{h}{3}\left(6f_0 - 9f_0 + 12f_1 - 3f_2 + 4f_0 - 8f_1 + 4f_2\right).$$

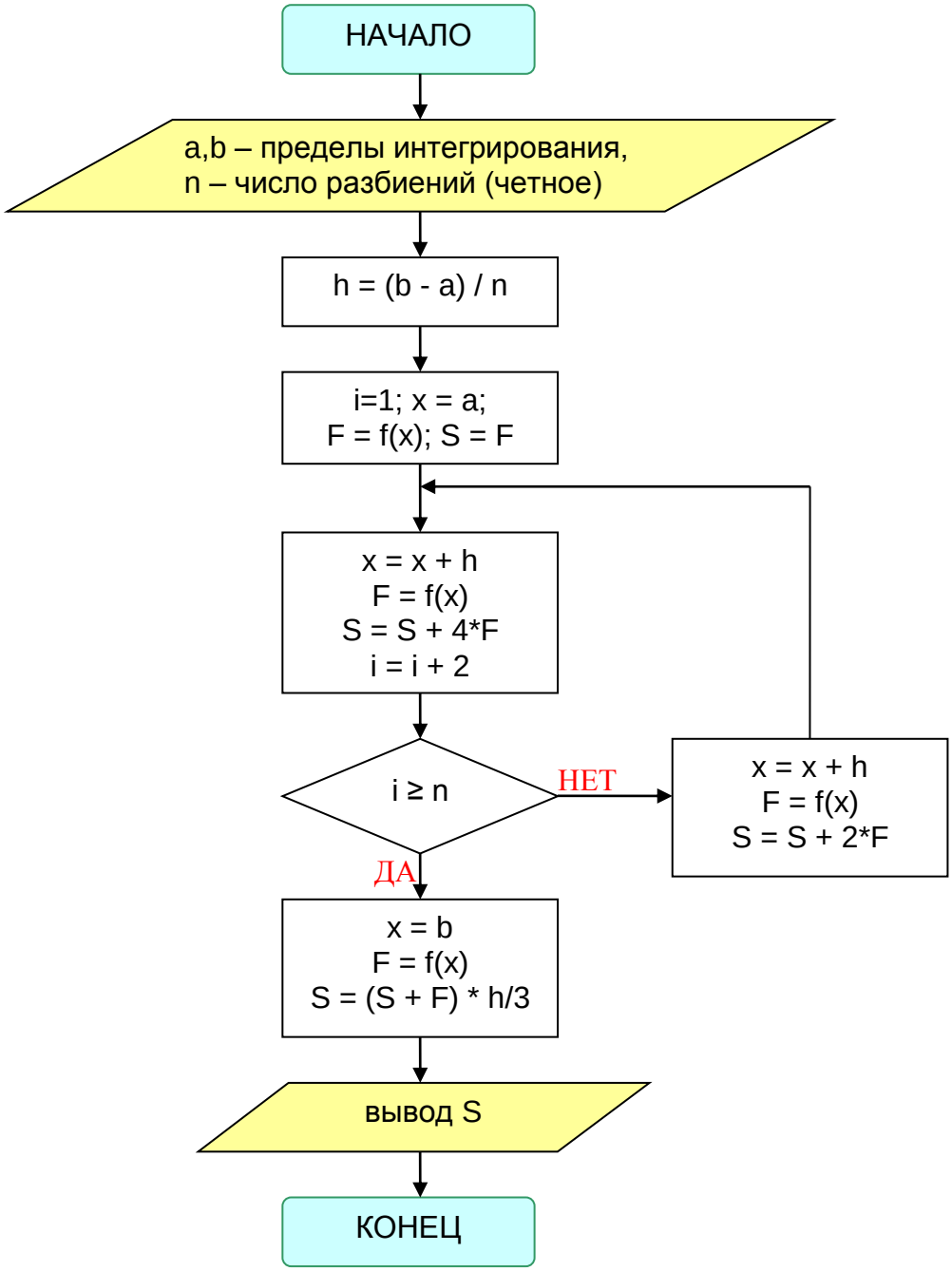
В итоге $\int_{x_0}^{x_2} f(x)dx = \frac{h}{3}\left(f_0 + 4f_1 + f_2\right) + r$.

Для *равномерной сетки* и *четного числа шагов n* формула Симпсона принимает вид:

$$\int_a^b f(x)dx = \frac{h}{3}\left(f_0 + 4f_1 + 2f_2 + 4f_3 + 2f_4 + \ldots + 2f_{n-2} + 4f_{n-1} + f_n\right) + R$$

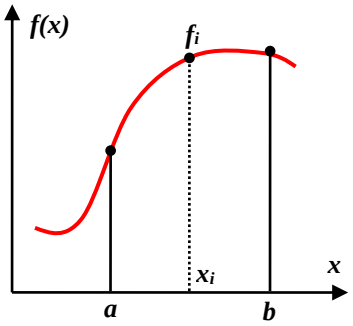
Здесь $r = -\frac{h^5}{90} f^{IV}(x_i)$, а $R = -\frac{h^4}{180}\int_a^b f^{IV}(x)dx$ в предположении непрерывности четвертой производной подинтегральной функции.

Блок-схема алгоритма метода Симпсона.



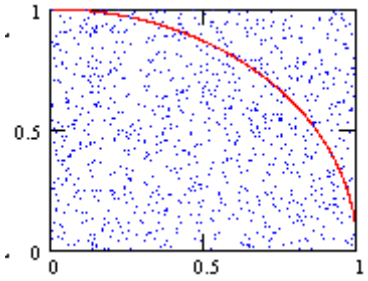
Методы Монте-Карло.

1) одномерная *случайная величина* – статистический вариант метода прямоугольников.



В качестве текущего узла x_i берется случайное число, равномерно распределенное на интервале интегрирования [*a*, *b*]. Проведя *N* вычислений, значение интеграла определим по следующей формуле: $\int_a^b f(x)dx = \frac{b-a}{N}\sum_{i=1}^{N} f(x_i) + R$. Для *R* можно утверждать хотя бы $\sim \frac{1}{\sqrt{N}}$.

2) двумерная случайная величина – оценка площадей.



Рассматриваются две *равномерно распределенных* случайных величины x_i и y_i, которые можно рассматривать как координаты точки в двумерном пространстве. За приближенное значение интеграла принимается количества точек *S*, попавших под кривую *y* = *f*(*x*), к общему числу испытаний *N*, т.е. $\int_a^b f(x)dx \approx \frac{S}{N}$.

И первый, и второй случай легко обобщаются на кратные интегралы.

5. Оценка апостериорной погрешности.

Мы записывали *априорные* оценки главного члена погрешности в виде $R_0 = Ah^p$, (1) где A – коэффициент, зависящий от метода интегрирования и вида подинтегральной функции; h – шаг интегрирования, p – порядок метода. Такого сорта оценку можно применить не только к методам интегрирования, но и ко многим другим численным алгоритмам.

Первая формула Рунге.

Пусть w – точное значение, к которому должен прийти численный метод (мы его не знаем). Результат численного расчета дает нам величину w_h такую, что $w = w_h + Ah^p + O\left(h^{p+1}\right)$. (2)

Теперь вычислим ту же величину w с шагом kh, где константа k может быть как больше, так и меньше единицы. Коэффициент A будет одинаковый, так как вычисление осуществляется одним и тем же методом. Получаем $w = w_{kh} + A(kh)^p + O\left((kh)^{p+1}\right)$. (3)

Приравняем правые части выражений (2) и (3) и пренебрежем бесконечно малыми величинами одинакового порядка малости.

$w_h + Ah^p = w_{kh} + k^p Ah^p$. Отсюда, учитывая (1), получим $R_0 = \dfrac{w_h - w_{kh}}{k^p - 1}$. (4) Эта формула, выражающая *апостериорную* оценку главного члена погрешности величины w путем двойного просчета с разным шагом, носит название первой формулы Рунге. При уменьшении шага главный член погрешности будет стремиться к полной погрешности R.

Вторая формула Рунге.

Так как модуль и знак апостериорной погрешности из формулы (4) известны, можно уточнить искомое значение $w_{corr} = w_h + R_0$. Это вторая формула Рунге. Однако теперь погрешность w_{corr} не определена, известно лишь, что она по модулю меньше R_0.

Алгоритм Эйткена.

Способ оценки погрешности для случая, когда порядок метода p неизвестен. Более того, алгоритм позволяет опытным путем определить и порядок метода. Для этого в третий раз вычислим значение величины w с шагом k^2h:

$w = w_{k^2h} + A(k^2h)^p + O\left((k^2h)^{p+1}\right)$. (5)

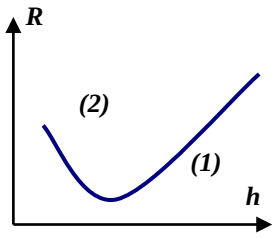
Приравняем правые части выражений (5) и (3): $w_{kh} + k^p Ah^p = w_{k^2h} + k^{2p} Ah^p$. Отсюда:

$w_{kh} - w_{k^2h} = R_0 k^p \left(k^p - 1\right)$. Подставим сюда значение R_0 из (4):

$k^p = \dfrac{w_{kh} - w_{k^2h}}{w_h - w_{kh}}$. Из этой формулы определяем знаменатель для (4). Кроме того, определяем порядок $p = \ln\left(\dfrac{w_{kh} - w_{k^2h}}{w_h - w_{kh}}\right) / \ln k$. Для правильно реализованных алгоритмов методов априорных и апостериорных порядки должны получиться совпадающими. Программная реализация формул Рунге позволяет вычислить определенные интегралы с заданной точностью, когда выбор необходимого числа разбиений интервала интегрирования осуществляется автоматически. Пример – уже рассмотренная ранее формула Ромберга.

6. Численное дифференцирование.

Методы численного дифференцирования применяются, если исходную функцию *f(x)* трудно или невозможно продифференцировать аналитически. Например, эта функция может быть задана таблично. Задача численного дифференцирования – выбрать легко вычисляемую функцию (обычно полином) $\varphi(x,\vec{a})$, для которой приближенно полагают $y'(x)=\varphi'(x,\vec{a})$.



Численное дифференцирование – некорректная задача, так как отсутствует устойчивость решения. При численном дифференцировании приходится вычитать друг из друга близкие значения функции. Это приводит к уничтожению первых значащих цифр, т.е. к потере части достоверных знаков числа. А так как значения функции обычно известны с определенной погрешностью, то все значащие цифры могут быть потеряны. На графике кривая (1) соответствует уменьшению погрешности дифференцирования при уменьшении шага; кривая (2) представляет собой неограниченно возрастающий (осциллирующий) вклад *неустранимой погрешности* исходных данных – значений функции *y*(*x*). Критерий выхода за оптимальный шаг при его уменьшении – «разболтка» решения: зависимость результатов вычислений становится нерегулярно зависящей от величины шага.

Пусть $\varphi(x,\vec{a})$ введена как *интерполяционный многочлен Ньютона*. В этом случае для произвольной неравномерной сетки:

$y'(x_i)\approx\dfrac{y_{i+1}-y_i}{x_{i+1}-x_i}$, для $i=0,1\ldots n$-1, интерполяция полиномом первой степени.

$y''(x_i)\approx\dfrac{2}{x_{i+2}-x_i}\left(\dfrac{y_{i+1}-y_i}{x_{i+1}-x_i}-\dfrac{y_{i+2}-y_i}{x_{i+2}-x_i}\right)$, интерполяция полиномом второй степени.

В общем случае $y^{(k)}(x)\approx k!\sum\limits_{p=0}^{k}y_p\prod\limits_{\substack{i=0\\ i\neq p}}^{k}\left(x_p-x_i\right)^{-1}$. Минимальное число узлов, необходимое для вычисления *k*-й производной, равно *k* + 1.

Оценка погрешности при численном дифференцировании может быть осуществлена по формуле $R_n^{(k)}\le\dfrac{\max\left|y^{(n+1)}\right|}{(n+1-k)!}\max\limits_i\left|x-x_i\right|^{n+1-k}$, где *n* – число узлов функции, *k* – порядок производной.

На практике чаще всего используются упрощенные формулы для равномерной сетки, при этом точность нередко повышается. Часто используются следующие формулы для трех узлов:

$y'(x_{i+1})\approx\dfrac{y_{i+2}-y_i}{2h}$, где $h = x_1 - x_0 = \text{const}$.

$y''(x_{i+1})\approx\dfrac{y_{i+2}-2y_{i+1}+y_i}{h^2}$.

Исходя из общего вида интерполяционного полинома можно вывести формулы для более высокого порядка точности или для более высоких производных.

7. Численное решение систем линейных уравнений.

