

Mathmod - библиотека программ для численного моделирования

Баранов С.В.

30.12.2024

Содержание

| | |
|---|----------|
| Mathmod - библиотека программ для численного моделирования | 1 |
| Теория погрешностей | 1 |
| Решение нелинейных уравнений | 2 |
| Постановка задачи | 2 |
| Основные вопросы | 2 |
| 1. Метод Ньютона | 3 |
| 2. Упрощённый метод Ньютона | 3 |
| 3. Метод секущих | 5 |
| 4. Метод ложного положения | 8 |
| 5. Метод бисекций | 10 |
| 6. Метод простой итерации | 10 |
| Решение систем линейных алгебраических уравнений (СЛАУ) | 12 |
| Постановка задачи | 12 |
| Основные вопросы | 14 |
| Прямые методы | 16 |
| Итерационные методы | 31 |
| Приближение функций | 38 |
| Установка | 40 |

Mathmod - библиотека программ для численного моделирования

Теория погрешностей

Пусть a - точное решение. a^* - приближенное значение. Тогда абсолютная погрешность:

Абсолютная погрешность:

$$\Delta(a^*) = |a - a^*|$$

Относительная погрешность:

$$\delta(a^*) = \frac{|a - a^*|}{|a|} = \frac{\Delta(a^*)}{|a|}$$

Верная цифра - значащая цифра числа a^* , абсолютная погрешность которой не превосходит единицы разряда, соответствующей этой цифре.

Округление - замена числа a его другим числом a^* с меньшим количеством значащих цифр.

При округлении возникает *погрешность округления*.

Решение нелинейных уравнений

Постановка задачи

Задача отыскания корней нелинейного уравнения с одним неизвестным вида:

$$f(x) = 0$$

Корень уравнения - значение \bar{x} , при котором $f(\bar{x}) = 0$. Геометрически, \bar{x} является абсциссой точки пересечения графика $f(x)$ с осью Ox .

Для заданной точности ϵ требуется найти приближенное значение корня \bar{x}^* такое, что:

$$|\bar{x} - \bar{x}^*| \leq \epsilon$$

Пусть функция $f(x)$ дифференцируема $m > 1$ раз в точке \bar{x} , тогда:

Простой корень - корень уравнения \bar{x} называется простым, если $f'(\bar{x}) \neq 0$.

Кратный корень - корень уравнения \bar{x} называется кратным, если $f'(\bar{x}) = 0$. Целое число m назовем кратностью корня \bar{x} , если $f^{(k)}(\bar{x}) = 0$, для $k = 1, 2, \dots, m-1$ и $f^{(m)}(\bar{x}) \neq 0$.

Этапы решения уравнения: 1. Локализация корня; 2. Вычисление корня с заданной точностью.

Отрезок локализации - отрезок $[a, b]$, который содержит 1 корень уравнения.

Основные вопросы

Сходящийся итерационный процесс - метод отыскания корня \bar{x} , заключающийся в построении последовательности приближений к нему $x^{(0)}, x^{(1)}, \dots, x^{(k)}, \dots$ такой, что:

$$\lim_{k \rightarrow \infty} x^{(k)} = \bar{x}$$

Порядок сходимости:

Пусть в некоторой малой окрестности корня \bar{x} уравнения $f(x) = 0$ итерационная последовательность удовлетворяет неравенству:

$$|\bar{x} - x^{(k)}| \leq C |\bar{x} - x^{(k-1)}|^p$$

, где $C > 0$ и $p \geq 1$ - постоянные. Тогда p называется *порядком сходимости*.

Порядок сходимости - скорость уменьшения погрешности между последовательными приближениями решения.

Одношаговый итерационный метод - метод, у которого очередное приближение $x^{(k+1)}$ находится только через одно предыдущее $x^{(k)}$. Для его работы нужно знать только одно начальное приближение $x^{(0)}$. (Пример - *метод Ньютона*)

Многошаговый итерационный метод (l - шаговый) - метод, у которого очередное приближение $x^{(k+1)}$ находится l предыдущих $x^{(k)}, x^{(k-1)}, \dots, x^{(k-l+1)}$. Для него следует задать l начальных приближений $x^{(0)}, x^{(1)}, \dots, x^{(l-1)}$. (Пример - *метод секций*)

Интервал неопределенности - окрестность $(\bar{x} - \epsilon, \bar{x} + \epsilon)$ внутри которой любую точку можно принять за приближение к корню.

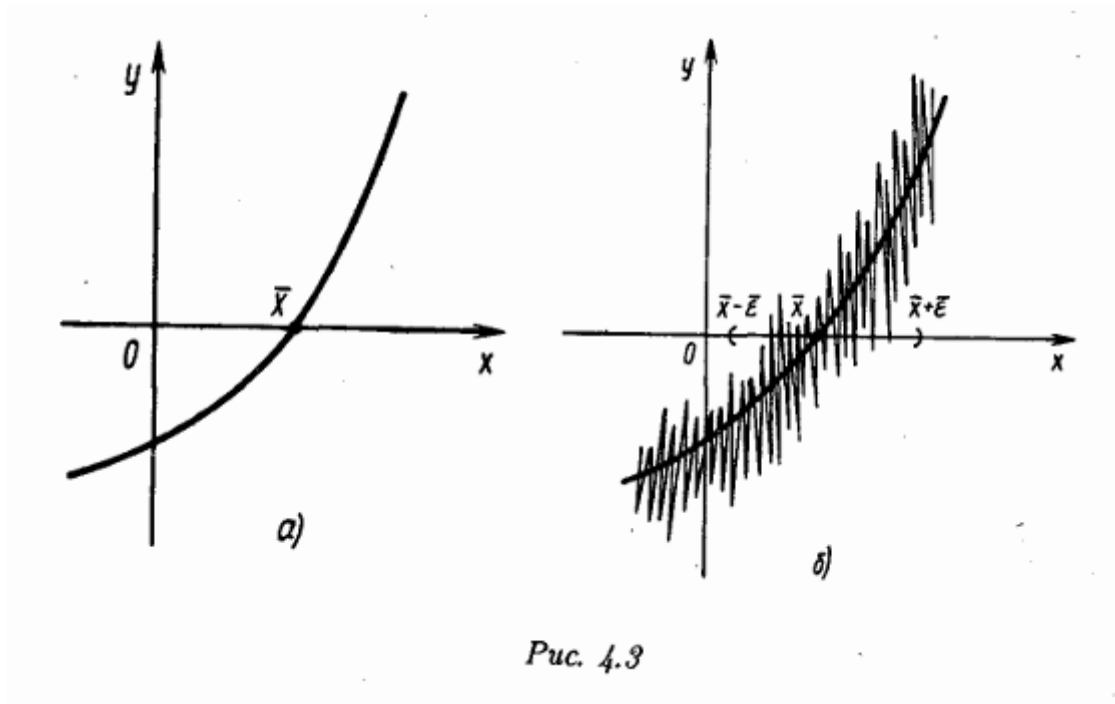


Figure 1: Интервал неопределенности

1. Метод Ньютона

- Быстрый итерационный метод для нахождения корня уравнения $f(x) = 0$.
- Требуется предоставления функции $f(x)$ и её производной $f'(x)$.
- **Функция:** `newton(f, df, x, epsilon=1e-6)`
- **Описание параметров:**
 - `f` - Функция, корень которой нужно найти;
 - `df` - Производная функции;
 - `x` - Начальное приближение корня;
 - `epsilon` - Заданная точность (по умолчанию 10^{-6}).

```
from mathmod.nonlinear_equations import newton
```

```
x, iteration = newton(f, df, x, epsilon=1e-6)
```

Расчетная формула:

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})}$$

Сходимость метода: квадратичная (при выборе начального приближения из достаточно малой окрестности).

Критерий окончания:

$$|x^{(n+1)} - x^{(n)}| < \varepsilon$$

2. Упрощённый метод Ньютона

- Упрощённая версия метода Ньютона, где производная функции вычисляется только один раз.

1. **Метод касательных.** Выведем расчетную формулу метода для решения нелинейного уравнения (4.1) из простых геометрических соображений. Соответствующая иллюстрация приведена на рис. 4.9.

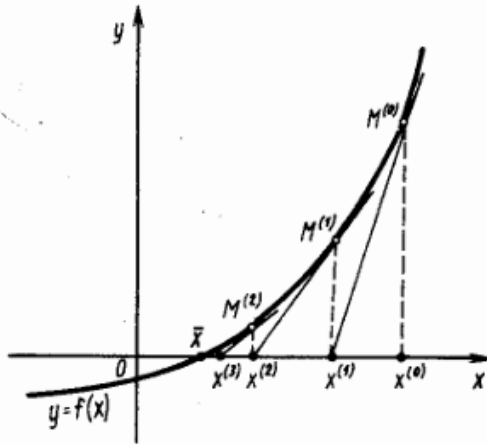


Рис. 4.9

Пусть $x^{(0)}$ — заданное начальное приближение к корню \bar{x} . В точке $M^{(0)}$ с координатами $(x^{(0)}, f(x^{(0)}))$ проведем касательную к графику функции $y = f(x)$ и за новое приближение $x^{(1)}$ примем абсциссу точки пересечения этой касательной с осью Ox . Аналогично,

за приближение $x^{(2)}$ примем абсциссу точки пересечения с осью Ox касательной, проведенной к графику в точке $M^{(1)}$ с координатами $(x^{(1)}, f(x^{(1)}))$. Продолжая этот процесс далее, получим последовательность $x^{(0)}, x^{(1)}, x^{(2)}, \dots, x^{(n)}, \dots$ приближений к корню \bar{x} .

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

$$y = f(x^{(n)}) + f'(x^{(n)})(x - x^{(n)}). \quad (4.35)$$

Полагая в равенстве (4.35) $y = 0$, замечаем, что при выполнении условия $f'(x^{(n)}) \neq 0$ абсцисса $x^{(n+1)}$ точки пересечения касательной с осью Ox удовлетворяет равенству

$$0 = f(x^{(n)}) + f'(x^{(n)})(x^{(n+1)} - x^{(n)}). \quad (4.36)$$

Выражая из него $x^{(n+1)}$, получаем расчетную формулу *метода Ньютона*:

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})}, \quad n \geq 0. \quad (4.37)$$

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

Figure 2: Метод Ньютона

- **Функция:** `simplified_newton(f, df, x0, epsilon=1e-6)`
- **Описание параметров:**
 - `f` - Функция, корень которой нужно найти;
 - `df` - Производная функции;
 - `x` - Начальное приближение корня;
 - `epsilon` - Заданная точность (по умолчанию 10^{-6}).

```
from mathmod.nonlinear_equations import simplified_newton
```

```
x, iteration = simplified_newton(f, df, x0, epsilon=1e-6)
```

Расчетная формула:

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})}$$

Сходимость метода: скорость сходимости тем выше, чем ближе начальное приближение $x^{(0)}$ к решению \bar{x} .

