Введение в численные методы. Краевые задачи. Задачи математической физики.

Баев А.Ж.

Казахстанский филиал МГУ

02 мая 2020

План на семестр

1. СЛАУ (точные методы)
2. СЛАУ (итерационные методы)
3. решение нелинейных уравнений
4. интерполяция
5. аппроксимация
6. интегрирование
7. **дифференцирование**

Краевая задача для дифференциального уравнения

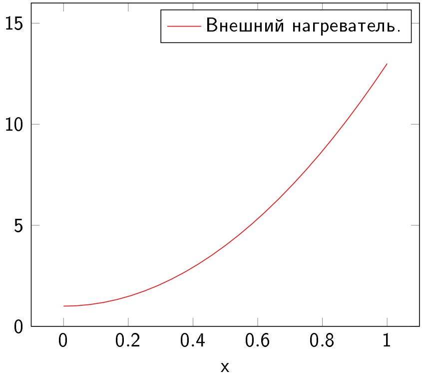
Дано $f(x)$, u_L и u_R. Определить $u(x)$.

$$\begin{cases} u''(x) + f(x) = 0 \\ u(0) = u_L \\ u(1) = u_R \end{cases}$$

Краевая задача для дифференциального уравнения

Пример. Дан однородный стержень длины 1. $f(x) = 1 + 12x^2$ — нагреватель. Температуры на концах равны $u_L = 1$, $u_R = 1$.

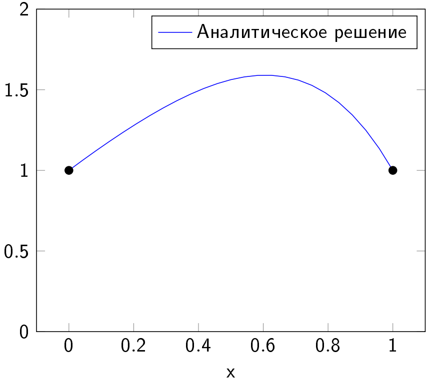
$$\begin{cases} u''(x) + (1 + 12x^2) = 0 \\ u(0) = 1 \\ u(1) = 1 \end{cases}$$



Краевая задача для дифференциального уравнения

Найдём аналитическое решение

$$\begin{cases} u(x) = -x^4 - \frac{x^2}{2} + C_1 x + C_2 \\ C_2 = 1 \\ -1 - \frac{1}{2} + C_1 + C_2 = 1 \end{cases} \Leftrightarrow u(x) = -x^4 - \frac{x^2}{2} + \frac{3}{2}x + 1$$



Краевая задача для дифференциального уравнения

Введём равномерную сетку $x_i = i \cdot h$, где $h = \frac{1}{n}$ и $i = 0 \dots n$.
Апрроксимация $y_i = u(x_i)$.

$$\begin{cases} \frac{y_{i-1} - 2y_i + y_{i+1}}{h^2} + f_i = 0, \\ y_0 = U_L, \\ y_n = U_R, \end{cases} \Leftrightarrow \begin{cases} -y_{i-1} + 2y_i - y_{i+1} = h^2 f_i, \\ y_0 = U_L, \\ y_n = U_R, \end{cases}$$

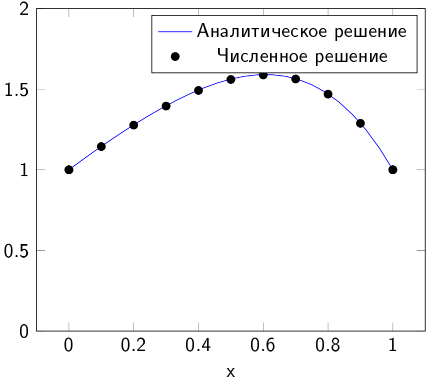
Решение сводится к СЛАУ с трёхдиагональной матрицей:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & \dots & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & \dots & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & \dots & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ \dots \\ y_{n-2} \\ y_{n-1} \\ y_n \end{pmatrix} = \begin{pmatrix} U_L \\ h^2 f_1 \\ h^2 f_2 \\ h^2 f_3 \\ \dots \\ h^2 f_{n-2} \\ h^2 f_{n-1} \\ U_R \end{pmatrix}$$

Краевая задача для дифференциального уравнения

```
def f(x):
    return 1 + 12 * x * x

n = 10

u = np.array([0] + [0] + [-1] * (n-1))
d = np.array([1] + [2] * (n - 1) + [1])
l = np.array([-1] * (n-1) + [0] + [0])
A = np.array([u, d, l])

h = 1.0 / n
x = np.linspace(0, 1, n + 1)
b = h * h * np.vectorize(f)(x)
b[0] = 1
b[n] = 1

y = sl.solve_banded((1, 1), np.array([u, d, l]), b)
```