Классы задач линейной алгебры

При численном решении большого круга задач в конечном итоге происходит их *линеаризация*, в связи с чем в соответствующих алгоритмах весьма широко используются методы линейной алгебры. В их числе:

- решение систем линейных алгебраических уравнений (СЛАУ);
- вычисление определителей матриц $\det\left|\hat{A}\right|$;
- нахождение обратных матриц $\hat{A}^{-1}$;
- определение собственных значений и собственных векторов матриц $\hat{A}\vec{x} = \lambda\vec{x}$;

Постановка задачи решения СЛАУ: $\hat{A}\vec{x} = \vec{b}$, (1) где $\hat{A}$ – квадратная матрица коэффициентов размерности *n*, $\vec{x}$ – вектор неизвестных, $\vec{b}$ – вектор свободных коэффициентов. Иногда СЛАУ представляют в виде расширенной матрицы $\hat{A}$ размерности $n \times n+1$, где в качестве последнего столбца фигурирует вектор свободных коэффициентов. В координатном представлении такая СЛАУ выглядит следующим образом:

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

Для решения СЛАУ применяют в основном два класса методов: *прямые* (выполняемые за заранее известное количество действий) и *итерационные* (обеспечивающие постепенную сходимость к корню уравнения, зависящую от многих факторов). Прямые методы обычно применяются для решения систем порядка $n < 200$, для бóльших *n* используются итерационные методы. Перед решением СЛАУ требуется проанализировать *корректную постановку задачи*:

1) Если $\det\left|\hat{A}\right| \neq 0$ – решение существует и единственно. Если же определитель равен нулю, то тогда, если матрица вырождена (т.е. ее можно преобразовать к виду, когда как минимум одна строка коэффициентов – нули) решений бесконечное множество, иначе решения не существует.
2) Если $\hat{A}^{-1}$ не имеет элементов с большими по модулю значениями – решение устойчиво (см. пример к главе 1). Показателем плохо обусловленных систем является $\det\left|\hat{A}\right| \approx 0$.

Алгоритм метода Гаусса

1) *Прямой ход*.

Идея метода состоит в последовательном исключении неизвестных из системы *n* линейных уравнений. На примере первого уравнения системы (2) рассмотрим выражение для x_1:

$$x_1 = \frac{a_{1,n+1}}{a_{11}} - \frac{a_{12}}{a_{11}}x_2 - \ldots\frac{a_{1n}}{a_{11}}x_n.$$

Подставим выражение для x_1 во *второе* и все остальные уравнения системы:

$$\left(a_{22} - \frac{a_{21}a_{12}}{a_{11}}\right)x_2 + \ldots + \left(a_{2n} - \frac{a_{21}a_{1n}}{a_{11}}\right)x_n = a_{2,n+1} - \frac{a_{21}a_{1,n+1}}{a_{11}}.$$

Для расширенной матрицы коэффициентов это означает, что каждый элемент первой строки следует поделить на диагональный элемент, а все остальные строки преобразовать, как показано выше. Таким образом, станут равны нулю все коэффициенты первого столбца, лежащие ниже главной диагонали. Затем аналогичная процедура проводится со второй строкой матрицы и

нижележащими строками, при этом первая строка и первый столбец уже не изменяются. И так далее до тех пор, пока все коэффициенты, лежащие ниже главной диагонали, не будут равны нулю.
Общие формулы прямого хода:

$$a^*_{kj} = \frac{a_{kj}}{a_{kk}}, \quad (3)$$

$k = 1\ldots n$, $j = 1\ldots n+1$. Звездочкой отмечены элементы k-й строки с измененными значениями, которые будут подставлены в следующую формулу. Для определенности будем считать первый индекс – по строкам, второй – по столбцам.

$$a_{ij} = a_{ij} - a_{ik} a^*_{kj}, \quad (4)$$

$i = k+1\ldots n$, $j = 1\ldots n+1$, k фиксировано в уравнении (3). Для уменьшения количества действий достаточно изменять значения элементов, находящихся выше главной диагонали.

2) *Обратный ход.*

Второй этап решения СЛАУ методом Гаусса называется обратным ходом и состоит в последовательном определении x_k, начиная с x_n, так как для последнего решение фактически получено. Общая формула:

$$x_k = a_{k,n+1} - \sum_{j=k+1}^{n} a_{kj} x_j \ . \quad (5)$$

Таким образом, вычисление корней $\vec{x}$ происходит за $2/3\ n^3$ арифметических действий.

3) *Выбор главного элемента.*

Для уменьшения погрешности вычислений следует стремиться к тому, чтобы на главной диагонали матрицы стояли максимальные по модулю значения коэффициентов. Алгоритмически этого можно добиться, переставляя строки таким образом, чтобы на диагонали стоял наибольший по модулю элемент текущего столбца. Такая процедура называется *выбором главного элемента* и осуществляется всякий раз при переходе к новой строке в прямом цикле метода Гаусса.

4) *Погрешность метода. Расчет невязок.*

Точность результатов будет определяться только точностью выполнения арифметических операций при преобразовании элементов матрицы, т.е. ошибкой округления. Контроль правильности полученного решения осуществляется подстановкой полученных значений $x_1\ldots x_n$ в исходную систему уравнений и вычислением *невязок*, т.е. разностей между правыми и левыми частями уравнений:

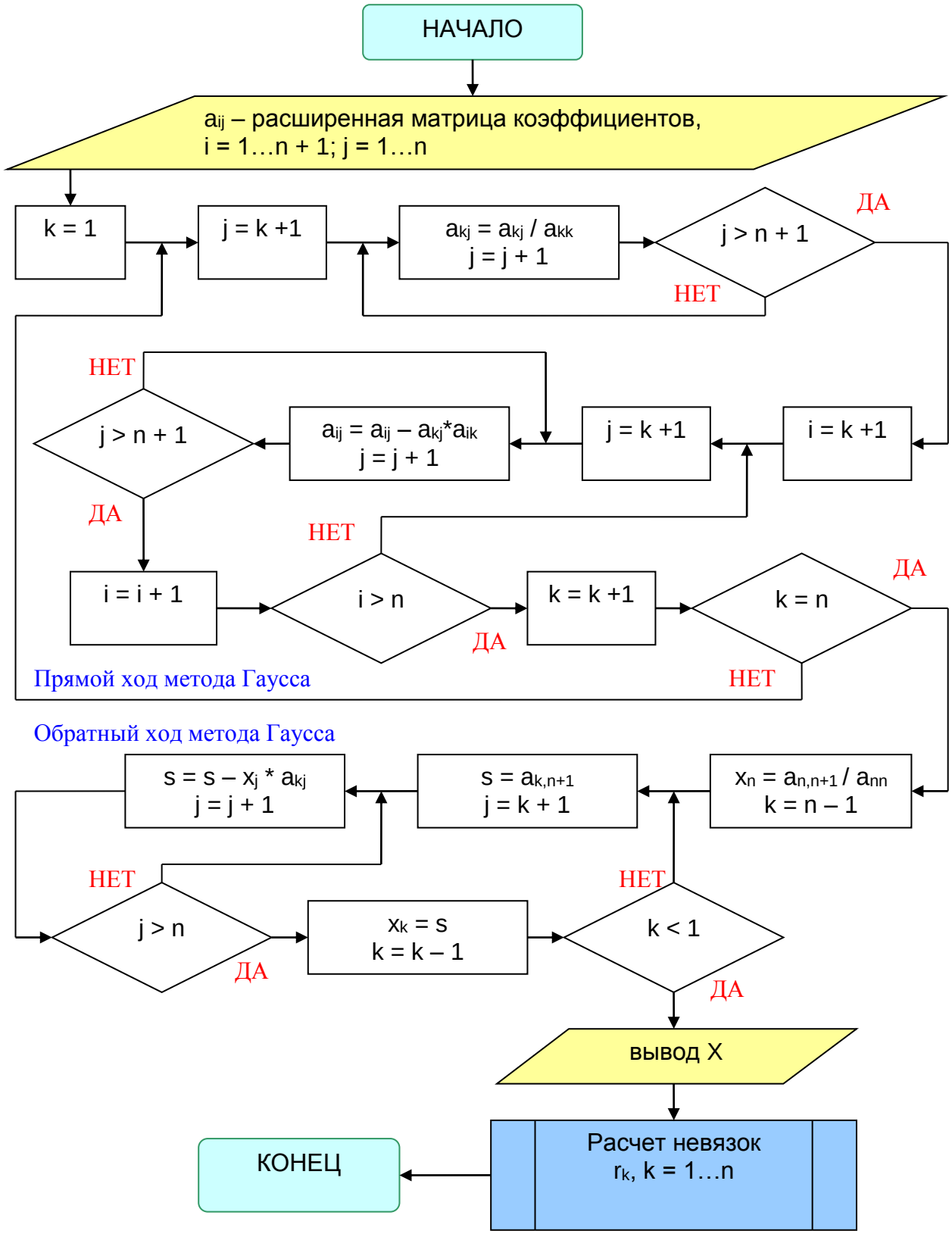
$$r_k = a_{k,n+1} - \sum_{j=1}^{n} a_{kj} x_j, \text{ где } k = 1\ldots n. \quad (6)$$

Специально отметим, что подставлять найденные значения $\vec{x}$ следует в *исходную* (не преобразованную к верхнетреугольному виду) систему.

5) *Преимущества и недостатки метода.*

Преимущество метода в том, что он позволяет достичь результата за заранее известное и фиксированное число действий. Точность результатов будет определяться правильным выбором порядка коэффициентов в матрице $\hat{A}$ и ее размерностью. Недостатком метода является резкое увеличение времени и погрешности вычислений с ростом n.

Блок-схема алгоритма метода Гаусса без выбора главного элемента.



Итерационные методы решения систем линейных уравнений.

Простейшим итерационным методом решения СЛАУ является *метод простой итерации*. При этом система уравнений $\hat{A}\vec{x}=\vec{b}$ (1) преобразуется к виду $\vec{x}=\hat{B}\vec{x}+\vec{c}$ (2), а ее решение находится как предел последовательности $\vec{x}^{\{n+1\}}=\hat{B}\vec{x}^{\{n\}}+\vec{c}$ (3), где *{n}* – номер итерации. Утверждается, что всякая система (2), эквивалентная (1), записывается в виде $\vec{x}=\vec{x}-(\hat{E}-\hat{B})\hat{A}^{-1}(\hat{A}\vec{x}-\vec{b})$.

Теорема о достаточном условии сходимости метода простой итерации утверждает, что если норма матрицы $\left\|\hat{B}\right\|<1$ ($\left\|\hat{B}\right\|=\max\limits_{i}\sum\limits_{j=1}^{n}\left|b_{ij}\right|$), то система уравнений (2) имеет единственное решение

и итерационный процесс (3) сходится к решению со скоростью геометрической прогрессии.

Теорема о необходимом и достаточном условии сходимости метода простой итерации: Пусть система (2) имеет единственное решение. Итерационный процесс (3) сходится к решению системы (2) при любом начальном приближении тогда и только тогда, когда все *собственные значения* матрицы $\hat{B}$ по модулю меньше 1.

На практике для обеспечения сходимости итерационных методов необходимо, чтобы значения диагональных элементов матрицы СЛАУ были преобладающими по абсолютной величине по сравнению с другими элементами.

Представим СЛАУ в следующей форме, удовлетворяющей (3):

$$\begin{cases} x_1=\left(a_{1,n+1}-a_{11}x_1-a_{12}x_2-\ldots-a_{1n}x_n\right)/a_{11}+x_1 \\ x_2=\left(a_{2,n+1}-a_{21}x_1-a_{22}x_2-\ldots-a_{2n}x_n\right)/a_{22}+x_2 \\ \ldots \\ x_n=\left(a_{n,n+1}-a_{n1}x_1-a_{n2}x_2-\ldots-a_{nn}x_n\right)/a_{nn}+x_n \end{cases} \quad (4)$$

Зададим начальные приближения $x_1^{\{0\}},x_2^{\{0\}},\ldots x_n^{\{0\}}$ и вычислим правую часть (4), получим новые приближения $x_1^{\{1\}},x_2^{\{1\}},\ldots x_n^{\{1\}}$, которые опять подставим в систему (4). Таким образом организуется итерационный процесс, который обрывается по условию $\left|x_k^{\{m+1\}}-x_k^{\{m\}}\right|<\xi$, где ξ – заданная погрешность.

К ускорению сходимости приводит использование приближения к решениям путем последовательного уточнения компонентов, причем *k*-я неизвестная находится из *k*-го уравнения. Такая модификация итерационного метода носит название метода Зейделя:

$$\begin{cases} x_1^{\{m+1\}}=\left(a_{1,n+1}-a_{11}x_1^{\{m\}}-a_{12}x_2^{\{m\}}-\ldots-a_{1n}x_n^{\{m\}}\right)/a_{11}+x_1^{\{m\}} \\ x_2^{\{m+1\}}=\left(a_{2,n+1}-a_{21}x_1^{\{m+1\}}-a_{22}x_2^{\{m\}}-\ldots-a_{2n}x_n^{\{m\}}\right)/a_{22}+x_2^{\{m\}} \\ \ldots \\ x_n^{\{m+1\}}=\left(a_{n,n+1}-a_{n1}x_1^{\{m+1\}}-a_{n2}x_2^{\{m+1\}}-\ldots-a_{nn}x_n^{\{m\}}\right)/a_{nn}+x_n^{\{m\}} \end{cases}$$

Критерий сходимости метода Зейделя: Пусть $\hat{A}$ – вещественная симметричная положительно определенная матрица. Тогда метод Зейделя сходится.