Критерий окончания:

$$|x^{(n+1)} - x^{(n)}| < \varepsilon$$

3. Метод секущих

- Не требует аналитической производной функции.
- Использует приближённую производную.
- **Функция:** `secant(f, x_minus_1, x_n, epsilon=1e-6)`
- **Описание параметров:**
 - `f` - Функция, корень которой нужно найти;
 - `x_minus_1` - Первой начальное приближение;
 - `x` - Второе начальное приближение;
 - `epsilon` - Заданная точность (по умолчанию 10^{-6}).

```
from mathmod.nonlinear_equations import secant
```

```
x, iteration = secant(f, x_minus_1, x_n, epsilon=1e-6)
```

Расчетная формула:

$$x^{(n+1)} = x^{(n)} - \frac{x^{(n-1)} - x^{(n)}}{f(x^{(n-1)}) - f(x^{(n)})} f(x^{(n)})$$

Сходимость метода: $p = \frac{\sqrt{5}+1}{2} \approx 1.618$ - сверхлинейная, если вычисляется простой корень. При неудачном выборе приближения, метод расходится. Поэтому требуется выбор двух близких к \bar{x} начальных приближений $x^{(0)}$ и $x^{(1)}$.

Критерий окончания:

$$|x^{(n+1)} - x^{(n)}| < \varepsilon$$

1. Упрощенный метод Ньютона. Если производная $f'(x)$ непрерывна, то ее значение вблизи простого корня \bar{x} почти постоянно. Поэтому можно попытаться вычислить f' лишь однажды в точке $x^{(0)}$, а затем заменить в формуле (4.37) значение $f'(x^{(n)})$ постоянной $f'(x^{(0)})$. В результате получим расчетную формулу *упрощенного метода Ньютона*:

112

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(0)})}, \quad n \geq 0. \quad (4.46)$$

Геометрическая иллюстрация метода приведена на рис. 4.11. В точке $(x^{(0)}, f(x^{(0)}))$ к графику функции $y = f(x)$ проводится касательная l_0 и за приближение $x^{(1)}$ принимается абсцисса точки пересечения этой касательной с осью Ox (как в методе Ньютона). Каждое следующее приближение $x^{(n+1)}$ получается здесь как абсцисса точки пересечения с осью Ox прямой, проходящей через точку $M^{(n)}$ с координатами $(x^{(n)}, f(x^{(n)}))$ и параллельной касательной l_0 .

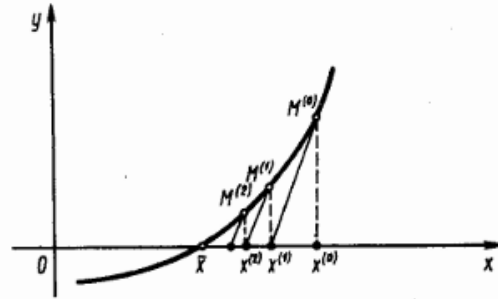


Рис. 4.11

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

Метод (4.46) можно рассматривать как метод простой итерации с итерационной функцией $\varphi(x) = x - \frac{f(x)}{f'(x^{(0)})}$. Так как $\varphi'(x) = 1 - \frac{f'(x)}{f'(x^{(0)})}$, то для знаменателя q соответствующей геометрической прогрессии имеем $q \approx \left| 1 - \frac{f'(\bar{x})}{f'(x^{(0)})} \right|$. Следовательно, скорость сходимости тем выше, чем ближе начальное приближение $x^{(0)}$ к решению \bar{x} .

Figure 3: Упрощенный метод Ньютона

чем выше, тем ближе окажется выбранная точка к x .

3. Метод секущих. Замена в формуле метода Ньютона производной $f'(x^{(n)})$ приближением $\frac{f(x^{(n-1)}) - f(x^{(n)})}{x^{(n-1)} - x^{(n)}}$ приводит к расчетной формуле метода секущих:

$$x^{(n+1)} = x^{(n)} - \frac{x^{(n-1)} - x^{(n)}}{f(x^{(n-1)}) - f(x^{(n)})} f(x^{(n)}), \quad n \geq 1. \quad (4.49)$$

Заметим, что этот метод двухшаговый, так как для нахождения очередного приближения $x^{(n+1)}$ требуется знание двух предыдущих приближений $x^{(n)}$ и $x^{(n-1)}$. В частности, для того чтобы начать вычисления, необходимо задать два начальных приближения $x^{(0)}$ и $x^{(1)}$. Все рассмотренные ранее методы требовали для вычисления $x^{(n+1)}$ только знание $x^{(n)}$, т. е. были одношаговыми.

114

На рис. 4.13 приведена геометрическая иллюстрация метода. Очередное приближение $x^{(n+1)}$ получается здесь как абсцисса точек пересечения с осью Ox секущей, соединяющей точки $M^{(n-1)}$ и $M^{(n)}$ графика функции $f(x)$ с координатами $(x^{(n-1)}, f(x^{(n-1)}))$ и $(x^{(n)}, f(x^{(n)}))$.

Примечательно то, что эта модификация метода Ньютона сохраняет свойство сверхлинейной сходимости, если вычисляется простой корень \bar{x} . Точнее, верно следующее утверждение.

Теорема 4.9. Пусть \bar{x} — простой корень уравнения $f(x) = 0$, в некоторой окрестности которого функция f дважды непрерывно дифференцируема, причем $f'(\bar{x}) \neq 0$. Тогда существует σ -окрестность корня \bar{x} такая, что при произвольном выборе приближений $x^{(0)}$ и $x^{(1)}$ из этой σ -окрестности метод секущих сходится с порядком $p =$

$$= \frac{\sqrt{5} + 1}{2} \approx 1.618, \text{ т. е. для } n \geq 1 \text{ справедлива оценка}$$

$$|x^{(n+1)} - \bar{x}| \leq c |x^{(n)} - \bar{x}|^p, \quad p = \frac{\sqrt{5} + 1}{2}.$$

Так как одна итерация метода секущих требует только одного нового вычисления значения функции f , а метод Ньютона — двух вычислений значений функций (f и f'), то трудоемкость двух итераций метода секущих приблизительно эквивалентна трудоемкости одной итерации по Ньютону. Две итерации метода секущих дают порядок $p^2 \approx 2.618 > 2$, поэтому его можно расценивать как более быстрый по сравнению с методом Ньютона.

К сожалению, метод обладает, вообще говоря, только локальной сходимостью. Он требует выбора двух близких к \bar{x} (в общем случае — очень близких) начальных приближений $x^{(0)}$ и $x^{(1)}$. Если эти приближения выбраны неудачно, то метод расходится (рис. 4.14).

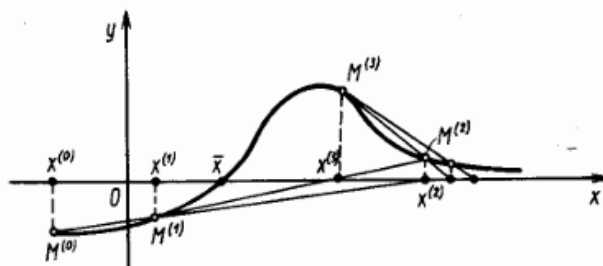


Рис. 4.14

115

4. Метод ложного положения

- **Функция:** `false_position(f, a, b, epsilon=1e-6)`
- **Описание параметров:**
- f : Функция, корень которой нужно найти;
- a : Левый конец начального отрезка локализации корня;
- b : Правый конец начального отрезка локализации корня;
- ϵ : Заданная точность (по умолчанию 10^{-6}).

```
from mathmod.nonlinear_equations import false_position
```

```
x, iteration = false_position(f, a, b, epsilon=1e-6)
```

Расчетная формула:

$$x^{(n+1)} = x^{(n)} - \frac{c - x^{(n)}}{f(c) - f(x^{(n)})} f(x^{(n)})$$

, где c — некоторая точка из окрестности корня.

Сходимость метода: линейная. Для достижения заданной точности требуется тем меньше итераций, чем ближе к корню лежит точка c .

Критерий окончания:

$$|x^{(n+1)} - x^{(n)}| < \epsilon$$

2. Метод ложного положения. В основе этой и следующих двух модификаций метода Ньютона лежит приближенное равенство

$$f'(x^{(n)}) \approx \frac{f(z^{(n)}) - f(x^{(n)})}{z^{(n)} - x^{(n)}}. \quad (4.47)$$

Оно верно при условии $z^{(n)} \approx x^{(n)}$ и следует из определения производной: $f'(x) = \lim_{z \rightarrow x} \frac{f(z) - f(x)}{z - x}$.

Пусть c — фиксированная точка, расположенная в окрестности простого корня \bar{x} . Заменим в расчетной формуле метода Ньютона (4.37) производную $f'(x^{(n)})$ правой частью приближенного равенства (4.47), полагая $z^{(n)} = c$. В результате придем к расчетной формуле *метода ложного положения*:

113

$$x^{(n+1)} = x^{(n)} - \frac{c - x^{(n)}}{f(c) - f(x^{(n)})} f(x^{(n)}), \quad n \geq 0. \quad (4.48)$$

Геометрическая иллюстрация метода приведена на рис. 4.12. Очередное приближение $x^{(n+1)}$ получается здесь как абсцисса точки пересечения с осью Ox прямой, проведенной через расположенные на графике функции $y = f(x)$ точки M и $M^{(n)}$ с координатами $(c, f(c))$ и $(x^{(n)}, f(x^{(n)}))$.

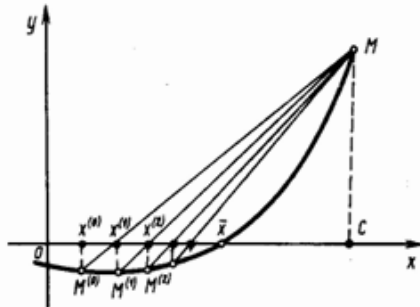


Рис. 4.12

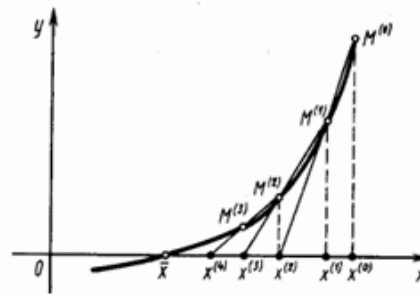


Рис. 4.13

Метод (4.48) обладает только линейной сходимостью. Его можно рассматривать как метод простой итерации с итерационной функцией

$$\varphi(x) = x - \frac{c - x}{f(c) - f(x)} f(x). \quad \text{Так как скорость сходимости определяется}$$

вблизи корня величиной $q \approx |\varphi'(\bar{x})| = \left| 1 - \frac{(c - \bar{x}) f'(\bar{x})}{f(c) - f(\bar{x})} \right|$, то она

тем выше, чем ближе окажется выбранная точка c к \bar{x} .

5. Метод бисекций

- Работает на основе деления отрезка, учитывая значения функции.
- Требуется, чтобы начальный отрезок $[a, b]$ удовлетворял условию $f(a) * f(b) < 0$. **Функция:** `bisection(f, a, b, epsilon)`

Описание параметров: - `f` - Функция, корень которой нужно найти; - `a` - Левый конец начального отрезка локализации корня; - `b` - Правый конец начального отрезка локализации корня; - `epsilon` - Заданная точность (по умолчанию 10^{-6}).

```
from mathmod.nonlinear_equations import bisection
```

```
x, iteration = bisection(f, a, b, epsilon=1e-6)
```

Расчетная формула:

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

Сходимость метода: со скоростью геометрической прогрессии, знаменатель которой $q = \frac{1}{2}$.

Критерий окончания:

$$b^{(n)} - a^{(n)} < 2\varepsilon$$

Тогда $x^{(n)} = \frac{a^{(n)} + b^{(n)}}{2}$ является искомым приближением к корню с точностью ε .

6. Метод простой итерации

Функция: `simple_iteration_method(phi, x0, epsilon=1e-6)`

Описание параметров: - `phi` - Итерационная функция $\phi(x)$, преобразующая исходное уравнение $f(x) = 0$ к виду $x = \phi(x)$; - `x0` - Начальное приближение корня; - `epsilon` - Заданная точность (по умолчанию 10^{-6}).

```
from mathmod.nonlinear_equations import simple_iteration_method
```

```
x, iteration = simple_iteration_method(phi, x0, epsilon=1e-6)
```

Расчетная формула:

$$x^{(n+1)} = \phi(x^{(n)})$$

Теорема о сходимости:

Если в окрестности корня функция $\phi(x)$ непрерывно дифференцируема и удовлетворяет условию:

$$\max_{x \in [a, b]} |\phi'(x)| \leq q$$

где $0 \leq q < 1$ - постоянная

Тогда независимо от выбора начального приближения $x^{(0)}$ из указанной окрестности корня итерационная последовательность не выходит из этой окрестности, метод сходится со скоростью геометрической прогрессии и справедлива следующая оценка погрешности (априорная оценка):

Априорная оценка - показывает, что итерационный метод сходится

$$|x^{(n)} - \bar{x}| \leq q^n |x^{(0)} - \bar{x}|$$

§ 4.3. Метод бисекции

1. **Описание метода.** Пусть требуется с заданной точностью $\epsilon > 0$ найти корень \bar{x} уравнения (4.1). Отрезок локализации $[a, b]$ (т. е. отрезок, содержащий только один корень \bar{x}) будем считать заданным. Предположим, что функция f непрерывна на отрезке $[a, b]$ и на его концах принимает значения разных знаков, т. е.

$$f(a)f(b) < 0. \quad (4.13)$$

На рис. 4.5 изображен случай, когда $f(a) < 0$ и $f(b) > 0$.

Для дальнейшего будет удобно обозначить отрезок $[a, b]$ через $[a^{(0)}, b^{(0)}]$. Примем за приближенное значение корня середину отрезка — точку $x^{(0)} = (a^{(0)} + b^{(0)})/2$. Так как положение корня \bar{x} на отрезке $[a^{(0)}, b^{(0)}]$ неизвестно, то можно лишь утверждать, что погрешность этого приближения не превышает половины длины отрезка (рис. 4.5):

$$|x^{(0)} - \bar{x}| \leq (b^{(0)} - a^{(0)})/2.$$

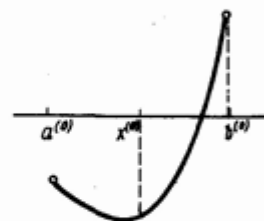


Рис. 4.5

Уменьшить погрешность приближения можно, уточняя отрезок локализации, т. е. заменяя начальный отрезок $[a^{(0)}, b^{(0)}]$ отрезком $[a^{(1)}, b^{(1)}]$ меньшей длины. Согласно *методу бисекции* (*половинного деления*) в качестве $[a^{(1)}, b^{(1)}]$ берут тот из отрезков $[a^{(0)}, x^{(0)}]$ и $[x^{(0)}, b^{(0)}]$, на концах которого выполняется условие $f(a^{(1)})f(b^{(1)}) \leq 0$. Этот отрезок содержит искомый корень. Действительно, если $f(a^{(1)})f(b^{(1)}) < 0$, то наличие корня следует из теоремы 4.1; если же $f(a^{(1)})f(b^{(1)}) = 0$, то корнем является один из концов отрезка. Середина полученного отрезка $x^{(1)} = (a^{(1)} + b^{(1)})/2$ дает теперь приближение к корню, оценка погрешности которого составляет

$$|x^{(1)} - \bar{x}| \leq (b^{(1)} - a^{(1)})/2 = (b - a)/2^2.$$

Figure 4: Метод бисекций

Чем меньше q , тем выше скорость сходимости.

Апостериорная оценка - критерий окончания итерационного процесса

$$|x^{(n)} - x^{(n-1)}| \leq \frac{1-q}{q} \epsilon$$

Если это условие выполнено, то можно считать, что $x^{(n)}$ является приближением к \bar{x} с точностью ϵ .

Решение систем линейных алгебраических уравнений (СЛАУ)

Постановка задачи

Система уравнений в общем виде:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \dots + a_{1m}x_m &= b_1, \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \dots + a_{2m}x_m &= b_2, \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + \dots + a_{3m}x_m &= b_3, \\ &\dots \\ a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + \dots + a_{mm}x_m &= b_m. \end{aligned}$$

В матричной форме эта система принимает вид:

$$Ax = b$$

, где

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1m} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2m} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3m} \\ \dots & \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & a_{m3} & \dots & a_{mm} \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_m \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_m \end{bmatrix}$$

m - порядок матрицы;

A - невырожденная матрица ($\Delta A \neq 0$, т.е. \exists единственное решение);

x - вектор неизвестных;

b - вектор свободных членов;

Пусть \bar{x} - точное решение, x^* - приближенное решение.

$$\bar{x} = \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \vdots \\ \bar{x}_m \end{bmatrix} \quad x^* = \begin{bmatrix} x_1^* \\ x_2^* \\ x_3^* \\ \vdots \\ x_m^* \end{bmatrix}$$

Тогда вектор $\epsilon = \bar{x} - x^*$ называется вектором погрешности.

Задача:

Найти решение системы $Ax = b$ с точностью ϵ . Это означает, что нужно найти вектор x^* такой, что $\|\bar{x} - x^*\| \leq \epsilon$, где $\|\cdot\|$ - одна из норм (единичная, евклидова, бесконечности).

§ 4.4. Метод простой итерации

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

93

$$x = \varphi(x). \quad (4.15)$$

Это преобразование (*приведение уравнения к виду, удобному для итерации*) можно выполнить различными способами; некоторые из них будут указаны ниже. Функцию φ далее будем называть *итерационной функцией*.

Выберем каким-либо образом приближенное значение корня $x^{(0)}$ и подставим его в правую часть уравнения (4.15). Получим значение $x^{(1)} = \varphi(x^{(0)})$. Подставляя теперь $x^{(1)}$ в правую часть уравнения (4.15), имеем $x^{(2)} = \varphi(x^{(1)})$. Продолжая этот процесс неограниченно, получим последовательность приближений к корню, вычисляемых по формуле

$$x^{(n+1)} = \varphi(x^{(n)}), \quad n \geq 0. \quad (4.16)$$

Очевидно, что метод простой итерации — одношаговый (см. § 4.1).

Если существует предел построенной последовательности $\bar{x} = \lim_{n \rightarrow \infty} x^{(n)}$, то, переходя к пределу в равенстве (4.16) и предполагая функцию φ непрерывной, получим равенство

$$\bar{x} = \varphi(\bar{x}). \quad (4.17)$$

Это значит, что \bar{x} — корень уравнения (4.15).

Figure 5: Метод простой итерации

Основные вопросы

Прямой метод - метод, который позволяет получить решение после выполнения конечного числа элементарных операций;

Итерационный метод - метод, который строит последовательность приближений к решению;

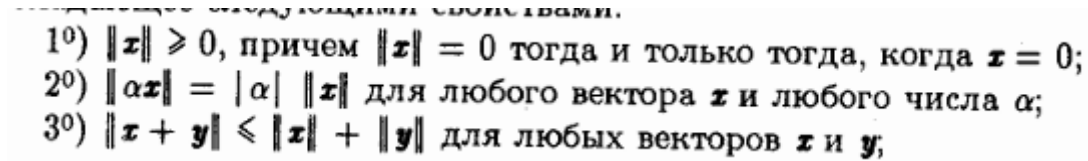
Норма - будем говорить, что в R^m введена норма, если каждому вектору $x \in R^m$ сопоставлено вещественное число, обозначаемое $\|x\|$.