Достоинствами метода простых итераций является простота программной реализации и более быстрый, по сравнению с линейными методами, поиск решения в матрицах большого размера. Недостатками являются сложный контроль условий сходимости и выбора начального приближения.

8. Интерполяция функций.

Постановка задачи интерполяции.

Интерполяция каноническим полиномом.

Интерполяция полиномом Лагранжа.

Интерполяция полиномом Ньютона.

Интерполяция сплайнами.

Многомерная интерполяция.

9. Аппроксимация функций методом наименьших квадратов.

Постановка задачи аппроксимации по МНК. Условия наилучшего приближения.

Если набор экспериментальных данных получен со значительной погрешностью, то интерполяция не только не требуется, но и нежелательна! Здесь требуется построить кривую, которая воспроизводила бы график исходной экспериментальной закономерности, т.е. была бы максимально близка к экспериментальным точкам, но в то же время была бы нечувствительна к случайным отклонениям измеряемой величины.

Введем непрерывную функцию $\varphi(x)$ для аппроксимации дискретной зависимости $f(x_i)$, i = 0…n. Будем считать, что $\varphi(x)$ построена по условию *наилучшего квадратичного приближения*, если

$$Q = \sum_{i=0}^{n} \rho_i \left(\phi(x_i) - f(x_i)\right)^2 = \min \quad (1)$$

Весу ρ для i-й точки придают смысл точности измерения данного значения: чем больше ρ, тем ближе аппроксимирующая кривая «притягивается» к данной точке. В дальнейшем будем по умолчанию полагать $\rho = 1$ для всех точек.

Рассмотрим случай *линейной аппроксимации*:

$$\varphi(x) = c_0\varphi_0(x) + c_1\varphi_1(x) + \ldots + c_m\varphi_m(x), \quad (2)$$

где $\varphi_0 \ldots \varphi_m$ – произвольные *базисные функции*, $c_0 \ldots c_m$ – неизвестные коэффициенты, $m < n$. Если число коэффициентов аппроксимации взять равным числу узлов, то среднеквадратичная аппроксимация совпадет с интерполяцией Лагранжа, при этом, если не учитывать вычислительную погрешность, $Q = 0$.

Если известна экспериментальная (исходная) погрешность данных ξ, то выбор числа коэффициентов, то есть величины m, определяется условием:

$$\sqrt{Q} \approx \xi. \quad (3)$$

Иными словами, если $\sqrt{Q} >> \xi$, число коэффициентов аппроксимации недостаточно для правильного воспроизведения графика экспериментальной зависимости. Если $\sqrt{Q} << \xi$, многие коэффициенты в (2) не будут иметь физического смысла.

Для решения задачи линейной аппроксимации в общем случае следует найти условия минимума суммы квадратов отклонений для (2). Задачу на поиск минимума можно свести к задаче поиска корня системы уравнений $\frac{\partial Q}{\partial c_k} = 0$, $k = 0 \ldots m$. (4).

Подстановка (2) в (1), а затем расчет (4) приведет в итоге к следующей системе *линейных алгебраических* уравнений:

$$\begin{cases} \sum_{i=0}^{n} \left(c_0\varphi_0(x_i) + c_1\varphi_1(x_i) + \ldots + c_m\varphi_m(x_i) - f_i\right)\varphi_0(x_i) = 0 \\ \sum_{i=0}^{n} \left(c_0\varphi_0(x_i) + c_1\varphi_1(x_i) + \ldots + c_m\varphi_m(x_i) - f_i\right)\varphi_1(x_i) = 0 \\ \ldots \\ \sum_{i=0}^{n} \left(c_0\varphi_0(x_i) + c_1\varphi_1(x_i) + \ldots + c_m\varphi_m(x_i) - f_i\right)\varphi_m(x_i) = 0 \end{cases}$$

Далее следует решить полученную СЛАУ относительно коэффициентов $c_0 \ldots c_m$. Для решения СЛАУ обычно составляется расширенная матрица коэффициентов, которую называют *матрицей Грама*, элементами которой являются скалярные произведения базисных функций и столбец свободных коэффициентов:

$$\begin{bmatrix} (\varphi_0,\varphi_0) & (\varphi_0,\varphi_1) & \cdots & (\varphi_0,\varphi_m)(\varphi_0,f) \\ (\varphi_1,\varphi_0) & (\varphi_1,\varphi_1) & \cdots & (\varphi_1,\varphi_m)(\varphi_1,f) \\ \cdots & & \cdots & \cdots \\ (\varphi_m,\varphi_0) & (\varphi_m,\varphi_1) & \cdots & (\varphi_m,\varphi_m)(\varphi_m,f) \end{bmatrix},$$

где $(\varphi_j,\varphi_k)=\sum_{i=0}^{n}\varphi_j(x_i)\varphi_k(x_i)$, $(\varphi_j,f)=\sum_{i=0}^{n}\varphi_j(x_i)f(x_i)$, $j = 0\ldots m$, $k = 0\ldots m$.

После того как с помощью, например, метода Гаусса найдены коэффициенты $c_0\ldots c_m$, можно построить аппроксимирующую кривую или вычислить координаты заданной точки. Таким образом, задача аппроксимации решена.

Аппроксимация каноническим полиномом.

Выберем базисные функции в виде последовательности степеней аргумента x:

$\varphi_0(x) = x^0 = 1$; $\varphi_1(x) = x^1 = x$; $\varphi_m(x) = x^m$, $m < n$.

Расширенная матрица Грама для степенного базиса будет выглядеть следующим образом:

$$\begin{bmatrix} n+1 & \sum_{i=0}^{n}x_i & \cdots & \sum_{i=0}^{n}x_i^m & \sum_{i=0}^{n}f_i \\ \sum_{i=0}^{n}x_i & \sum_{i=0}^{n}x_i^2 & \cdots & \sum_{i=0}^{n}x_i^{m+1} & \sum_{i=0}^{n}x_i f_i \\ \cdots & & \cdots & & \cdots \\ \sum_{i=0}^{n}x_i^m & \sum_{i=0}^{n}x_i^{m+1} & \cdots & \sum_{i=0}^{n}x_i^{2m} & \sum_{i=0}^{n}x_i^m f_i \end{bmatrix}.$$

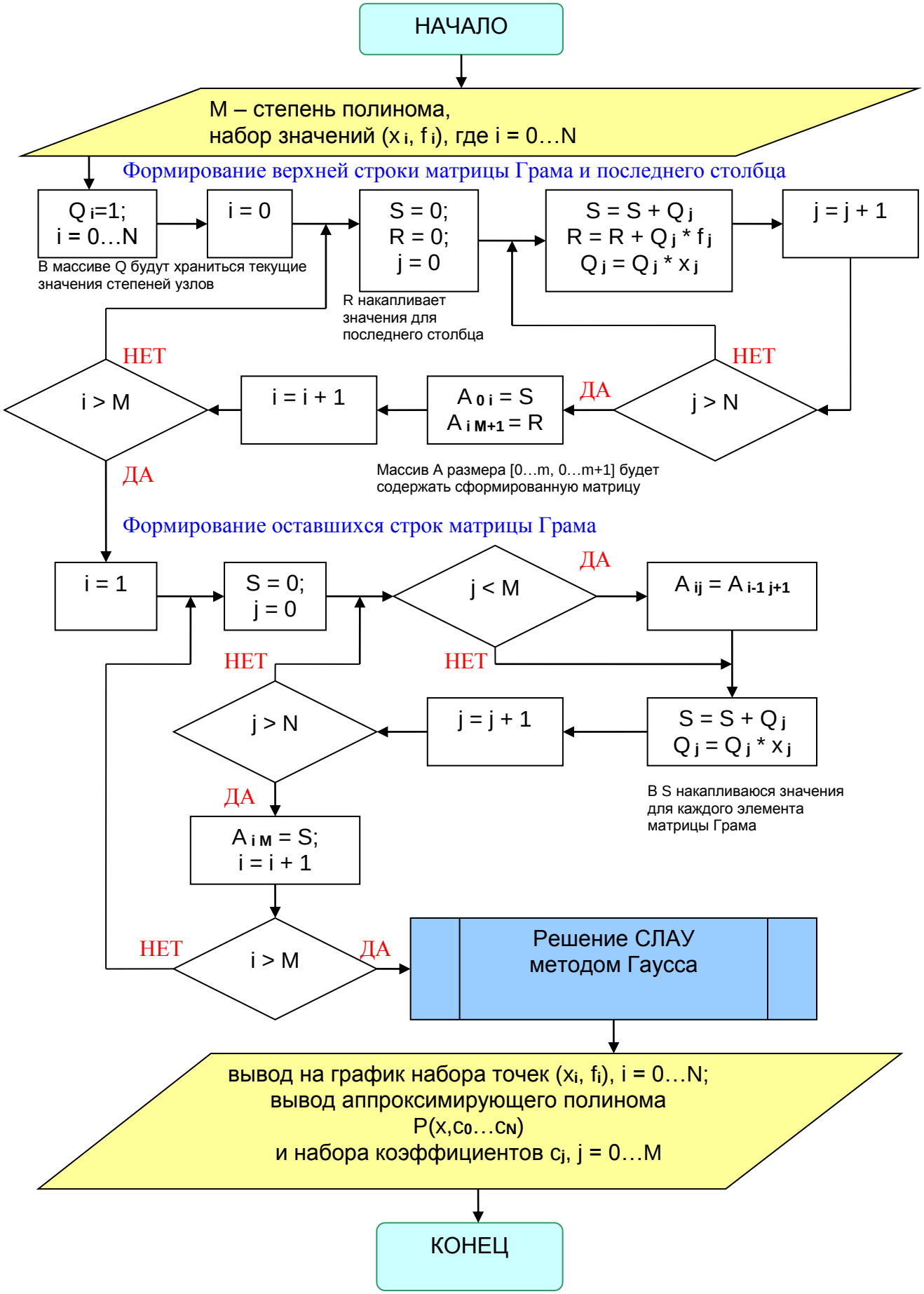
Особенность вычислений такой матрицы (для уменьшения количества выполняемых действий) состоит в том, что необходимо сосчитать только элементы первой строки и двух последних столбцов: остальные элементы заполняются сдвигом предшествующей строки (за исключением двух последних столбцов) на одну позицию влево. В некоторых языках программирования, где отсутствует быстрая процедура возведения в степень, пригодится алгоритм расчета матрицы Грама, представленный далее.

Выбор базисных функций в виде степеней *x не является оптимальным* с точки зрения достижения наименьшей погрешности. Это является следствием **неортогональности** выбранных базисных функций. Свойство *ортогональности* заключается в том, что для каждого типа полинома существует отрезок $[x_0, x_n]$, на котором обращаются в нуль скалярные произведения полиномов разного порядка:

$\int_{x_0}^{x_n}\rho(x)\varphi_j(x)\varphi_k(x)dx = 0$, $j \neq k$, ρ – некоторая весовая функция.

Если бы базисные функции были ортогональны, то все недиагональные элементы матрицы Грама были бы близки к нулю, что увеличило бы точность вычислений, в противном случае при $n \to \infty$ определитель матрицы Грама очень быстро стремится к нулю, т.е. система становится плохо обусловленной.

Блок-схема алгоритма формирования матрицы Грама и аппроксимации полиномом.



Аппроксимация ортогональными классическими полиномами.

Представленные ниже полиномы, относящиеся ко *многочленам Якоби*, обладают свойством ортогональности в изложенном выше смысле. То есть, для достижения высокой точности вычислений рекомендуется выбирать базисные функции для аппроксимации в виде этих полиномов.

1) *Полиномы Чебышева*.

Определены и ортогональны на [–1, 1] с весом $\rho = \left(1 - x^2\right)^{-\frac{1}{2}}$. В интервал ортогональности всегда можно вписать область определения исходной функции с помощью линейных преобразований.

Строятся следующим образом (рекуррентная формула):

$T_0(x) = 1$;

$T_1(x) = x$;

$T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x)$.

2) *Полиномы Лежандра*.

Определены и ортогональны на [–1, 1] с весом $\rho = 1$.

Строятся следующим образом (рекуррентная формула):

$L_0(x) = 1$;

$L_1(x) = x$;

$$L_{k+1}(x) = \frac{1}{k+1}\left((2k+1)xL_k(x) + kL_{k-1}(x)\right).$$

Сглаживание и линейная регрессия.

Рассмотрим несколько наиболее простых с точки зрения программной реализации случаев аппроксимации (сглаживания).

1) *Линейная регрессия*.

В случае линейного варианта МНК (линейная регрессия) $\varphi(x) = a + bx$ можно сразу получить значения коэффициентов a и b по следующим формулам:

$$b = \frac{\sum_{i=0}^{n}\left(x_i - \bar{x}\right)\left(f_i - \bar{f}\right)}{\sum_{i=0}^{n}\left(x_i - \bar{x}\right)^2},$$

$$a = \bar{f} - b\bar{x},$$

где $\bar{x} = \sum_{i=0}^{n}\frac{x_i}{n}$, $\bar{f} = \sum_{i=0}^{n}\frac{f_i}{n}$.