1. Нормы векторов:

$$\|x\|_1 = \sum_{i=1}^m |x_i|, \quad \|x\|_2 = \left(\sum_{i=1}^m |x_i|^2 \right)^{1/2}, \quad \|x\|_\infty = \max_{1 \leq i \leq m} |x_i|. \quad (5.4)$$

Figure 6: Нормы векторов

Свойства норм векторов:



1⁰) $\|x\| \geq 0$, причем $\|x\| = 0$ тогда и только тогда, когда $x = 0$;
2⁰) $\|\alpha x\| = |\alpha| \|x\|$ для любого вектора x и любого числа α ;
3⁰) $\|x + y\| \leq \|x\| + \|y\|$ для любых векторов x и y ;

Figure 7: Свойства норм векторов

2. Нормы матриц:

Свойства норм матриц:

3. Абсолютная и относительная погрешность векторов и матриц:

- Погрешность векторов:

$\Delta x^* = \|\bar{x} - x^*\|$ - абсолютная погрешность;

$\delta x^* = \frac{\Delta x^*}{\|x^*\|}$ - относительная погрешность;

- 4. **Вектор невязки** - вектор, показывающий насколько найденное решение СЛАУ отклоняется от точного решения.

$$r = b - Ax^*$$

Число обусловленности - коэффициент возможного возрастания относительной погрешности решения, вызванное погрешностью задания правой части.

Пусть \bar{x} - точное решение системы $A\bar{x} = b$, а x^* - решение системы. Тогда верны следующие оценки:

$$\delta(x^*) \leq \nu_\delta(\delta(A^*) + \delta(b^*))$$

, где

$$\nu_\delta = \|A\| * \|A^{-1}\| \quad \delta(A^*) = \frac{\|A - A^{-1}\|}{\|A\|}$$

$$\|A\|_1 = \max_{1 \leq j \leq m} \sum_{i=1}^m |a_{ij}|, \quad (5.11)$$

$$\|A\|_2 = \max_{1 \leq j \leq m} \sqrt{\lambda_j(A^T A)}, \quad (5.12)$$

где $\lambda_j(A^T A)$ — собственные числа¹ матрицы $A^T A$;

$$\|A\|_\infty = \max_{1 \leq i \leq m} \sum_{j=1}^m |a_{ij}|. \quad (5.13)$$

Figure 8: Нормы матриц

- 1⁰) $\|A\| \geq 0$, причем $\|A\| = 0$ тогда и только тогда, когда $A = 0$.
 2⁰) $\|\alpha A\| = |\alpha| \cdot \|A\|$ для любой матрицы A и любого числа α .
 3⁰) $\|A + B\| \leq \|A\| + \|B\|$ для любых матриц A и B .
 Дополнительно к этому верны следующие свойства:
 4⁰) $\|A \cdot B\| \leq \|A\| \cdot \|B\|$ для любых матриц A и B ;
 5⁰) для любой матрицы A и любого вектора x справедливо неравенство

$$\|Ax\| \leq \|A\| \cdot \|x\|. \quad (5.10)$$

Figure 9: Свойства норм матриц

$\nu(A) = \text{cond}(A) = \|A^{-1}\| * \|A\|$ - стандартное число обусловленности.

Матрица *плохо* обусловлена, если $\text{cond}(A) \gg 1$. Следовательно, тогда существует решение, обладающее чрезвычайно высокой чувствительностью к малым погрешностям входного данного b .

Прямые методы

Прямой метод - метод, который позволяет получить решение после выполнения конечного числа элементарных операций.

1. Метод Гаусса (схема единственного деления)

- **Функция:** `gauss_single_division(A, b)`
- A - Матрица левой части;
- b - Вектор правой части;

Трудоемкость метода - $\frac{2}{3}m^3$

- Прямой ход - матрица A преобразуется к треугольному виду ($m - 1$ - шагов).
- Обратный ход - вычисляются значения неизвестных, начиная с последнего уравнения (m^2 - шагов).

Условие применимости - схема единственного деления не может быть реализована, если один из главных элементов равен нулю.

Описание метода:

1. Схема единственного деления. Рассмотрим сначала простейший вариант метода Гаусса, называемый *схемой единственного деления*.

Прямой ход состоит из $m - 1$ шагов исключения.

1-й шаг. Целью этого шага является исключение неизвестного x_1 из уравнений с номерами $i = 2, 3, \dots, m$. Предположим, что коэффициент $a_{i1} \neq 0$. Будем называть его *главным* (или *ведущим*) *элементом 1-го шага*.

Найдем величины

$$\mu_{i1} = a_{i1}/a_{11} \quad (i = 2, 3, \dots, m), \quad (5.29)$$

называемые *множителями 1-го шага*. Вычтем последовательно из второго, третьего, ..., m -го уравнений системы (5.1) первое уравнение, умноженное соответственно на $\mu_{21}, \mu_{31}, \dots, \mu_{m1}$. Это позволит обратить в

Figure 10: Метод Гаусса (схема единственного деления)

```
from mathmod.linear_systems import gauss_single_division  
  
x = gauss_single_division(A, b)
```

Пример:

и вычтем последовательно из $(k + 1)$ -го, ..., m -го уравнений полученной на предыдущем шаге системы k -е уравнение, умноженное соответственно на $\mu_{k+1,k}$, $\mu_{k+2,k}$, ..., μ_{mk} .

После $(m-1)$ -го шага исключения получим систему уравнений

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \dots + a_{1m}x_m &= b_1, \\ a_{22}^{(1)}x_2 + a_{23}^{(1)}x_3 + \dots + a_{2m}^{(1)}x_m &= b_2^{(1)}, \\ a_{33}^{(2)}x_3 + \dots + a_{3m}^{(2)}x_m &= b_3^{(2)}, \\ &\vdots \\ a_{mm}^{(m-1)}x_m &= b_m^{(m-1)}. \end{aligned} \quad (5.33)$$

матрица $A^{(m-1)}$ которой является верхней треугольной. На этом вычисления прямого хода заканчиваются.

Обратный ход. Из последнего уравнения системы (5.33) находим x_m . Подставляя найденное значение x_m в предпоследнее уравнение, получим x_{m-1} . Осуществляя обратную подстановку, далее последовательно находим x_{m-2} , x_{m-3} , ..., x_1 . Вычисления неизвестных здесь проводятся по формулам

$$\begin{aligned} x_m &= b_m^{(m-1)} / a_{mm}^{(m-1)}, \\ x_k &= (b_k^{(k-1)} - a_{k,k+1}^{(k-1)} x_{k+1} - \dots - a_{k,m}^{(k-1)} x_m) / a_{kk}^{(k-1)}, \quad (k = m-1, \dots, 1). \end{aligned} \quad (5.34)$$

Figure 12: Метод Гаусса (схема единственного деления)

$$\begin{aligned} 2x_1 + x_2 - x_3 &= 1, \\ 4x_1 + 3x_2 - x_3 &= 7, \\ 8x_1 + 7x_2 + 3x_3 &= 25. \end{aligned}$$

В матричной форме система записывается так:

$$\left[\begin{array}{ccc|c} 2 & 1 & -1 & 1 \\ 4 & 3 & -1 & 7 \\ 8 & 7 & 3 & 25 \end{array} \right]$$

Прямой ход

Шаг 1: Приведение первого столбца

Делим первую строку на ведущий элемент $a_{11} = 2$:

$$\left[\begin{array}{ccc|c} 1 & 0.5 & -0.5 & 0.5 \\ 4 & 3 & -1 & 7 \\ 8 & 7 & 3 & 25 \end{array} \right]$$

Обнуляем элементы ниже ведущего элемента в первом столбце, используя формулу:

$$a_{ij} \leftarrow a_{ij} - \mu_{ik} \cdot a_{kj}, \quad b_i \leftarrow b_i - \mu_{ik} \cdot b_k,$$

, где $\mu_{ik} = \frac{a_{ik}}{a_{kk}}$

Для второй строки:

$$\mu_{21} = 4, \quad a_{2j} \leftarrow a_{2j} - 4 \cdot a_{1j}$$

Для третьей строки:

$$\mu_{31} = 8, \quad a_{3j} \leftarrow a_{3j} - 8 \cdot a_{1j}$$

Получаем:

$$\left[\begin{array}{ccc|c} 1 & 0.5 & -0.5 & 0.5 \\ 0 & 1 & 1 & 5 \\ 0 & 3 & 7 & 21 \end{array} \right]$$

Шаг 2: Приведение второго столбца

Делим вторую строку на ведущий элемент $a_{22} = 1$ (он уже равен 1):

$$\left[\begin{array}{ccc|c} 1 & 0.5 & -0.5 & 0.5 \\ 0 & 1 & 1 & 5 \\ 0 & 3 & 7 & 21 \end{array} \right]$$

Обнуляем элементы ниже ведущего элемента во втором столбце:

$$\mu_{32} = 3, \quad a_{3j} \leftarrow a_{3j} - 3 \cdot a_{2j}$$

Получаем:

$$\left[\begin{array}{ccc|c} 1 & 0.5 & -0.5 & 0.5 \\ 0 & 1 & 1 & 5 \\ 0 & 0 & 4 & 6 \end{array} \right]$$

Обратный ход

Решаем треугольную систему методом подстановки.

Шаг 1: Найдём x_3 :

$$x_3 = \frac{6}{4} = 1.5$$

Шаг 2: Найдём x_2 :

$$x_2 = 5 - 1 \cdot x_3 = 5 - 1 \cdot 1.5 = 3.5$$

Шаг 3: Найдём x_1 :

$$x_1 = 0.5 - 0.5 \cdot x_2 - 0.5 \cdot x_3 = 0.5 - 0.5 \cdot 3.5 + 0.5 \cdot 1.5 = -0.5$$

Решение системы:

$$x_1 = -0.5, \quad x_2 = 3.5, \quad x_3 = 1.5$$

2. Метод Гаусса (схема частичного выбора)

- **Функция:** `gauss_partial_pivot(a, b)`

Трудоемкость метода - $\frac{2}{3}m^3$

Описание метода:

$$a_{ij}^{(k)} = a_{ij}^{(k-1)} - \mu_{ik} a_{kj}^{(k-1)}, \quad b_i^{(k)} = b_i^{(k-1)} - \mu_{ik} b_k^{(k-1)}, \quad i = k + 1, \dots, m. \quad (5.42)$$

Figure 13: Метод Гаусса (схема частичного деления)

Отличие от *схемы единственного деления* заключается в том, что на k -м шаге исключение в качестве главного элемента выбирают **максимальный** по модулю коэффициент $a_{i_k k}$.

Вычислительная устойчивость:

Гарантия ограниченности роста элементов матрицы делает схему частичного выбора вычислительно устойчивой. Становится справедлива оценка погрешности:

$$\delta(x^*) \lesssim f(m) \text{cond}_E(A) \epsilon_M$$

, где:

x^* - вычисленное ЭВМ решение системы. $\delta(x^*) = \frac{\|x - x^*\|_2}{\|x\|_2}$ - относительная погрешность;

$cond_E(A) = \|A\|_E \|A^{-1}\|_E$ - число обусловленности;

ϵ_M - машинный эпсилон;

$f(m) = C(m)\phi(m)$, где $C(m)$ - некоторая медленно растущая функция, зависящая от порядка системы;

$\phi(m)$ - коэффициент роста.

Далее исключение неизвестного x_k производят, как в схеме единственного деления.

```
from mathmod.linear_systems import gauss_partial_pivot
```

```
x = gauss_partial_pivot(A,b)
```

Пример:

Рассмотрим систему:

$$A = \begin{bmatrix} 2 & -9 & 5 \\ 0 & 3.5 & -10 \\ 0 & 0.0001 & 3 \end{bmatrix} \quad b = \begin{bmatrix} -4 \\ -6.5 \\ 3.0001 \end{bmatrix}$$

После вычисления $\mu_{32} = 0.0001/3.5 \approx 2.85714 \cdot 10^{-5}$ последнее уравнение системы преобразуется к виду $3.00029x_3 = 3.00029$.

Обратный ход. Из последнего уравнения находим $x_3 = 1$. Далее, имеем $x_2 = (-6.5 + 10x_3)/3.5 = 1$, $x_1 = (-4 + 9x_2 - 5x_3)/2 = (-4 + 9 - 5)/2 = 0$. В данном случае ответ получился точным.

Figure 14: Метод Гаусса (схема частичного деления)

3. Метод Холецкого (LLT-разложение)

- Для решения СЛАУ с симметрично положительно определённой матрицей.
- **Функция:** `cholesky(A, b)`

Трудоёмкость метода - $\frac{1}{3}m^3$

Условия применимости - требуется, чтобы диагональные элементы l_{ii} матрицы L были положительными.

Достоинства метода:

- Гарантированная устойчивость;
- Требует вдвое меньше вычислительных затрат по сравнению с методом Гаусса;
- Позволяет экономично использовать память ЭВМ при записи исходных данных и результатов вычислений за счёт симметричности матрицы A .

Описание метода:

A - симметрично положительно определённая матрица ($A = A^T$ и $\forall x \neq 0$ скалярное произведение $(Ax, x) > 0$).

L - нижнетреугольная матрица. L^T - транспонированная.

$$A = LL^T$$

, где

$$LL^T = \begin{bmatrix} l_{11} & 0 & 0 & \dots & 0 \\ l_{21} & l_{22} & 0 & \dots & 0 \\ l_{31} & l_{32} & l_{33} & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ l_{m1} & l_{m2} & l_{m3} & \dots & l_{mm} \end{bmatrix} \cdot \begin{bmatrix} l_{11} & l_{21} & l_{31} & \dots & l_{m1} \\ 0 & l_{22} & l_{23} & \dots & l_{m2} \\ 0 & 0 & l_{33} & \dots & l_{m3} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & l_{mm} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1m} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2m} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3m} \\ \dots & \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & a_{m3} & \dots & a_{mm} \end{bmatrix}$$

Отсюда:

$$l_{k1}^2 + l_{k2}^2 + \dots + l_{kk}^2 = a_{kk}$$

$$l_{kk} = \sqrt{a_{kk} - \sum_{j=1}^{k-1} l_{kj}^2} \quad k = 2 \dots m$$

$$l_{ik} = \frac{a_{i,k} - \sum_{j=1}^{k-1} l_{ij} \cdot l_{kj}}{l_{k,k}} \quad i = k+1, \dots m$$

Если разложение получено, то решение системы:

$$Ly = b \quad L^T x = y$$

```
from mathmod.linear_systems import cholesky
```

```
x = cholesky(A, b)
```

Пример:

Рассмотрим систему:

$$A = \begin{bmatrix} 6.25 & -1 & 0.5 \\ -1 & 5 & 2.12 \\ 0.5 & 2.12 & 3.6 \end{bmatrix} \quad b = \begin{bmatrix} 7.5 \\ -8.68 \\ -0.24 \end{bmatrix}$$

$$l_{11} = \sqrt{a_{11}} = \sqrt{6.25} = 2.5 \quad l_{21} = \frac{a_{21}}{l_{11}} = \frac{-1}{2.5} = -0.4$$

$$l_{31} = \frac{a_{31}}{l_{11}} = \frac{0.5}{2.5} = 0.2 \quad l_{22} = \sqrt{a_{22} - l_{21}^2} = \sqrt{5 - 0.16} = 2.2$$

$$l_{32} = a_{32} - l_{31}l_{21} = (2.12 - \frac{0.2 \cdot (-0.4)}{2.2}) = 1$$

$$l_{33} = \sqrt{a_{33} - l_{31}^2 - l_{32}^2} = \sqrt{3.6 - 0.2^2 - 1^2} = 1.6$$

Матрица L :

$$L = \begin{bmatrix} 2.25 & 0 & 0 \\ -0.4 & 2.2 & 0 \\ 0.2 & 1 & 1.6 \end{bmatrix}$$

Решение состоит из 2-х шагов:

1. Решаем $Ly = b$ для y методом прямой подстановки.
2. Решаем $L^T x = y$ для x методом обратной подстановки.

Решение:

1. Прямой ход для y :

$$Ly = b$$

$$\begin{bmatrix} 2.5 & 0 & 0 \\ -0.4 & 2.2 & 0 \\ 0.2 & 1 & 1.6 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 7.5 \\ -8.68 \\ -0.24 \end{bmatrix}$$

- Рассчитываем y :

$$\begin{aligned} 2.5 \cdot y_1 &= 7.5, \\ -0.4 \cdot y_1 + 2.2 \cdot y_2 &= -8.68, \\ 0.2 \cdot y_1 + y_2 + 1.6 \cdot y_3 &= -0.24 \end{aligned}$$

- Решая, получаем:

$$y_1 = 3, y_2 = -3.4, y_3 = 1.6$$

2. Обратный ход для x :

$$L^T x = y$$

$$\begin{bmatrix} 2.5 & -0.4 & 0.2 \\ 0 & 2.2 & 1 \\ 0 & 0 & 1.6 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ -3.4 \\ 1.6 \end{bmatrix}$$

- Рассчитываем x :

$$\begin{aligned} 2.5 \cdot x_1 - 0.4 \cdot x_2 + 0.2 \cdot x_3 &= 3, \\ 2.2 \cdot x_2 + x_3 &= -3.4, \\ 1.6 \cdot x_3 &= 1.6. \end{aligned}$$

- Решая, получаем:

$$x_1 = 0.8, \quad x_2 = -2, \quad x_3 = 1$$

4. Метод LU-разложения

- **Функция:** `lu(A, b)`

Трудоемкость метода - $\frac{2}{3}m^3 + m^2$

Теорема о возможности применения LU - разложения - если все главные миноры матрицы A отличны от нуля, то существуют единственная нижняя треугольная матрица L и верхняя треугольная матрица U такие, что:

$$A = LU$$

, где

$$L = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ \mu_{21} & 1 & 0 & \dots & 0 \\ \mu_{31} & \mu_{32} & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \mu_{m1} & \mu_{m2} & \mu_{m3} & \dots & 1 \end{bmatrix}$$

Описание метода:

§ 5.7. Метод Гаусса

и разложение матрицы на множители.

LU-разложение

Вернемся еще раз к методу Гаусса с тем, чтобы рассмотреть его с более общих позиций. Излагаемый ниже подход оказался чрезвычайно плодотворным и привел не только к более глубокому пониманию метода, но и позволил создать высокоэффективные машинные алгоритмы его реализации, а также рассматривать другие точные методы с единой точки зрения.

Рассмотрим сначала простейший вариант метода Гаусса для решения системы линейных алгебраических уравнений

$$Ax = b. \tag{5.51}$$

Figure 15: Метод LU - разложения

```
from mathmod.linear_systems import lu_solve
```

```
x = lu_solve(A, b)
```

Пример:

Рассмотрим систему:

$$A = \begin{bmatrix} 2 & -1 & -2 \\ -4 & 6 & 3 \\ -4 & -2 & 8 \end{bmatrix} \quad b = \begin{bmatrix} -5 \\ 6 \\ 8 \end{bmatrix}$$

Мы хотим представить её в виде произведения:

$$A = LU$$

, где:

$$L = \begin{bmatrix} 1 & 0 & 0 \\ \mu_{21} & 1 & 0 \\ \mu_{31} & \mu_{32} & 1 \end{bmatrix}, \quad U = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}.$$

1. Схема единственного деления и LU -разложение. При выполнении вычислений 1-го шага исключения по схеме единственного деления система уравнений приводится к виду

$$A^{(1)}x = b^{(1)}, \quad (5.52)$$

где

151

$$A^{(1)} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1m} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2m}^{(1)} \\ 0 & a_{32}^{(1)} & a_{33}^{(1)} & \dots & a_{3m}^{(1)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & a_{m2}^{(1)} & a_{m3}^{(1)} & \dots & a_{mm}^{(1)} \end{bmatrix}, \quad b^{(1)} = \begin{bmatrix} b_1 \\ b_2^{(1)} \\ b_3^{(1)} \\ \vdots \\ b_m^{(1)} \end{bmatrix},$$

а коэффициенты $a_{ij}^{(1)}$, $b_i^{(1)}$ ($i, j = 2, 3, \dots, m$) вычисляются по формулам (5.29), (5.31).

Введем матрицу

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ -\mu_{21} & 1 & 0 & \dots & 0 \\ -\mu_{31} & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ -\mu_{m1} & 0 & 0 & \dots & 1 \end{bmatrix}.$$

Как нетрудно проверить, справедливы равенства

$$A^{(1)} = M_1 A, \quad b^{(1)} = M_1 b,$$

т. е. преобразование системы (5.51) к виду (5.52) эквивалентно умножению левой и правой частей системы на матрицу M_1 .