2) *Линейное сглаживание по трём точкам*.

3) *Линейное сглаживание по пяти точкам*.

10. Решение нелинейных уравнений с одним неизвестным.

Общие сведения о численном решении уравнений с одним неизвестным.

Пусть задана непрерывная функция $f(x)$. Требуется найти корни уравнения $f(x) = 0$ численными методами – это и является постановкой задачи. Численное решение уравнения распадается на несколько подзадач:

1) Анализ количества, характера и расположения корней (обычно путем построения графика функции или исходя из физического смысла исследуемой модели). Здесь возможны следующие варианты:

- единственный корень;
- бесконечное множество решений;
- корней нет;
- имеется несколько решений, как действительных, так и мнимых (например, для полинома степени n). Корни четной кратности выявить сложно.

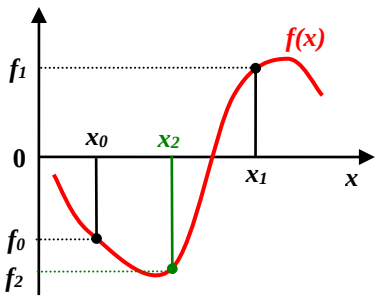
2) Локализация корней (разбиение на интервалы) и выбор начального приближения к каждому корню. В простейшем случае можно протабулировать функцию с заданным шагом.

Если в двух соседних узлах функция будет иметь разные знаки, то между этими узлами лежит нечетное число корней уравнения (по меньшей мере один).

3) Вычисление каждого (или интересующего нас) корня уравнения с требуемой точностью. Уточнение происходит с помощью методов, изложенных ниже.

Метод дихотомии (бисекций).

Иначе называется методом половинного деления. Пусть задан начальный интервал $[x_0, x_1]$, на котором $f(x_0)f(x_1) \le 0$ (т.е. внутри имеется не менее чем один корень). Найдем $x_2 = ½ (x_0 + x_1)$ и вычислим $f(x_2)$. Если $f(x_0)f(x_2) \le 0$, используем для дальнейшего деления отрезок $[x_0, x_2]$, если > 0 – используем для дальнейшего деления отрезок $[x_1, x_2]$, и продолжаем деление пополам.



Итерации продолжаются, пока длина отрезка не станет меньше 2ξ – заданной точности. Тогда середина последнего отрезка даст значение корня с требуемой точностью. В качестве иного критерия можно взять $|f(x)| \le \xi_y$.

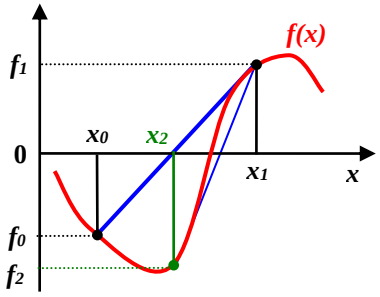
Скорость сходимости метода невелика, однако он прост и надежен. Метод неприменим к корням четной кратности. Если на отрезке несколько корней, то заранее неизвестно, к какому из них сойдется процесс.

Если на заданном интервале предполагается несколько корней, то существует возможность последовательно исключать найденные корни из рассмотрения. Для этого воспользуемся вспомогательной функцией $g(x) = \dfrac{f(x)}{x - \tilde{x}}$, где $\tilde{x}$ – только что найденный корень. Для функций $f(x)$ и $g(x)$ совпадают все корни, за исключением $\tilde{x}$ (в этой точке полюс функции $g(x)$). Для достижения требуемой точности рекомендуется грубо приблизиться к корню по функции $g(x)$, а затем уточнить его, используя $f(x)$.

Метод хорд.

Идея метода проиллюстрирована рисунком. Задается интервал $[x_0, x_1]$, на котором $f(x_0)f(x_1) \le 0$, между точками x_0 и x_1 строится хорда, стягивающая $f(x)$. Очередное приближение берется в точке x_2, где хорда пересекает ось абсцисс. В качестве нового интервала для продолжения

итерационного процесса выбирается тот, на концах которого функция имеет разные знаки. Условия выхода из итерационного цикла: $|x_k - x_{k-1}| < \xi$ или $|f(x)| \le \xi_y$.



Для вывода итерационной формулы процесса найдем точку пересечения хорды (описываемой уравнением прямой) с осью абсцисс: $ax_2 + b = 0$, где $a = \dfrac{f(x_1) - f(x_0)}{x_1 - x_0}$; $b = f(x_0) - ax_0$.

Отсюда легко выразить $x_2 = x_0 - \dfrac{x_1 - x_0}{f(x_1) - f(x_0)} f(x_0)$.

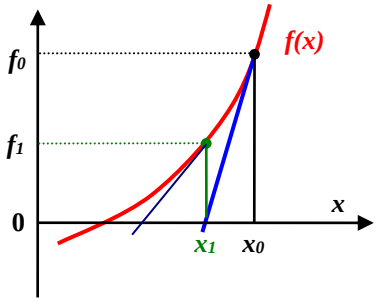
Метод хорд в большинстве случаев работает быстрее, чем метод дихотомии. Недостатки метода те же, что и в предыдущем случае.

Метод Ньютона (касательных).

Пусть x_0 – начальное приближение к корню, а $f(x)$ имеет непрерывную производную. Следующее приближение к корню найдем в точке x_1, где касательная к функции $f(x)$, проведенная из точки (x_0, f_0), пересекает ось абсцисс. Затем точно так же обрабатываем точку (x_1, f_1), организуя итерационный процесс. Выход из итерационного процесса по условию $|x_k - x_{k-1}| < \xi$.

Уравнение касательной, проведенной из точки (x_0, f_0): $y(x) = f'(x_0)(x - x_0) + f(x_0)$ дает для $y(x_1) = 0$ следующее выражение:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}, \quad (1)$$



которое и используется для организации итерационного процесса. Итерации сходятся, только если всюду выполняется условие $|f(x)f''(x)| < (f'(x))^2$; в противном случае сходимость будет не при любом начальном приближении, а только в некоторой окрестности корня. Итерации будут сходиться к корню с той стороны, с которой $|f(x)f''(x)| \ge 0$.

Метод обладает *самой высокой скоростью сходимости*: погрешность очередного приближения примерно равна квадрату погрешности предыдущего приближения. Метод можно использовать для уточнения корней в области *комплексных чисел*, что необходимо при решении многих прикладных задач, например при численном моделировании электромагнитных колебательных и волновых процессов с учетом временной и пространственной диссипации энергии.

Недостатком метода можно указать необходимость знать явный вид первой и второй производных, так как их численный расчет приведет к уменьшению скорости сходимости метода. Иногда, ради упрощения расчетов, используют т.н. *модифицированный метод Ньютона*, в котором значение $f'(x)$ вычисляется только в точке x_0, при этом число итераций увеличивается, но расчеты на каждой итерации упрощаются.

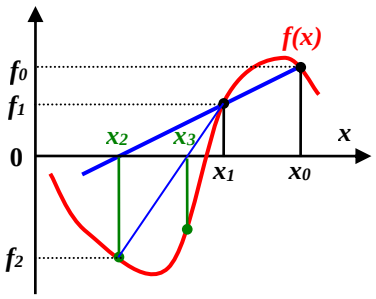
Метод секущих.

В отличие от метода Ньютона, можно заменить производную первой разделенной разностью, найденной по двум последним итерациям, т.е. заменить касательную секущей. При этом первый шаг итерационного процесса запишется так:

$$x_2 = x_1 - \frac{(x_1 - x_0) f(x_1)}{f(x_1) - f(x_0)}.$$

Для начала итерационного процесса необходимо задать x_0 и x_1, которые не обязательно ограничивают интервал, на котором функция должна менять знак; это могут быть любые две точки на кривой. Выход из итерационного процесса по условию $|x_k - x_{k-1}| < \xi$.

Сходимость может быть немонотонной даже вблизи корня. При этом вблизи корня может происходить потеря точности, т.н. «разболтка решения», особенно значительная в случае кратных корней. От разболтки страхуются *приемом Гарвика*: выбирают некоторое ξ_x и ведут итерации до выполнения условия $|x_k - x_{k-1}| < \xi_x$. Затем продолжают расчет, пока $|x_k - x_{k-1}|$ убывает. Первое же возрастание может свидетельствовать о начале разболтки, а значит, расчет следует прекратить, а последнюю итерацию не использовать.



Метод простых итераций.

Суть метода простых итераций в принципе совпадает с методом, изложенным для решения систем линейных алгебраических уравнений. Для нелинейного уравнения метод основан на переходе от уравнения

$$f(x) = 0 \quad (2)$$

к эквивалентному уравнению $x = \varphi(x)$. Этот переход можно осуществить разными способами, в зависимости от вида $f(x)$. Например, можно положить

$$\varphi(x) = x + bf(x), \quad (3)$$

где b = const, при этом корни исходного уравнения (2) не изменятся.

Если известно начальное приближение к корню x_0, то новое приближение $x_1 = \varphi(x_0)$, т.е. общая схема итерационного процесса:

$$x_{k+1} = \varphi(x_k). \quad (4)$$

Наиболее простой критерий окончания процесса $|x_{k+1} - x_k| < \xi$.

Критерий сходимости метода простых итераций: если вблизи корня $|\varphi'(x)| < 1$, то итерации сходятся. Если указанное условие справедливо для любого x, то итерации сходятся при любом начальном приближении. Исследуем выбор константы b в функции (3) с точки зрения обеспечения максимальной скорости сходимости. В соответствии с критерием сходимости наибольшая скорость сходимости обеспечивается при $|\varphi'(x)| = 0$. При этом, исходя из (3),

$b = -1/f'(x)$, и итерационная формула (4) переходит в

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

т.е. в формулу метода Ньютона (1). Таким образом, метод Ньютона является частным случаем метода простых итераций, обеспечивающим самую высокую скорость сходимости из всех возможных вариантов выбора функции $\varphi(x)$.

11. Численное решение систем нелинейных уравнений

Постановка задачи.

Требуется решить систему нелинейных уравнений $\vec{f}(\vec{x}) = 0$ (1). В координатном виде эту задачу можно записать так: $f_k(x_1, x_2, ...x_k) = 0$, где $1 \le k \le n$.

Убедиться в существовании решения и количестве корней, а также выбрать нулевое приближение в случае системы двух уравнений с двумя неизвестными можно, построив графики функций в удобных координатах. В случае сложных функций можно посмотреть поведение аппроксимирующих их полиномов. Для трех и более неизвестных, а также для комплексных корней, удовлетворительных способов подбора начального приближения нет.

Метод Ньютона.

Обозначим $\vec{x}_{(s)}$ некоторое приближение к корню системы уравнений $\tilde{\vec{x}}$. Пусть малое $\Delta\vec{x} = \tilde{\vec{x}} - \vec{x}_{(s)}$. Вблизи $\vec{x}_{(s)}$ каждое уравнение системы можно *линеаризовать* следующим образом:

$$f_k(\vec{x}) \approx f_k(\vec{x}_{(s)}) + \sum_{i=1}^{n} \frac{\partial f_k(\vec{x}_{(s)})}{\partial x_i} \Delta x_i \ , \ 1 \le k \le n. \ (2)$$

Это можно интерпретировать как первые два члена разложения функции в ряд Тейлора вблизи $\vec{x}_s$. В соответствии с (1), приравнивая (2) к нулю, получим:

$$\sum_{i=1}^{n} \frac{\partial f_k(\vec{x}_{(s)})}{\partial x_i} \Delta x_i = -f_k(\vec{x}_{(s)}) \ , \ 1 \le k \le n. \ (3)$$

Мы получили систему линейных уравнений, неизвестными в которой выступают величины $\Delta\vec{x}_i$. Решив ее, например, методом Гаусса, мы получим некое новое приближение к $\tilde{\vec{x}}$, т.е. $\vec{x}_{(s+1)} = \vec{x}_{(s)} + \Delta\vec{x}$. Выражение (3) можно представить как обобщение на систему уравнений итерационного метода Ньютона, рассмотренного в предыдущей главе:

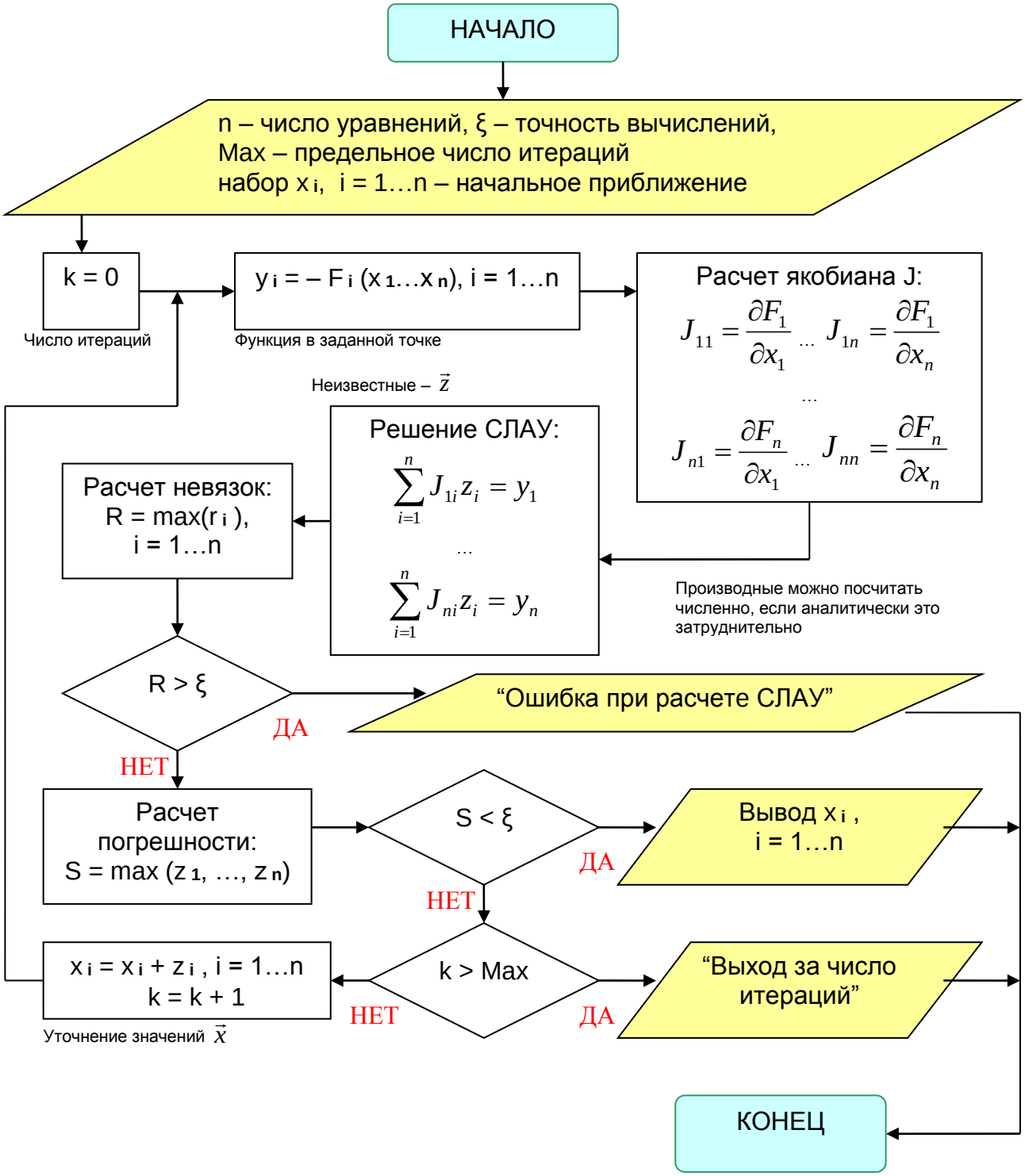
$$\vec{x}_{(s+1)} = \vec{x}_{(s)} - \hat{J}_{(s)}^{-1} \cdot f(\vec{x}_{(s)}) \ , \ (4)$$

где в данном случае

$$\hat{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_1} & \dots & \frac{\partial f_2}{\partial x_n} \\ \dots & \dots & \dots & \dots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \dots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$ – матрица Якоби, которая считается для каждого (s) приближения.

Критерием окончания итерационного процесса является условие $\left\|\vec{x}_{(s)} - \vec{x}_{(s+1)}\right\| \le \xi$ (Можем принять под $\|\vec{r}\|$ как норму $\|\vec{r}\|_1 = \frac{1}{n}\sum_{i=1}^{n}|x_i|$, так и $\|\vec{r}\|_c = \max_i |x_i|$). Достоинством метода является высокая скорость сходимости. Сходимость метода зависит от выбора начального приближения: если $\det\left[\frac{\partial\vec{f}}{\partial\vec{x}}\right] \ne 0$, то итерации сходятся к корню. Недостатком метода является вычислительная сложность: на каждой итерации требуется находить матрицу частных производных и решать систему линейных уравнений. Кроме того, если аналитический вид частных производных неизвестен, их надо считать численными методами.

Блок-схема метода Ньютона для решения систем нелинейных уравнений.



Так как метод Ньютона отличается высокой скоростью сходимости при выполнении условий сходимости, на практике критерием работоспособности метода является число итераций: если оно оказывается большим (для большинства задач >100), то начальное приближение выбрано плохо.

Частным случаем решения (4) методом Ньютона системы из двух нелинейных уравнений

$$\begin{cases} F(x,y)=0 \\ G(x,y)=0 \end{cases}$$

являются следующие легко программируемые формулы итерационного процесса:

$$\begin{cases} x_{s+1}=x_s+\dfrac{X_s}{\Im_s} \\ y_{s+1}=y_s+\dfrac{Y_s}{\Im_s} \end{cases}, \text{ где } \Im_s=\det\begin{bmatrix} \dfrac{\partial F(x_s,y_s)}{\partial x} & \dfrac{\partial F(x_s,y_s)}{\partial y} \\ \dfrac{\partial G(x_s,y_s)}{\partial x} & \dfrac{\partial G(x_s,y_s)}{\partial y} \end{bmatrix},$$

$$X_s=-\det\begin{bmatrix} F(x_s,y_s) & \dfrac{\partial F(x_s,y_s)}{\partial y} \\ G(x_s,y_s) & \dfrac{\partial G(x_s,y_s)}{\partial y} \end{bmatrix}, \; Y_s=\det\begin{bmatrix} F(x_s,y_s) & \dfrac{\partial F(x_s,y_s)}{\partial x} \\ G(x_s,y_s) & \dfrac{\partial G(x_s,y_s)}{\partial x} \end{bmatrix}$$

Метод простых итераций.

Метод простых итераций для решения (1) аналогичен методу, рассмотренному при решении нелинейных уравнений с одним неизвестным. Прежде всего, выбирается начальное приближение $\vec{x}^{(0)}$, а исходная система уравнений преобразуется к эквивалентной системе вида

$$\begin{cases} x_1^{(1)}=\varphi_1\left(x_1^{(0)},x_2^{(0)},\ldots x_n^{(0)}\right) \\ x_2^{(1)}=\varphi_2\left(x_1^{(0)},x_2^{(0)},\ldots x_n^{(0)}\right) \\ \quad\cdots \\ x_n^{(1)}=\varphi_n\left(x_1^{(0)},x_2^{(0)},\ldots x_n^{(0)}\right) \end{cases}, \quad (5)$$

и по ней осуществляется итерационный цикл. Если итерации сходятся, то они сходятся к решению уравнения (1). Обозначим $M_{kj}=\max\left|\dfrac{\partial\varphi_k}{\partial x_j}\right|$. Достаточным условием сходимости является $\|M\|<1$. Скорость сходимости метода сильно зависит от вида конкретно подбираемых функций φ_i, которые должны одновременно удовлетворять условиям эквивалентности (5) и (1), и обеспечивать сходимость итерационного процесса.

Например, для исходной системы уравнений $\begin{cases} y^3-x^2=1 \\ yx^3-x=4 \end{cases}$ эквивалентная итерационная система (5) может быть представлена в следующем виде:

$$\begin{cases} x^*=x+\nu\left(yx^3-x-4\right) \\ y^*=y+\mu\left(y^3-x^2-1\right) \end{cases},$$

где множители μ = –0.15и ν = –0.1 подбираются из анализа условий сходимости.

Метод спуска.

Рассмотрим функцию $\Phi(\vec{x})=\sum_{k=1}^{n}\left|f_k(\vec{x})\right|^2$. Она неотрицательна и обращается в нуль в том и только в том случае, если $\vec{f}(\vec{x})=0$. То есть, если мы найдем *глобальный* минимум $\Phi(\vec{x})$, то полученные значения $\vec{x}$ как раз и будут решениями уравнения (1). Подробнее о решении таких задач см. следующую главу.

12. Поиск минимума функций.

Задачи поиска максимума эквивалентны задачам поиска минимума, так как требуется лишь поменять знак перед функцией. Для поиска минимума необходимо определить интервал, на котором функция могла бы иметь минимум. Для этого можно использовать (1) графическое представление функции, (2) аналитический анализ аппроксимирующей функции и (3) сведения о математической модели исследуемого процесса (т.е. законы поведения данной функции).

Численные методы поиска минимума функции одной переменной.

Определения.

Функция $f(x)$ имеет *локальный минимум* при некотором $\tilde{x}$, если существует некоторая конечная ξ-окрестность этого элемента, в которой $f(\tilde{x}) < f(x)$, $|x - \tilde{x}| \le \xi$. Требуется, чтобы на множестве X функция $f(x)$ была по крайней мере кусочно-непрерывной.

Точка, в которой функция достигает наименьшего на множестве X значения, называется *абсолютным минимумом* функции. Для нахождения абсолютного минимума требуется найти все локальные минимумы и выбрать наименьшее значение.

Задачу называют *детерминированной*, если погрешностью вычисления (или экспериментального определения) функции $f(x)$ можно пренебречь. В противном случае задачу называют *стохастической*. Все изложенные далее методы применимы только к детерминированным задачам.

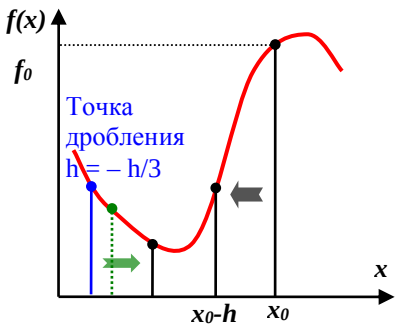
Методы поиска минимума по нахождению корней уравнений.

Если функция $f(x)$ аналитически дифференцируема, то решаем $f'(x) = 0$ методами, изложенными в предыдущих главах. При этом условие $f''(x) > 0$ в найденной точке указывает нам на минимум. Для использования этих методов необходимо знать либо аналитический вид первой и второй производных, либо рассчитать их численно, если это не приведет к потере точности.

Метод дробления.

Наиболее простой метод поиска минимума. Пусть дана начальная точка x_0, а также величина и знак шага h, определяющие движение из этой точки в сторону предполагаемого минимума $f(x)$. Метод заключается в последовательном дроблении исходного шага h с изменением его знака при выполнении условия $f(x_{k+1}) > f(x_k)$, где k – порядковый номер вычисляемой точки. Например, как только очередное значение функции стало больше предыдущего, выполняется $h = -h/3$ и процесс продолжается до тех пор, пока

$$|x_{k+1} - x_k| \le \xi \quad (1)$$



Данный метод является одним из самых медленных для поиска минимума. Основное достоинство данного алгоритма – возможность использования в программах управления экспериментальными исследованиями, когда значения функции f(x) последовательно измеряются с шагом $h \ge h_{min}$.

Метод золотого сечения.

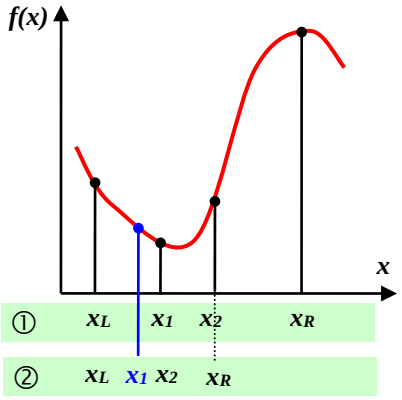
Пусть $f(x)$ задана и кусочно-непрерывна на $[x_L, x_R]$, и имеет на этом отрезке только один локальный минимум. Золотое сечение, о котором упоминал ещё Евклид, состоит в разбиении интервала $[x_L, x_R]$ точкой x_1 на две части таким образом, что отношение длины всего отрезка к его большей части равно отношению большей части к меньшей:

$$\frac{x_R - x_L}{x_R - x_1} = \frac{x_R - x_1}{x_1 - x_L}. \quad (2)$$

Таким образом, возьмем на отрезке две точки x_1 и x_2, симметрично относительно границ делящие исходный отрезок в отношении золотого сечения:

$$x_1 = x_L + (1-\tau)\cdot(x_R - x_L),$$
$$x_2 = x_L + \tau(x_R - x_L),$$

где коэффициент $\tau = \frac{\sqrt{5}-1}{2} \approx 0.618034$.



Если $f(x_1) < f(x_2)$, мы должны сузить отрезок справа, т.е. новое значение $x_R = x_2$, в противном случае $x_L = x_1$. Оставшаяся внутри нового отрезка точка является первым приближением к минимуму и делит этот отрезок в отношении золотого сечения. Таким образом, на каждой итерации приближения к минимуму (см. рисунок) нам нужно ставить только одну точку (x_1 или x_2), в которой считать значение функции и сравнивать его с предыдущим. Условием выхода из итерационного процесса будет, подобно предыдущему случаю, условие $|x_2 - x_1| \le \xi$.