Аналогично можно показать, что вычисления 2-го шага исключения приводят систему (5.52) к виду

$$A^{(2)}x = b^{(2)},$$

где

$$A^{(2)} = M_2 A^{(1)}, \quad b^{(2)} = M_2 b^{(1)},$$

Figure 16: Метод LU - разложения

$$A^{(2)} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1m} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2m}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \dots & a_{3m}^{(2)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & a_{m3}^{(2)} & \dots & a_{mm}^{(2)} \end{bmatrix}, M_2 = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & -\mu_{32} & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & -\mu_{m2} & 0 & \dots & 1 \end{bmatrix}, b^{(2)} = \begin{bmatrix} b_1 \\ b_2^{(1)} \\ b_3^{(2)} \\ \vdots \\ b_m^{(2)} \end{bmatrix}.$$

После $(m - 1)$ -го шага, завершающего прямой ход, система оказывается приведенной к виду

$$A^{(m-1)} x = b^{(m-1)} \quad (5.53)$$

с верхней треугольной матрицей $A^{(m-1)}$. Здесь
152

$$A^{(m-1)} = M_{m-1} A^{(m-2)}, \quad b^{(m-1)} = M_{m-1} b^{(m-2)},$$

$$A^{(m-1)} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1m} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2m}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \dots & a_{3m}^{(2)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & a_{mm}^{(m-1)} \end{bmatrix}, M_{m-1} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & -\mu_{m,m-1} & 1 \end{bmatrix},$$

$$b^{(m-1)} = \begin{bmatrix} b_1 \\ b_2^{(1)} \\ b_3^{(2)} \\ \vdots \\ b_m^{(m-1)} \end{bmatrix}.$$

Заметим, что матрица $A^{(m-1)}$ получена из матрицы A последовательным умножением на M_1, M_2, \dots, M_{m-1} :

$$A^{(m-1)} = M_{m-1} \dots M_2 M_1 A. \quad (5.54)$$

Figure 17: Метод LU - разложения

Аналогично,

$$b^{(m-1)} = M_{m-1} \dots M_2 M_1 b. \quad (5.55)$$

Из равенства (5.54) вытекает следующее представление:

$$A = M_1^{-1} M_2^{-1} \dots M_{m-1}^{-1} A^{(m-1)}. \quad (5.56)$$

Как легко проверить,

$$M_1^{-1} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ \mu_{21} & 1 & 0 & \dots & 0 \\ \mu_{31} & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \mu_{m1} & 0 & 0 & \dots & 1 \end{bmatrix}, \quad M_2^{-1} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & \mu_{32} & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \mu_{m2} & 0 & \dots & 1 \end{bmatrix}, \dots,$$

$$M_{m-1}^{-1} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & \mu_{m,m-1} & 1 \end{bmatrix}.$$

Для этого достаточно перемножить матрицы M_k^{-1} и M_k ($k = 1, \dots, m-1$), в результате чего получится единичная матрица.

153

Введем обозначения $U = A^{(m-1)}$, $L = M_1^{-1} M_2^{-1} \dots M_{m-1}^{-1}$. Вычисляя матрицу L , убеждаемся в том, что она имеет следующий вид:

$$L = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ \mu_{21} & 1 & 0 & \dots & 0 \\ \mu_{31} & \mu_{32} & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \mu_{m1} & \mu_{m2} & \mu_{m3} & \dots & 1 \end{bmatrix}. \quad (5.57)$$

Тогда равенство (5.56) в новых обозначениях примет вид

$$A = LU. \quad (5.58)$$

Figure 18: Метод LU - разложения

Шаг 1: Прямой ход (разложение)

1. Выбираем первый элемент матрицы A как ведущий:

$$u_{11} = 2$$

Элементы верхней матрицы U :

$$u_{12} = -1, \quad u_{13} = -2.$$

2. Вычисляем коэффициенты для матрицы L :

$$\mu_{21} = \frac{a_{21}}{u_{11}} = \frac{-4}{2} = -2, \quad \mu_{31} = \frac{a_{31}}{u_{11}} = \frac{-4}{2} = -2.$$

3. Обновляем элементы второй строки:

$$u_{22} = a_{22} - \mu_{21} \cdot u_{12} = 6 - (-2) \cdot (-1) = 4,$$

$$u_{23} = a_{23} - \mu_{21} \cdot u_{13} = 3 - (-2) \cdot (-2) = -1.$$

4. Обновляем элементы третьей строки:

$$u_{32} = a_{32} - \mu_{31} \cdot u_{12} = -2 - (-2) \cdot (-1) = -4$$

$$u_{33} = a_{33} - \mu_{31} \cdot u_{13} = 8 - (-2) \cdot (-2) = 4$$

5. Промежуточные результаты:

$$L = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -2 & \mu_{32} & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 2 & -1 & -2 \\ 0 & 4 & -1 \\ 0 & -4 & 4 \end{bmatrix},$$

6. Вычисляем μ_{32} :

$$\mu_{32} = \frac{u_{32}}{u_{22}} = -1$$

7. Обновляем элементы третьей строки:

$$u_{33} = u_{33} - \mu_{32} \cdot u_{23} = 4 - (-1) \cdot (-1) = 3$$

8. Итоговые матрицы

$$L = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -2 & -1 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 2 & -1 & -2 \\ 0 & 4 & -1 \\ 0 & 0 & 3 \end{bmatrix}.$$

Шаг 2: решение системы

Для решения системы $Ax = b$, где:

$$A = LU,$$

Решение состоит из 2-х шагов:

1. Решаем $Ly = b$ для y методом прямой подстановки.
2. Решаем $Ux = y$ для x методом обратной подстановки.

Решение:

1. Прямой ход для y :

$$Ly = b$$

$$\begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -2 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} -5 \\ 6 \\ 8 \end{bmatrix}$$

- Рассчитываем y :

$$\begin{aligned} y_1 &= -5, \\ -2 \cdot y_1 + y_2 &= 6, \\ -2 \cdot y_1 - 1 \cdot y_2 + y_3 &= 8. \end{aligned}$$

- Решая, получаем:

$$y_1 = -5, y_2 = -4, y_3 = -6$$

2. Обратный ход для x :

$$Ux = y$$

$$\begin{bmatrix} 2 & -1 & -2 \\ 0 & 4 & -1 \\ 0 & 0 & 3 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -5 \\ -4 \\ -6 \end{bmatrix}$$

- Рассчитываем x :

$$\begin{aligned} 2 \cdot x_1 - x_2 - 2 \cdot x_3 &= -5, \\ 4 \cdot x_2 - x_3 &= -4, \\ 3 \cdot x_3 &= -6. \end{aligned}$$

- Решая, получаем:

$$x_1 = -5.25, \quad x_2 = -1.5, \quad x_3 = -2$$

5. Метод прогонки

- Функция: `three_diag(A,b)`

Трудоемкость метода - $8m$

Условие применимости - коэффициенты системы удовлетворяют условиям диагонального преобладания:

$$|b_k| \geq |a_k| + |c_k| \quad |b_k| > |a_k|$$

Тогда:

$$\gamma_i = b_i + a_i \alpha_{i-1} \neq 0 \quad |\alpha_i| \leq 1 \quad \forall i = 1, 2, \dots, m$$

Описание метода:

A - терхдиагональная матрица:

$$A = \begin{bmatrix} b_1 & c_1 & 0 & \dots & \dots & 0 \\ a_2 & b_2 & c_2 & \dots & \dots & 0 \\ 0 & a_3 & b_3 & c_3 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & a_{m-1} & b_{m-1} & c_{m-1} \\ 0 & 0 & 0 & \dots & a_m & b_m \end{bmatrix} \quad b = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ b_{m-1} \\ b_m \end{bmatrix}$$

- **Прямой ход (прямая прогонка)** - вычисление прогоночных коэффициентов

$$\alpha_i = -\frac{c_i}{\gamma_i} \quad \beta_i = \frac{d_i - \alpha_i \beta_{i-1}}{\gamma_i} \quad \gamma_i = b_i + a_i \alpha_{i-1}$$

- **Обратная прогонка (обратная прогонка)** - вычисление значения неизвестных. Сначала $x_m = \beta_m$. Затем значения остальных неизвестных по формуле:

$$x_i = \alpha_i x_{i+1} + \beta_i \quad i = m-1, m-2, \dots, 1$$

```
from mathmod.linear_systems import three_diag
```

```
x = three_diag(A,b)
```

Пример:

Рассмотрим систему:

$$A = \begin{bmatrix} 5 & -1 & 0 & 0 \\ 2 & 4.6 & -1 & 0 \\ 0 & 2 & 3.6 & -0.8 \\ 0 & 0 & 3 & 4.4 \end{bmatrix} \quad b = \begin{bmatrix} 2 \\ 3.3 \\ 2.6 \\ 7.2 \end{bmatrix}$$

Прямой ход

$$\gamma_1 = b_1 = 5, \quad \alpha_1 = -c_1/\gamma_1 = 0.2, \quad \beta_1 = d_1/\gamma_1 = 2.0/5 = 0.4,$$

$$\gamma_2 = b_2 + a_2 \alpha_1 = 4.6 + 2.0 \cdot 0.2 = 5, \quad \alpha_2 = -c_2/\gamma_2 = 1/5 = 0.2,$$

$$\beta_2 = (d_2 - a_2 \beta_1)/\gamma_2 = (3.3 - 2.0 \cdot 0.4)/5 = 0.5,$$

$$\gamma_3 = b_3 + a_3 \alpha_2 = 3.6 + 2.0 \cdot 0.2 = 4, \quad \alpha_3 = -c_3/\gamma_3 = 0.8/4 = 0.2,$$

$$\beta_3 = (d_3 - a_3 \beta_2)/\gamma_3 = (2.6 - 2.0 \cdot 0.5)/4 = 0.4,$$

$$\gamma_4 = b_4 + a_4\alpha_3 = 4.4 + 3.0 \cdot 0.2 = 5, \quad \beta_4 = (d_4 - a_4\beta_3)/\gamma_4 = (7.2 - 3.0 \cdot 0.4)/5 = 1.2.$$

Обратный ход

$$x_4 = \beta_4 = 1.2,$$

$$x_3 = \alpha_3 z_4 + \beta_3 = 0.2 \cdot 1.2 + 0.4 = 0.64,$$

$$x_2 = \alpha_2 z_3 + \beta_2 = 0.2 \cdot 0.64 + 0.5 = 0.628,$$

$$x_1 = \alpha_1 z_2 + \beta_1 = 0.2 \cdot 0.628 + 0.4 = 0.5256.$$

Итак, получаем решение:

$$x_1 = 0.5256, \quad x_2 = 0.628, \quad x_3 = 0.64, \quad x_4 = 1.2.$$

Итерационные методы

Итерационный метод - метод, который строит последовательность приближений к решению;

1. Метод Якоби

- **Функция:** `jacobi(A, b, epsilon=1e-6, norma=1)`
 - A - Матрица коэффициентов (n x n);
 - b - Вектор правой части;
 - epsilon - Заданная точность (по умолчанию 10^{-6});
 - norma - Норма, по которой считается критерий окончания (например, 1, 2, np.inf).

Теорема о сходимости:

Пусть выполнено условие:

$$\|B\| < 1$$

Тогда решение системы \bar{x} существует и единственно при произвольном приближении $x^{(0)}$ МПИ сходится и справедлива оценка погрешности (*априорная оценка*):

$$\|x^{(n)} - \bar{x}\| \leq \|B\|^n \|x^{(0)} - \bar{x}\|$$

Апостериорная оценка:

$$\|x^{(n)} - \bar{x}\| \leq \frac{\|B\|}{1 - \|B\|} \|x^{(n)} - x^{(n-1)}\|$$

Критерий окончания:

$$\|x^{(n)} - x^{(n-1)}\| \leq \epsilon_1$$

, где:

$$\epsilon_1 = \frac{\|B\|}{1 - \|B\|} \epsilon$$

Более простой критерий окончания:

$$\|x^{(n)} - x^{(n-1)}\| \leq \epsilon$$

Описание метода:

$$\begin{aligned}x_1^{k+1} &= b_{11}x_1^k + b_{12}x_2^k + b_{13}x_3^k + \dots + b_{1m}x_m^k + c_1, \\x_2^{k+1} &= b_{21}x_1^k + b_{22}x_2^k + b_{23}x_3^k + \dots + b_{2m}x_m^k + c_2, \\x_3^{k+1} &= b_{31}x_1^k + b_{32}x_2^k + b_{33}x_3^k + \dots + b_{3m}x_m^k + c_3, \\&\dots \\x_m^{k+1} &= b_{m1}x_1^k + b_{m2}x_2^k + b_{m3}x_3^k + \dots + b_{mm}x_m^k + c_m,\end{aligned}$$

```
from mathmod.linear_systems import jacobi
```

```
x, iteration_count = jacobi(A, b, epsilon=1e-6, norma=1)
```

Пример:

2. Метод Гаусса-Зейделя

- Итерационный метод для решения СЛАУ с диагонально преобладающей матрицей.
- **Функция:** `gauss_zeidel(A, b, epsilon=1e-6, norma=1)`
 - A - Матрица коэффициентов (n x n);
 - b - Вектор правой части;
 - epsilon - Заданная точность (по умолчанию 10^{-6});
 - norma - Норма, по которой считается критерий окончания (например, 1, 2, np.inf).

3. Метод релаксации

- **Функция:** `relaxation_method(A, b, epsilon=1e-6, omega=1, norma=1)`
 - A - Матрица коэффициентов (n x n);
 - b - Вектор правой части;
 - epsilon - Заданная точность (по умолчанию 10^{-6});
 - omega - Параметр релаксации (по умолчанию 1.0 — метод Зейделя);
 - norma - Норма, по которой считается критерий окончания (например, 1, 2, np.inf).

§ 6.1. Метод простой итерации

1. Приведение системы к виду, удобному для итераций. Для того чтобы применить метод простой итерации к решению системы линейных алгебраических уравнений

$$Ax = b \quad (6.1)$$

с квадратной невырожденной матрицей A , необходимо предварительно преобразовать эту систему к виду

$$\mathbf{x} = B\mathbf{x} + \mathbf{c}. \quad (6.2)$$

Здесь B — квадратная матрица с элементами b_{ij} ($i, j = 1, 2, \dots, m$), c — вектор-столбец с элементами c_i ($i = 1, 2, \dots, m$).

В развернутой форме записи система (6.2) имеет следующий вид:

$$\begin{aligned} x_1 &= b_{11}x_1 + b_{12}x_2 + b_{13}x_3 + \dots + b_{1m}x_m + c_1, \\ x_2 &= b_{21}x_1 + b_{22}x_2 + b_{23}x_3 + \dots + b_{2m}x_m + c_2, \\ &\vdots \\ x_m &= b_{m1}x_1 + b_{m2}x_2 + b_{m3}x_3 + \dots + b_{mm}x_m + c_m, \end{aligned} \quad (6.3)$$

Вообще говоря, операция приведения системы к виду, удобному для итераций (т.е. к виду (6.2)), не является простой и требует специальных знаний, а также существенного использования специфики системы. В некоторых случаях в таком преобразовании нет необходимости, так как сама исходная система уже имеет вид (6.2).

Самый простой способ приведения системы к виду, удобному для итераций, состоит в следующем. Из первого уравнения системы (6.1) выразим неизвестное z_1 :

$$x_1 = a_{11}^{-1} (b_1 - a_{12}x_2 - a_{13}x_3 - \dots - a_{1m}x_m),$$

из второго уравнения — неизвестное x_2 :

$$x_2 = a_{22}^{-1} (b_2 - a_{21}x_1 - a_{23}x_3 - \dots - a_{2m}x_m),$$

и т.д. В результате получим систему

$$\begin{aligned} x_1 &= b_{12}x_2 + b_{13}x_3 + \dots + b_{1,m-1}x_{m-1} + b_{1m}x_m + c_1, \\ x_2 &= b_{21}x_1 + b_{23}x_3 + \dots + b_{2,m-1}x_{m-1} + b_{2m}x_m + c_2, \\ x_3 &= b_{31}x_1 + b_{32}x_2 + \dots + b_{3,m-1}x_{m-1} + b_{3m}x_m + c_3, \\ &\vdots \\ x_m &= b_{m1}x_1 + b_{m2}x_2 + b_{m3}x_3 + \dots + b_{m,m-1}x_{m-1} + c_m, \end{aligned} \quad (6.4)$$

Figure 19: Метод простых итераций

в которой на главной диагонали матрицы B находятся нулевые элементы. Остальные элементы выражаются по формулам

175

$$b_{ij} = -a_{ij}/a_{ii}, \quad c_i = b_i/a_{ii} \quad (i, j = 1, 2, \dots, m, j \neq i). \quad (6.5)$$

Конечно, для возможности выполнения указанного преобразования необходимо, чтобы диагональные элементы матрицы A были ненулевыми.

Часто систему (6.1) преобразуют к виду $\mathbf{x} = \mathbf{x} - \tau(A\mathbf{x} - \mathbf{b})$, где τ — специально выбираемый числовой параметр (см. п. 5).

2. Описание метода. Выберем начальное приближение $\mathbf{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \dots, x_m^{(0)})^T$. Подставляя его в правую часть системы (6.2) и вычисляя полученное выражение, находим первое приближение

$$\mathbf{x}^{(1)} = B\mathbf{x}^{(0)} + \mathbf{c}.$$

Подставляя приближение $\mathbf{x}^{(1)}$ в правую часть системы (6.2), получим

$$\mathbf{x}^{(2)} = B\mathbf{x}^{(1)} + \mathbf{c}.$$

Продолжая этот процесс далее, получим последовательность $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)}, \dots$ приближений, вычисляемых по формуле

$$\mathbf{x}^{(k+1)} = B\mathbf{x}^{(k)} + \mathbf{c}, \quad k = 0, 1, 2, \dots \quad (6.6)$$

В развернутой форме записи формула (6.6) выглядит так:

Figure 20: Метод простых итераций

Пример 6.1. Используя метод простой итерации в форме Якоби, найдем решение системы

$$\begin{aligned} 6.25x_1 - x_2 + 0.5x_3 &= 7.5, \\ -x_1 + 5x_2 + 2.12x_3 &= -8.68, \\ 0.5x_1 + 2.12x_2 + 3.6x_3 &= -0.24. \end{aligned} \quad (6.15)$$

с точностью $\epsilon = 10^{-3}$ в норме $\|\cdot\|_\infty$.

Вычисляя коэффициенты по формулам (6.5), приведем систему к виду (6.4)

$$\begin{aligned} x_1 &= 0.16x_2 - 0.08x_3 + 1.2, \\ x_2 &= 0.2x_1 - 0.424x_3 - 1.736, \\ x_3 &= -0.1389x_1 - 0.5889x_2 - 0.0667. \end{aligned} \quad (6.16)$$

В последнем уравнении коэффициенты даны с точностью до погрешности округления. Здесь

179

$$B = \begin{bmatrix} 0 & 0.16 & -0.08 \\ 0.2 & 0 & -0.424 \\ -0.1389 & -0.5889 & 0 \end{bmatrix}, \quad c = \begin{bmatrix} 1.2 \\ -1.736 \\ -0.0667 \end{bmatrix}.$$

Достаточное условие сходимости метода простой итерации выполнено, так как $\|B\|_\infty = \max \{0.24, 0.624, 0.7278\} = 0.7278 < 1$.

Figure 21: Метод простых итераций

§ 6.2. Метод Зейделя

1. **Описание метода.** Пусть система (6.1) приведена к виду (6.4) с коэффициентами, вычисленными по формулам (6.5).

Метод Зейделя¹ можно рассматривать как модификацию метода Якоби. Основная идея модификации состоит в том, что при вычислении очередного $(k+1)$ -го приближения к неизвестному x_i при $i > 1$ используют уже найденные $(k+1)$ -е приближения к неизвестным x_1, \dots, x_{i-1} , а не k -е приближения, как методе Якоби.