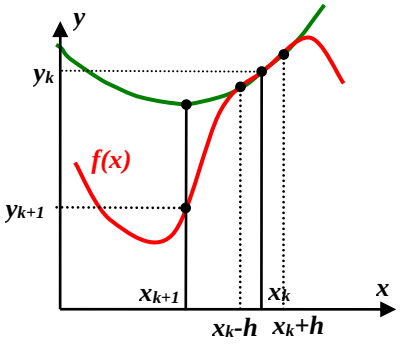
Метод отличается высокой скоростью сходимости, обычно изысканной компактностью программной реализации и всегда находит точку, минимальную на заданном интервале.

Метод парабол.

Пусть $f(x)$ имеет первую и вторую производную. Разложим $f(x)$ в ряд Тейлора в некоторой точке x_k, ограничиваясь при этом тремя членами разложения:

$$f(x) \approx f(x_k) + f'(x_k)(x - x_k) + \frac{1}{2} f''(x_k)(x - x_k)^2. \quad (3)$$

Иными словами, аппроксимируем нашу функцию в точке x_k параболой. Для этой параболы можно аналитически вычислить положение экстремума как корень уравнения первой производной от (3): $f'(x_k) + (x - x_k) f''(x_k) = 0$. Пусть минимум аппроксимирующей параболы находится в точке x_{k+1}. Тогда вычислив значение функции $f(x_{k+1})$, мы получаем новую точку приближения к минимуму.



Обычно в практических реализациях данного метода не используют аналитический вид первой и второй производных $f(x)$. Их заменяют конечно-разностными аппроксимациями. Наиболее часто берут симметричные разности с постоянным шагом h:

$$f'(x_k) \approx \frac{1}{2h}\left(f(x_k + h) - f(x_k - h)\right);$$

$$f''(x_k) \approx \frac{1}{h^2}\left(f(x_k + h) - 2f(x_k) + f(x_k - h)\right).$$

Это эквивалентно аппроксимации функции параболой, проходящей через три близкие точки x_k+h, x_k, x_k-h. Окончательное выражение, по которому можно строить итерационный процесс, таково:

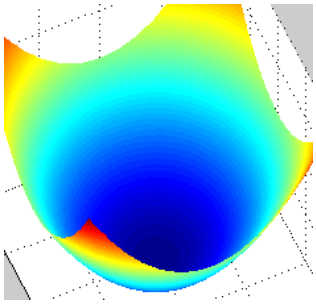
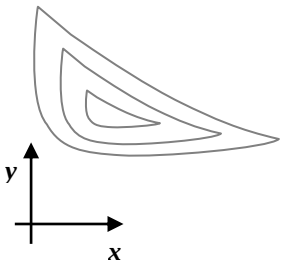
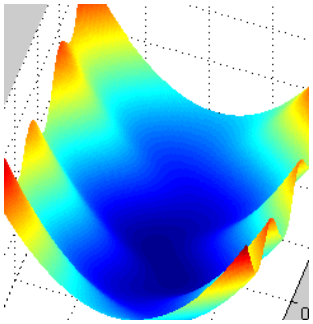
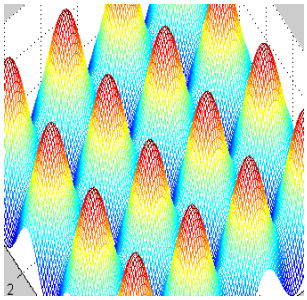
$$x_{k+1} = x_k - \frac{h}{2} \cdot \frac{f(x_k + h) - f(x_k - h)}{f(x_k + h) - 2f(x_k) + f(x_k - h)}. \quad (4)$$

Данный метод отличается от вышеизложенных высокой скоростью сходимости. Вблизи экстремума, вплоть до расстояний ~h^2, сходимость практически не отличается от квадратичной. Однако алгоритм требует постоянного контроля сходимости. Например, итерационный процесс будет сходиться к минимуму, если

1) знаменатель формулы (4) должен быть >0. Если это не так, нужно сделать шаг в обратном направлении, причем достаточно большой. Обычно в итерационном процессе полагают $h << |x_{k+1} - x_k|$. Иногда ради упрощения расчетов полагают $h = |x_{k+1} - x_k|$, однако это существенно уменьшает скорость сходимости.
2) $f(x_{k+1}) < f(x_k)$. Если это не так, то от x_k следует сделать шаг $\tau(x_{k+1} - x_k)$ с τ = ½. Если и при этом условие убывания не выполнено, уменьшают τ и вновь делают шаг.

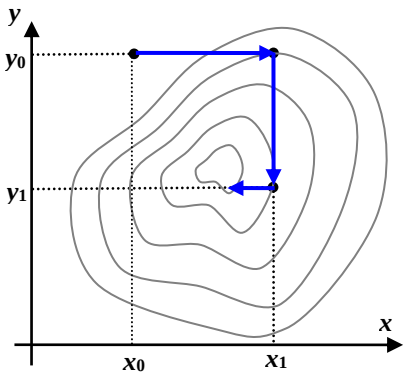
Численные методы поиска минимума функции нескольких переменных.

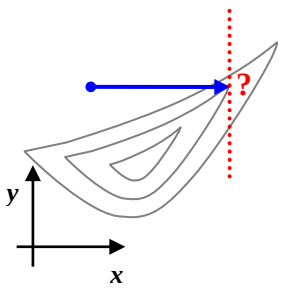
Будем рассматривать методы поиска минимума $\inf(f(\vec{r}))$ в многомерных задачах на примере функции двух переменных $f(x, y)$, так как эти методы легко аппроксимировать на случай трех и более измерений. Все эффективные методы поиска минимума сводятся к построению траекторий, вдоль которых функция убывает. Разные методы отличаются способами построения таких траекторий, так как метод, приспособленный к одному типу рельефа, может оказаться плохим для рельефа другого типа. Различают следующие типы рельефа:

1) Котловинный (гладкая функция)	2) Истинный овраг	3) Разрешимый овраг	4) Неупорядоченный
			

Метод координатного спуска.

Пусть требуется найти минимум $f(x, y)$. Выберем нулевое приближение (x_0, y_0). Рассмотрим функцию одной переменной $f(x, y_0)$ и найдем ее минимум, используя любой из рассмотренных выше способов. Пусть этот минимум оказался в точке (x_1, y_0). Теперь точно так же будем искать минимум функции одной переменной $f(x_1, y)$. Этот минимум окажется в точке (x_1, y_1). Одна итерация спусков завершена. Будем повторять циклы, постепенно приближаясь ко дну котловины, пока не выполнится условие $\max|\vec{r}_{k+1} - \vec{r}_k| < \xi$.



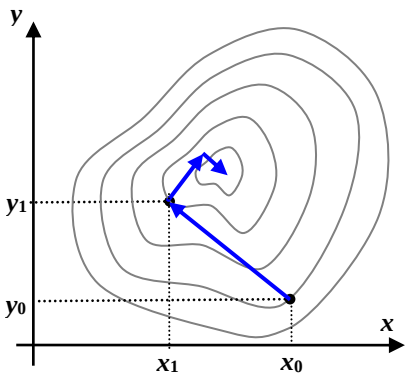


Сходимость метода зависит от вида функции и выбора нулевого приближения. Вблизи невырожденного минимума гладкой функции спуск по координатам линейно сходится к минимуму. Если линии уровня образуют истинный овраг, возможен случай, когда спуск по одной координате приводит на дно оврага, а любое движение по следующей координате ведет на подъем. Процесс координатного спуска в данном случае не сходится к минимуму.

При попадании траектории спуска в разрешимый овраг сходимость

становится чрезвычайно медленной. В физических задачах овражный рельеф указывает на то, что не учтена какая-то закономерность, определяющая связь между переменными. Явный учет этой закономерности облегчает использование численных методов.

Метод градиентного (наискорейшего) спуска.

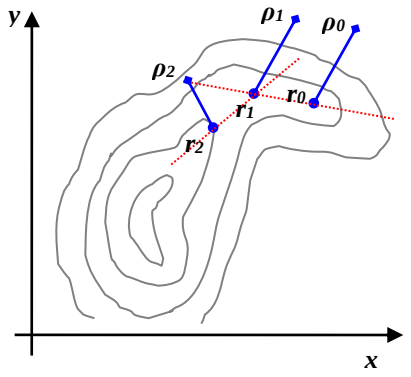


В этом методе функция минимизируется по направлению, в котором она быстрее всего убывает, т.е. в направлении, обратном $\overrightarrow{\text{grad}}\, f(\vec{r})$. Вдоль этого направления функция зависит от одной параметрической переменной *t*, для нахождения минимума которой можно воспользоваться любым известным методом поиска минимума функции одной переменной. Спуск из точки начального приближения $\vec{r}_0 = (x_0, y_0)$ против градиента до минимума *t* определяет новую точку $\vec{r}_1$, в которой вновь определяется градиент и делается следующий спуск. Условием окончания процесса, как и в случае координатного спуска, будет $\max|\vec{r}_{k+1} - \vec{r}_k| < \xi$.

С помощью метода градиентного спуска минимум гладких функций в общем случае находится быстрее, чем при использовании координатного спуска, однако нахождение градиента численными методами может свести на нет полученный выигрыш. Сходимость плохая для функций с овражным рельефом, т.е. с точки зрения сходимости градиентный спуск не лучше спуска по координатам.

Каждый спуск заканчивается в точке, где линия градиента касательна к линии (поверхности) уровня. Это означает, что каждый следующий спуск должен быть перпендикулярен предыдущему. Таким образом, вместо поиска градиента в каждой новой точке можно сосчитать градиент в начальной точке, и *развернуть оси координат* так, чтобы одна их осей была параллельна градиенту, а затем осуществлять спуск координатным методом.

Метод оврагов.



Ставится задача найти минимум $f(\vec{r})$ для овражной функции. Для этого выбираются две близкие точки $\vec{\rho}_0$ и $\vec{\rho}_1$, и осуществляется спуск из этих точек (любым методом), причем высокой точности сходимости не требуется. Конечные точки спуска $\vec{r}_0$ и $\vec{r}_1$ будут лежать вблизи дна оврага. Затем осуществляется движение вдоль прямой, соединяющей $\vec{r}_0$ и $\vec{r}_1$ в сторону уменьшения $f(\vec{r})$ (как бы вблизи дна оврага). Движение может быть осуществлено только на один шаг ~ *h*, направление выбирается из сравнения значения функции в точках $\vec{r}_0$ и $\vec{r}_1$. Таким образом, находится новая точка $\vec{\rho}_2 = \vec{r}_1 \pm (\vec{r}_1 - \vec{r}_0)h$. Так как возможно, что точка $\vec{\rho}_2$ уже лежит на склоне оврага, а не на дне, то из нее снова осуществляется спуск в новую точку $\vec{r}_2$. Затем намечается новый путь по дну оврага вдоль прямой, соединяющей $\vec{r}_2$ и $\vec{r}_1$. Если $f(\vec{r}_{k+1}) > f(\vec{r}_k)$ – процесс прекращается, а в качестве минимума в данном овраге используется значение $f(\vec{r}_k)$.

Метод оврагов рассчитан на то, чтобы пройти вдоль оврага и выйти в котловину около минимума. В этой котловине значения минимума лучше уточнять другими методами.

Проблемы поиска минимума в задачах с большим числом измерений.

Пусть в n-мерном векторном пространстве задана скалярная функция $f(\vec{r})$. Наложим дополнительные условия $\varphi_i(\vec{r}) = 0$, $1 \le i \le m$; $\psi_j(\vec{r}) \ge 0$, $1 \le j \le p$. Условия типа равенств выделяют в пространстве некоторую $(n - m)$-мерную поверхность, а условия типа неравенств выделяют n-мерную область, ограниченную гиперповерхностями $\psi_j(\vec{r}) = 0$. Число таких условий может быть произвольным. Следовательно, задача **inf** $f(\vec{r})$ есть поиск минимума функции n переменных в некоторой $(n - m)$-мерной области $\mathfrak{E}$. Функция может достигать минимального значения как внутри области, так и на ее границе. Однако перейти к $(n - m)$-мерной системе координат практически никогда не удается, поэтому спуск приходится вести во всем n-мерном пространстве.

Даже если нулевое приближение лежит в области $\mathfrak{E}$, естественная траектория спуска сразу выходит из этой области. Для принудительного возврата процесса в область $\mathfrak{E}$, например, используется *метод штрафных функций*: к $f(\vec{r})$ прибавляются члены, равные нулю в $\mathfrak{E}$, и возрастающие при нарушении дополнительных условий (ограничений). Метод прост и универсален, однако считается недостаточно надежным. Более качественный результат дает использование *симплекс-методов* линейного программирования, однако данный вопрос в рамках настоящего курса не рассматривается.

13. Решение обыкновенных дифференциальных уравнений.

Постановка задачи. Типы задач для ОДУ.