На $(k + 1)$ -й итерации компоненты приближения $\mathbf{z}^{(k+1)}$ вычисляются по формулам

$$\begin{aligned} x_1^{(k+1)} &= b_{12}x_2^{(k)} + b_{13}x_3^{(k)} + \dots + b_{1m}x_m^{(k)} + c_1, \\ x_2^{(k+1)} &= b_{21}x_1^{(k+1)} + b_{23}x_3^{(k)} + \dots + b_{2m}x_m^{(k)} + c_2, \\ x_3^{(k+1)} &= b_{31}x_1^{(k+1)} + b_{32}x_2^{(k+1)} + \dots + b_{3m}x_m^{(k)} + c_3, \\ &\vdots \\ x_m^{(k+1)} &= b_{m1}x_1^{(k+1)} + b_{m2}x_2^{(k+1)} + b_{m3}x_3^{(k+1)} + \dots + c_m. \end{aligned} \quad (6.23)$$

Введем нижнюю и верхнюю треугольные матрицы

$$B_1 = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ b_{21} & 0 & 0 & \dots & 0 \\ b_{31} & b_{32} & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & b_{m3} & \dots & 0 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0 & b_{12} & b_{13} & \dots & b_{1m} \\ 0 & 0 & b_{23} & \dots & b_{2m} \\ 0 & 0 & 0 & \dots & b_{3m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}.$$

Тогда расчетные формулы метода примут компактный вид:

Figure 22: Метод Зейделя

§ 6.3. Метод релаксации

Метод последовательной верхней релаксации является одним из наиболее эффективных и широко используемых итерационных методов для решения систем линейных алгебраических уравнений с симметричными положительно определенными матрицами A . Этот метод часто называют *SOR*-методом¹. Частично популярность *SOR*-метода можно объяснить его простотой и тем, что он хорошо известен широкому кругу прикладников.

Суть *метода релаксации* состоит в следующем. После вычисления очередной i -й компоненты $(k+1)$ -го приближения по формуле метода Зейделя

$$\begin{aligned} \tilde{x}_i^{(k+1)} = & b_{i1}x_1^{(k+1)} + b_{i2}x_2^{(k+1)} + \dots + b_{i,i-1}x_{i-1}^{(k+1)} + b_{i,i+1}x_{i+1}^{(k)} + \\ & + \dots + b_{im}x_m^{(k)} + c_i \end{aligned}$$

производят дополнительно смещение этой компоненты на величину $(\omega - 1)(\tilde{x}_i^{(k+1)} - x_i^{(k)})$, где ω — *параметр релаксации*. Таким образом, i -я компонента $(k+1)$ -го приближения вычисляется по формуле

$$x_i^{(k+1)} = \tilde{x}_i^{(k+1)} + (\omega - 1)(\tilde{x}_i^{(k+1)} - x_i^{(k)}) = \omega \tilde{x}_i^{(k+1)} + (1 - \omega)x_i^{(k)}.$$

На рис. 6.2 показано несколько первых итераций метода при значении параметра релаксации $\omega = 1.25$.

¹ От англ. successive over relaxation.

```
from mathmod.linear_systems import relaxation_method
```

```
x, iteration_count = relaxation_method(A, b, epsilon=1e-6, omega=1, norma=1)
```

В обозначении
вычисления

$$x^{(k+1)} = ($$

Как нетрудно
методом Зейделя
последовательной
довольственной
называют метод
значений ω .

Если A —
при любом ω
дится. Часто
чтобы *SOR*-
Зейделя. Од
задача. Во м

Существует
страненный
метров ω_i для
го приближе

Приближение функций

Постановка задачи Известны значения некоторой функции $f(x)$ только на множестве дискретных точек x_0, x_1, \dots, x_n , но само аналитическое выражение для функции неизвестно. Заменим функцию $f(x)$ некоторой известной и достаточно легко вычисляемой функцией $\Phi(x)$ такой, что $\Phi(x) \approx f(x)$. Подобный процесс замены неизвестной функции некоторой близкой функцией называется **аппроксимацией**, а функция $\Phi(x)$ называется **аппроксимирующей функцией**.

Для аппроксимации функций широко используются классы функций вида:

$$\Phi_m(x) = a_0\phi_0(x) + a_1\phi_1(x) + \dots + a_m\phi_m(x),$$

являющиеся линейными комбинациями фиксированного набора базисных функций $\phi_0(x), \phi_1(x), \dots, \phi_m(x)$.

Функцию $\phi_m(x)$ называют **обобщенным многочленом** по системе функций $\phi_0(x), \phi_1(x), \dots, \phi_m(x)$

Число m - **степенью многочлена**.

Существуют два основных подхода в аппроксимации функций:

1. Пусть точки $f(x_i), i = 0, 1, \dots, n$ получены в результате достаточно точных измерений или вычислений, т.е. есть основания считать их лишенными ошибок. Тогда следует выбирать аппроксимирующую функцию $\phi(x)$ такой, чтобы она совпадала со значениями исходной функции в заданных точках. Геометрически это означает, что кривая $\phi(x)$ проходит через точки $(x_i, f(x_i))$ плоскости. Такой метод приближения называется **интерполяцией**.
2. Если точки $f(x_i), i = 0, 1, \dots, n$ содержат ошибки (данные экспериментов, статистические данные и т.п.), то функция $\phi(x)$ выбирается из условия минимума некоторого функционала, обеспечивающего сглаживание ошибок. Такой прием называется **аппроксимацией** функции «в среднем». Геометрически это будет означать, что кривая $\phi(x)$ будет занимать некоторое «среднее» положение, не обязательно совпадая с исходными точками $(x_i, f(x_i))$ плоскости.

Постановка задачи интерполяции Пусть в точках x_0, x_1, \dots, x_n , расположенных на отрезке $[a, b]$ и попарно различных.

Тогда задача итерполяции состоит в построении функции $g(x)$, удовлетворяющей условию:

$$g(x) = y_i \quad (i = 0, 1, \dots, n)$$

Интерполяция - способ приближения функции $f(x)$ путем построения функции $g(x)$, график которой проходит через точки (x_i, y_i) .

Экстраполяция - способ приближения функции $f(x)$ в точке $x < x_{min}$ или $x > x_{max}$.

$[x_{min}, x_{max}]$ - минимальный и максимальный из узлов интерполяции.

Теорема о существовании и единственности интерполяционного многочлена:

Если $m = n$, то решение задачи интерполяции обобщенным многочленом ($\Phi_m(x) = a_0\phi_0(x) + a_1\phi_1(x) + \dots + a_m\phi_m(x)$) существует и единственно при любом наборе данных y_0, y_1, \dots, y_n , тогда и только тогда, когда системы функций $\phi_0(x), \phi_1(x), \dots, \phi_n(x)$ являются линейно независимыми в точках x_0, x_0, \dots, x_n .

Полиномиальная интерполяция. Многочлен Лагранжа

$$L_n(x) = \sum_{i=0}^n y_i l_{n,i}(x)$$

, где:

$$l_{ij}(x) = \prod_{\substack{k=1 \\ k \neq j}}^n \frac{(x - x_k)}{(x_j - x_k)} = \frac{(x - x_0)(x - x_1) \dots (x - x_{j-1})(x - x_{j+1}) \dots (x - x_n)}{(x_j - x_0)(x_j - x_1) \dots (x_j - x_{j-1})(x_j - x_{j+1}) \dots (x_j - x_n)}$$

Оценка погрешности:

$$\max_{[a,b]} |f(x) - P_n(x)| \leq \frac{M_{n+1}}{(n+1)!} \max_{[a,b]} |\omega_{n+1}(x)|$$

$$\omega_{n+1}(x) = (x - x_0)(x - x_1) \dots (x - x_n)$$

$$M_{n+1} = \max_{[a,b]} |f^{n+1}(x)|$$

или

$$\max_{[x_0, x_n]} |f(x) - P_n(x)| \leq \frac{M_{n+1}}{4(n+1)} h_{max}^{n+1}$$

$$h_{max} = \max_{1 \leq i \leq n} h_i$$

Эта формула позволяет утверждать, что для достаточно гладкой функции f при фиксированной степени интерполяционного многочлена погрешность интерполяции на отрезке $[x_0, x_n]$ при $h_{max} \rightarrow 0$ стремится к нулю не медленнее, чем некоторая величина, пропорциональная h_{max}^{n+1} .

Интерполяция многочленом n имеет $(n+1)$ -й порядок точности относительно h_{max} .

Пример:

Пусть даны точки: - $x_0 = 0, y_0 = 1$ - $x_1 = 1, y_1 = 2$ - $x_2 = 2, y_2 = 4$

Построим интерполяционный многочлен Лагранжа:

$$L_2(x) = y_0 \cdot \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} + y_1 \cdot \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} + y_2 \cdot \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}$$

Подставляя значения:

$$L_2(x) = 1 \cdot \frac{(x - 1)(x - 2)}{(0 - 1)(0 - 2)} + 2 \cdot \frac{(x - 0)(x - 2)}{(1 - 0)(1 - 2)} + 4 \cdot \frac{(x - 0)(x - 1)}{(2 - 0)(2 - 1)}$$

$$L_2(x) = 1 \cdot \frac{(x - 1)(x - 2)}{2} - 2 \cdot \frac{x(x - 2)}{1} + 4 \cdot \frac{x(x - 1)}{2}$$

Интерполяционный многочлен Ньютона с конечными разностями Пусть интерполируемая функция задана таблицей с постоянными шагом h , т.е. $x_i = x_0 + ih, i = 0, 1 \dots n$. Тогда введя безразмерную величину $t = \frac{x - x_0}{h}$, можно записать многочлен Ньютона:

$$P_n(x) = P_n(x + ht) = y_0 + \frac{\Delta y_0}{1!} t + \frac{\Delta^2 y_0}{2!} t(t-1) + \frac{\Delta^3 y_0}{3!} t(t-1)(t-2) + \dots + \frac{\Delta^n y_0}{n!} t(t-1)(t-2) \dots (t-n+1)$$

Метод наименьших квадратов Постановка задачи

Пусть даны точки x_0, x_1, \dots, x_n , и известны значения исходной функции $f_i = f(x_i), i = 0, 1, \dots, n$. Требуется найти многочлен P_m заданной степени $m (m = n)$ такой, чтобы величина среднеквадратического отклонения:

$$\sigma(P_m, f) = \sqrt{\frac{1}{1+n} \sum_{i=0}^n (P_m(x_i) - f_i)^2} = \sqrt{\frac{1}{1+n} \sum_{i=0}^n \left(\sum_{j=0}^n a_j x_i^j - f_i \right)^2}$$

была минимальной

Нормальная система:

$$\sum_{j=0}^m a_j \sum_{i=0}^n x_i^{k+j} = \sum_{i=0}^n f_i x_i^k \quad k = 0, 1, \dots, m$$

$$s_k = \sum_{i=0}^n x_i^k \quad b_k = \sum_{i=0}^n f_i x_i^k$$

Установка

Для использования библиотеки клонируйте репозиторий и установите необходимые зависимости: ““bash git clone <https://github.com/BaranovSerV/mathmod.git> cd mathmod pip install -r requirements.txt