Известно, что с помощью дифференциальных уравнений можно описать задачи движения системы взаимодействующих материальных точек, химической кинетики, электрических цепей, и др. Конкретная прикладная задача может приводить к дифференциальному уравнению любого порядка, или к системе уравнений любого порядка. Так как любое ОДУ порядка p $u^{(p)}(x) = f(x, u, u', u'', \dots u^{(p+1)})$ заменой $u^{(k)}(x) = y_k(x)$ можно свести к эквивалентной системе из p уравнений первого порядка, представленных в *каноническом* виде :

$y'_k(x) = y_{k+1}(x)$ для $0 \le k \le p-2$ (1)

$y'_{p-1}(x) = f(x, y_0, y_1, \dots y_{p-1})$, при этом $y_0(x) \equiv u(x)$. (2)

Покажем такое преобразование на примере уравнения Бесселя: $y'' + \frac{y'}{x} + \left(1 - \frac{p^2}{x^2}\right) y = 0$.

Предполагая тождественную замену $y_1(x) \equiv y(x)$ представим систему ОДУ в следующем виде:

$$\begin{cases} y'_1(x) = y_2(x) \\ y'_2(x) = \left(\frac{p^2}{x^2} - 1\right) y_1(x) - \frac{y_2(x)}{x} \end{cases}$$

Аналогично произвольную систему дифференциальных уравнений любого порядка можно заменить некоторой эквивалентной системой уравнений первого порядка. Следовательно, алгоритмы численного решения достаточно реализовать для решения системы дифференциальных уравнений первого порядка.

Известно, что система p-го порядка имеет множество решений, которые в общем случае зависят от p параметров $\{C_1, C_2, \dots C_p\}$. Для определения значений этих параметров, т.е. для выделения единственного решения необходимо наложить p дополнительных условий на $u_k(x)$. По способу задания условий различают три вида задач, для которых доказано существование и единственность решения. Это

1) *Задача Коши*. Задается координата $u_k(x_0) = u_{k0}$ начальной точки интегральной кривой в $(p+1)$ – мерном пространстве ($k = 1 \dots p$). Решение при этом требуется найти на некотором отрезке $x_0 \le x \le x_{max}$.
2) *Краевая задача*. Это задача отыскания частного решения системы ОДУ на отрезке $a \le x \le b$, в которой дополнительные условия налагаются на значения функции $u_k(x)$ более чем в одной точке этого отрезка.
3) *Задача на собственные значения*. Кроме искомых функций и их производных в уравнение входят дополнительно m неизвестных параметров $\lambda_1, \lambda_2, \dots \lambda_m$, которые называются собственными значениями. Для единственности решения на некотором интервале необходимо задать $p+m$ граничных условий.

В большинстве случаев необходимость численного решения систем ОДУ возникает в случае, когда *аналитическое решение* найти либо невозможно, либо нерационально, а *приближенное решение* (в виде набора интерполирующих функций) не дает требуемой точности. *Численное решение* системы ОДУ, в отличие от двух вышеприведенных случаев, никогда не покажет общего решения системы, так как даст только таблицу значений неизвестных функций, удовлетворяющих начальным условиям. По этим значениям можно построить графики данных функций или рассчитать для некоторого $x > x_0$ соотвествующие $u_k(x)$, что обычно и требуется в физических или инженерных задачах. При этом требуется, чтобы соблюдались условия *корректно поставленной задачи*.

Метод Эйлера (метод ломаных).

Рассмотрим задачу Коши для уравнения первого порядка:

$$u'(x) = f(x, u(x)),\ x_0 \le x \le x_{max},\ u(x_0) = u_0. \quad (3)$$

В окрестности точки x_0 функцию $u(x)$ разложим в ряд Тейлора:

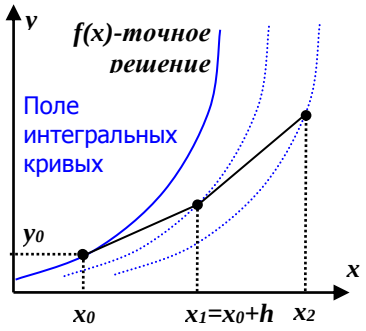
$$u(x) = u(x_0) + u'(x_0)(x - x_0) + \frac{1}{2}u''(x_0)(x - x_0)^2 + \ldots \quad (4)$$

Идея этого и последующих методов основывается на том, что если $f(x,u)$ имеет q непрерывных производных, то в разложении можно удержать члены вплоть до $O(h^{q+1})$, при этом стоящие в правой части производные можно найти, дифференцируя (3) требуемое число раз. В случае метода Эйлера ограничимся только двумя членами разложения.

Пусть h – малое приращение аргумента. Тогда (4) превратится в

$$u(x_0 + h) = u(x_0) + h \cdot u'(x_0) + O(h^2).$$

Так как в соответствии с (3) $u'(x_0) = f(x_0, u_0)$, то $u(x_0 + h) \approx u_0 + h \cdot f(x_0, u_0)$.



Теперь приближенное решение в точке $x_1 = x_0 + h$ можно вновь рассматривать как начальное условие, т.е. организуется расчет по следующей рекуррентной формуле:

$$y_{k+1} = y_k + h \cdot f(x_k, y_k) \quad (5),$$

где $y_0 = u_0$, а все y_k – приближенные значения искомой функции (см. рисунок). В методе Эйлера происходит движение не по интегральной кривой, а по касательной к ней. На каждом шаге касательная находится уже для новой интегральной кривой (что и дало название методу – метод ломаных), таким образом ошибка будет возрастать с отдалением x от x_0.

При $h \to 0$ приближенное решение сходится к точному равномерно с первым порядком точности. То есть, метод дает **весьма низкую точность вычислений**: погрешность на элементарном шаге h составляет $\frac{1}{2} h^2 y''(\frac{1}{2}(x_k + x_{k+1}))$, а для всей интегральной кривой порядка h^1. При h = const для оценки апостериорной погрешности может быть применена первая формула Рунге, хотя для работы метода обеспечивать равномерность шага в принципе не требуется.

Метод Эйлера легко обобщается для систем ОДУ. При этом общая схема процесса (5) может быть записана так:

$$y_{i\,k+1} = y_{i\,k} + h \cdot f_i\left(x_k, y_{1\,k}, y_{2\,k}, \ldots, y_{m\,k}\right) \quad (6),$$

где $i = 1 \ldots m$ – число уравнений, k – номер предыдущей вычисленной точки.

Усовершенствованный метод Эйлера-Коши с уточнением.

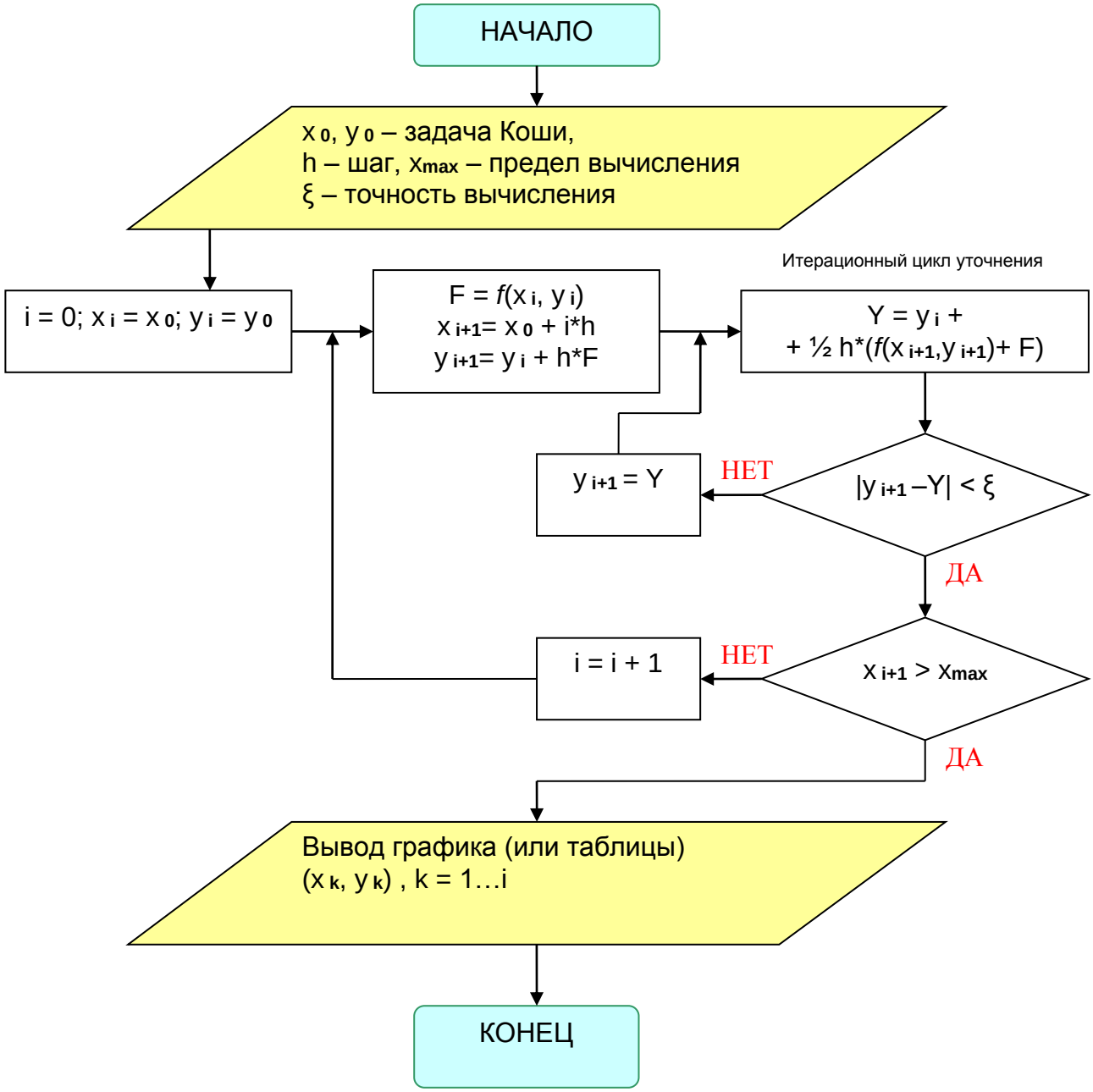
Данный метод базируется на предыдущем, однако здесь апостериорная погрешность контролируется на каждом шаге вычисления. Как и в предыдущем случае, рассматриваем задачу Коши на сетке с постоянным шагом h. Грубое значение $y_{k+1}^{\{0\}}$ вычисляется по формуле (5) и затем итерационным циклом уточняется по формуле

$$y_{k+1}^{\{m+1\}} = y_k^{\{m\}} + \frac{1}{2}h \cdot \left(f\left(x_k, y_k^{\{m\}}\right) + f\left(x_{k+1}, y_{k+1}^{\{m\}}\right)\right), \quad (7)$$

где m – номер итерации. Итерационный цикл повторяется до тех пор, пока $\left|y_{k+1}^{\{m+1\}} - y_{k+1}^{\{m\}}\right| < \varepsilon$.

Данная формула также легко обобщается на решение систем ОДУ. Априорная погрешность метода при $m = 1$ на каждом шаге порядка h^3.

Блок-схема метода Эйлера-Коши с уточнением.



Метод Рунге-Кутты II порядка.

Увеличение точности решения ОДУ из предыдущей задачи при заданном шаге h может быть достигнуто учетом большего количества членов разложения функции в ряд Тейлора. Для метода Рунге-Кутты второго порядка следует взять три первых коэффициента, т.е. обеспечить:

$$u_{k+1} = u_k + hu'_k + \frac{h^2}{2}u''_k + O(h^3)\,. \quad (8)$$

Переходя к приближенному решению $y \approx u$ и заменяя производные в (8) конечными разностями, получаем в итоге следующее выражение:

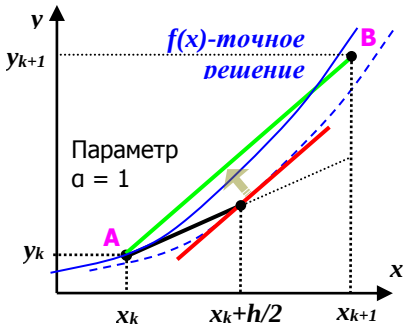
$$y_{k+1} = y_k + h \cdot \left((1-\alpha) f(x_k, y_k) + \alpha \cdot f\left(x_k + \frac{h}{2}, y_k + \frac{h}{2\alpha} f(x_k, y_k) \right) \right), \quad (9)$$

где $0 \le \alpha \le 1$– свободный параметр. Можно показать, что если $f(x,u)$ непрерывна и ограничена вместе со своими вторыми производными, то решение, полученное по данной схеме, равномерно сходится к точному решению с погрешностью порядка h^2.

Для параметра α наиболее часто используют следующие значения:

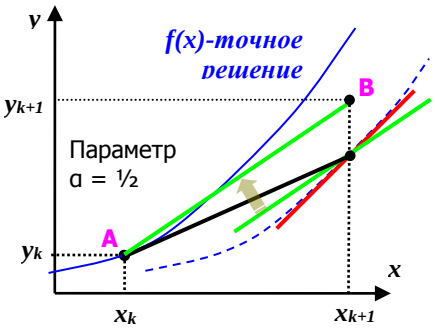
1) **α = 1**. В этом случае

$y_{k+1} = y_k + h \cdot f\left(x_k + \frac{h}{2}, y_k + \frac{h}{2} f(x_k, y_k)\right)$. Графически это уточнение можно интерпретировать так: сначала по схеме ломаных делается шаг $h/2$, и находится значение $y(x_k + \frac{h}{2}) = y_k + \frac{h}{2} f_k$. В найденной точке определяется наклон касательной к интегральной кривой, который и будет определять приращение функции для целого шага, т.е. отрезок [AB] (см. рисунок) будет параллелен касательной, проведенной в точке $(x_k + h/2, y(x_k + h/2))$ к интегральной кривой.



2) **α = ½**. В этом случае

$y_{k+1} = y_k + \frac{h}{2} \cdot \left(f(x_k, y_k) + f(x_k + h, y_k + h \cdot f_k)\right)$. Можно представить, что в этом случае по методу Эйлера сначала вычисляется значение функции и наклон касательной к интегральной кривой в точке x_{k+1}. Затем находится среднее положение касательной из сравнения соответствующих наклонов в точках x_k и x_{k+1}, которое и будет использоваться для расчета точки y_{k+1}.



<u>Метод Рунге-Кутты IV порядка.</u>

Данная схема является наиболее употребительной. Здесь в разложении функции в ряд Тейлора учитываются члены до h^4 включительно, т.е. погрешность на каждом шаге пропорциональна h^5. Для практических вычислений используются следующие соотношения, обобщенные в данном случае на решение системы ОДУ:

$y_{\underset{i}{k+1}} = y_{\underset{i}{k}} + \frac{1}{3}(q_{i1} + 2q_{i2} + q_{i3} + q_{i4})$, где $i = 1 \ldots p$, p – число уравнений в системе.

$q_{i1} = \frac{h}{2} f_i\left(x_k, y_{\underset{1}{k}}, y_{\underset{2}{k}}, \ldots, y_{\underset{p}{k}}\right)$; k – номер точки, для которой осуществляется расчет;

$q_{i2} = \frac{h}{2} f_i\left(x_k + \frac{h}{2}, y_{\underset{1}{k}} + q_{11}, y_{\underset{2}{k}} + q_{21}, \ldots, y_{\underset{p}{k}} + q_{p1}\right)$;

$q_{i3} = h \cdot f_i\left(x_k + \frac{h}{2}, y_{\underset{1}{k}} + q_{12}, y_{\underset{2}{k}} + q_{22}, \ldots, y_{\underset{p}{k}} + q_{p2}\right)$;

$q_{i4} = \frac{h}{2} f_i\left(x_k + h, y_{\underset{1}{k}} + q_{13}, y_{\underset{2}{k}} + q_{23}, \ldots, y_{\underset{p}{k}} + q_{p3}\right)$.

К достоинствам метода следует отнести высокую точность вычислений. Схемы более высокого порядка точности практически не употребляются в силу своей громоздкости. Также немаловажно, что метод является *явным*, т.е. значение y_{k+1} вычисляется по ранее найденным значениям за известное заранее число действий.

Все представленные выше схемы допускают расчет с переменным шагом. Например, шаг можно уменьшить там, где функция быстро изменяется, и увеличить в обратном случае. Так, *метод Рунге-Кутты-Мерсона* позволяет оценивать погрешность на каждом шаге и, в зависимости от полученной оценки принимать решение об изменении шага. Автоматический выбор шага позволяет значительно сократить время вычислений.

Метод Адамса.

Метод основан на аппроксимации интерполяционными полиномами правых частей ОДУ.

Пусть с помощью любого из методов, рассмотренных выше, вычислено решение заданного дифференциального уравнения в точках x_1, x_2, x_3 (а в точке x_0 решение и так известно – поставлена задача Коши). Полученные значения функции обозначим как y_0, y_1, y_2, y_3, а значения правой части дифференциального уравнения как f_0, f_1, f_2, f_3, где $f_k = f(x_k, y_k)$. Начиная с четвертой точки, на каждом шаге интегрирования дифференциального уравнения вычисления осуществляются по схеме

$$\mathbf{P(EC)^{\{m\}}E}$$

где P – прогноз решения; E – вычисление f(x,y); C – коррекция решения; m – количество итераций коррекции. Схемы такого типа называют *«прогноз-коррекция»*: это подразумевает сначала приблизительное вычисление решение по формуле низкого порядка, а затем уточнение с учетом полученной информации о поведении интегральной кривой.

Прогноз осуществляется по экстраполяционной формуле Адамса:

$$y_k = y_{k-1} + \frac{h}{24}\left(55 f_{k-1} - 59 f_{k-2} + 37 f_{k-3} - 9 f_{k-4}\right) + O(h^5). \quad (10)$$

Коррекция осуществляется по интерполяционной формуле Адамса:

$$y_k = y_{k-1} + \left(9 f_{k-1} + 19 f_{k-2} - 5 f_{k-3} + f_{k-4}\right) + O(h^5). \quad (11)$$

Вычисление осуществляется по формуле:

$$f_k = f(x_k, y_k)$$

Количество итераций $m \le p$, где p – порядок используемого метода. В ходе каждой итерации решается *нелинейное уравнение* (11) относительно неизвестной y_4 (обычно методом простых итераций).

Иногда в методе Адамса используется схеме **PECE** на каждом шаге процесса интегрирования, т.е. осуществляется только одна коррекция. В силу сложности вычислений метод используется только в мощных программных пакетах численного анализа. Формулы метода также легко переносятся на решение систем ОДУ первого порядка.

Постановка краевой задачи. Метод стрельбы.

Краевая задача – задача отыскания частного решения системы

$u'_k(x) = f_k\left(x, u_1, u_2, ..., u_p\right)$ (12), $1 \le k \le p$, на отрезке $a \le x \le b$, на котором дополнительные условия налагаются на значения функции $u_k(x)$ более чем в одной точке этого отрезка. В качестве примера моно привести задачу нахождения статического прогиба $u(x)$ нагруженной струны с закрепленными концами:

$u''(x) = -f(x)$, $a \le x \le b$, $u(a) = u(b) = 0$. Здесь $f(x)$ имеет смысл внешней изгибающей нагрузки на единицу длины струны, деленной на упругость струны.

Найти точное решение краевой задачи в элементарных функциях удается редко: для этого надо найти общее решение системы (12) и суметь явно определить из краевых условий значения входящих в него постоянных. Одним из методов, предполагающих *численное решение* поставленной задачи, является *метод стрельбы*, в котором краевая задача для системы (12) сводится к *задаче Коши* для той же системы.

Рассмотрим систему двух дифференциальных уравнений первого порядка

$$\begin{cases} u'(x) = f(x,u,v) \\ v'(x) = g(x,u,v) \end{cases} \quad (13)$$

с краевыми условиями

$$\begin{cases} \varphi(u(a),v(a)) = 0 \\ \psi(u(b),v(b)) = 0 \end{cases} \quad (14)$$

Сначала выберем некоторое значение $\alpha = u(a)$, так что $\varphi(\alpha, v(a)) = 0$. Это уравнение мы можем решить и определить $\beta(\alpha) = v(a)$. Таким образом у нас появились два числа α и β, которые будут определять задачу Коши $u(a) = \alpha$, $v(a) = \beta$ для системы (13), удовлетворять *левому* краевому условию, но не удовлетворять второму уравнению (14). Тем не менее, решим систему ОДУ с задачей Коши каким-либо из известных нам численных методов. Получим решение вида

$\begin{cases} u(x,\alpha) \\ v(x,\alpha) \end{cases}$ (15), зависящее от α как от параметра.

Теперь мы должны каким-либо способом менять параметр α до тех пор, пока не подберется такое значение, для которого будет выполнено условие $\psi(u(b,\alpha),v(b,\alpha)) = 0$ (16), т.е. *правое* краевое условие. Для этого *случайным образом* берут значения α_i до тех пор, пока среди величин $\psi(\alpha_i)$ не окажется *разных по знаку*.

Если это осуществилось, пара таких значений определяет интервал $[\alpha_i, \alpha_{i+1}]$, который можно обработать методом дихотомии до получения корня уравнения (16). Нахождение каждого нового значения функции (16) требует нового численного интегрирования для решения ОДУ, что делает этот метод достаточно медленным.

Решение уравнений в частных производных.

К уравнениям в частных производных приводят задачи газодинамики, теплопроводности, переноса излучения, электромагнитных полей, процессов переноса в газах, и др. Независимыми переменными в физических задачах обычно являются время *t*, координаты $\vec{r}$, скорости частиц $\vec{v}$. Пример – уравнение теплопроводности

$$C\frac{\partial U}{\partial t} = div\left(K \cdot \overrightarrow{grad U}\right) + q, \quad (17)$$

где U – температура, $C(U,\vec{r},t)$ – теплоемкость, $K(U,\vec{r},t)$ – коэффициент теплопроводности, q – плотность источников тепла.

Для решения дифференциальных уравнений в частных производных применяется *сеточный метод*, суть которого – в разбиении области, в которой ищется решение, сеткой узлов заданной конфигурации, после чего составляется *разностная схема* уравнения и находится его решение, например методом *разностной аппроксимации*.

Рассмотрим в качестве примера одномерную задачу, близкую по смыслу к (17):

$$\frac{\partial U}{\partial t} = K\frac{\partial^2 U}{\partial x^2}. \quad (18)$$

Здесь $0 \le x \le a$, $0 \le t \le$ T.

Граничные условия:

$U(x,0) = \mu(x);$

$U(0,t) = \mu_1(t);$

$U(a,t) = \mu_2(t).$

Для одной и той же задачи можно составить много разностных схем. Метод разностной аппроксимации заключается в том, что каждая производная, входящая в дифференциальное уравнение и краевые условия, заменяется разностным выражением, включающим в себя только значения в узлах сетки.

Введем равномерную прямоугольную сетку по x и t с шагом h и τ соответственно и заменить производные соответствующими разностными отношениями. Тогда

$$\frac{1}{\tau}(y_{\underset{k}{m+1}} - y_{\underset{k}{m}}) = \frac{K}{h^2}\left(y_{\underset{k+1}{m+1}} - 2y_{\underset{k}{m+1}} + y_{\underset{k-1}{m+1}} \right). \quad (19)$$

Здесь $1 \le k \le N$-1 – число точек по координате x; $0 \le m \le M$ – число точек по координате t. Число неизвестных в (19) больше числа уравнений, недостающие уравнения выводятся из начальных и граничных условий:

$y_{\underset{k}{0}} = \mu(x_k)$; $0 \le k \le N.$

$y_{\underset{0}{m+1}} = \mu_1(t_{m+1})$; $y_{\underset{N}{m+1}} = \mu_2(t_{m+1})$; $0 \le m \le M.$

Схема (19) содержит в каждом уравнении несколько неизвестных значений функции. Такие схемы называют *неявными*. Для фактического вычисления решения следует переписать схему так:

$$y_{\underset{k-1}{m+1}} - (2 + \frac{h^2}{K\tau}) \cdot y_{\underset{k}{m+1}} + y_{\underset{k+1}{m+1}} = \frac{h^2}{K\tau} y_{\underset{k}{m}}, \text{ где } 1 \le k \le N\text{-}1.$$

$$y_{\underset{0}{m+1}} = \mu_1(t_{m+1}); \; y_{\underset{N}{m+1}} = \mu_2(t_{m+1}). \quad (20)$$

Теперь схема представляет собой систему линейных уравнений для определения величин $y_{\underset{k}{m+1}}$; правые части этих уравнений известны, поскольку содержат значения решений для предыдущего индекса времени.

Другим вариантом решения сеточной задачи является использование *интегро-интерполяционных методов* (методов баланса), в которых дифференциальное уравнение интегрируют по ячейке сетки, приближенно вычисляя интегралы по квадратурным формулам.

Приложение 1.
Случайные величины в компьютерном моделировании.

Параметры случайных величин.

Величину ξ называют случайной с плотностью распределения *ρ(x)*, если вероятность того, что величина примет значения между x_1 и x_2, равна $\int_{x_1}^{x_2} \rho(x)dx$.

Псевдослучайные распределения. Иллюстрация возможностей Mathcad.

Моделирование броуновского движения в системе MathLab.

Литература

1. Н. Бахвалов, Н. Жидков, Г. Кобельков. Численные методы. М., 2002, 632 с.
2. Н. Калиткин. Численные методы. М., 1972,
3. А. Мудров. Численные методы для ПЭВМ на языках Бейсик, Фортран и Паскаль. Томск, 1992, 270с.
4. А. Самарский. Введение в численные методы. М., , 270с.
5. Ю. Тарасевич. Численные методы на Mathcad'е. Астрахань, 2000, 70с.
6. Г. Коткин, В. Черкасский. Компьютерное моделирование физических процессов с использованием MathLab. Новосибирск, 2001.
7. М. Лапчик, М. Рагулина, Е. Хеннер. Численные методы.М., 2004, 384с.

Во всех делах твоих помни о конце твоем и во век не согрешишь.

Библия, “Книга Премудрости
Иисуса, сына Сирахова” (некан.), **7**, 39