Краевая задача для дифференциального уравнения

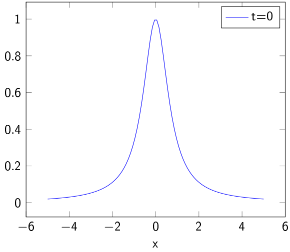


Уравнение переноса

Дана функция начального профиля $u_0(x)$ и скорость переноса C. Найти положение профиля $u(x, T)$ в заранее фиксированный момент времени T.

$$\begin{cases} \frac{\partial u(x,t)}{\partial t} + C\frac{\partial u(x,t)}{\partial x} = 0, x \in R, t > 0 \\ u(x, 0) = u_0(x), x > 0 \\ u(0, t) = v_0(t), t > 0 \end{cases}$$

Например, $u_0(x) = \frac{1}{2x^2+1}$.



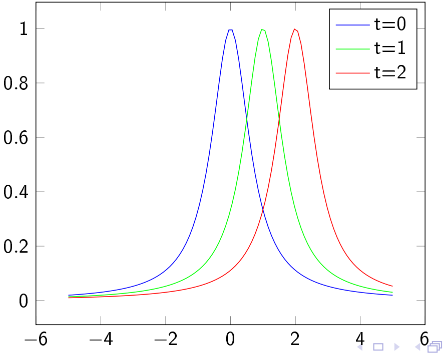
Уравнение переноса

$$\frac{\partial u(x,t)}{\partial t} + C\frac{\partial u(x,t)}{\partial x} = 0$$

Аналитическое решение

$$u(x,t) = u_0(x - Ct)$$

Например, $u_0(x) = \frac{1}{2x^2+1}$, $v_0(t) = \frac{1}{2t^2+1}$, $C = 1$, то есть $u(x,t) = \frac{1}{2(x-t)^2+1}$.



Уравнение переноса

Построим численное решение на двумерной равномерной сетке $n \times m$:

$$x_i = L + i \cdot h,$$

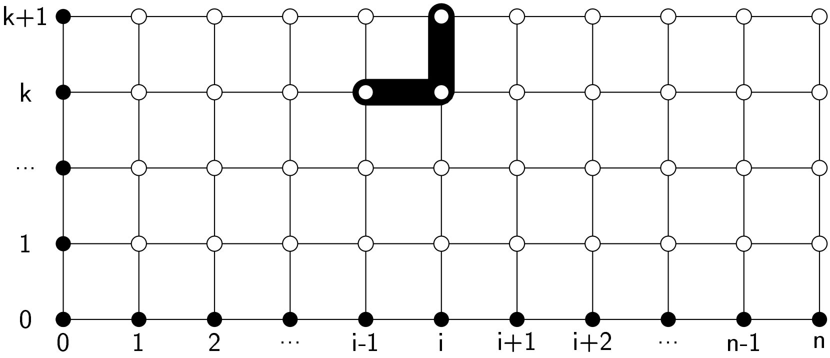
$$t^k = k \cdot \tau,$$

где $h = \frac{R-L}{n}$, $\tau = \frac{T}{m}$.
Аппроксимация решения $y_i^k \approx u(x_i, t^k)$.

$$\begin{cases} \frac{\partial u(x,t)}{\partial t} + C\frac{\partial u(x,t)}{\partial x} = 0, x > 0, t > 0 \\ u(x,0) = u_0(x), x \in [L, R] \\ u(0,t) = v_0(t), t > 0 \end{cases}$$

$$\begin{cases} \frac{y_i^{k+1} - y_i^k}{\tau} + C\frac{y_i^k - y_{i-1}^k}{h} = 0, i = 0..n-1, k = 0..m-1 \\ y_i^0 = u_0(x_i), i = 0..n \\ y_0^k = v_0(t^k), k = 0..m \end{cases}$$

Уравнение переноса



Уравнение переноса

Параметры задачи:

```
def u0(x):
    return 1.0 / (1 + 2 * x * x)
C = 1.0
T = 2.0
L, R = -5.0, 5.0
```

Параметры метода:

```
n = 40
m = 40
h = (R - L) / n
tau = T / m
```

Сетки:

```
x = np.linspace(L, R, n + 1)
t = np.linspace(0.0, T, m + 1)
y = np.zeros((m + 1, n + 1))
```

Уравнение переноса

$$\begin{cases} y_i^{k+1} = y_i^k - \frac{C\tau}{h}(y_{i+1}^k - y_i^k), i = 0..n-1 \\ y_i^0 = u_0(x_i) \end{cases}$$

Метод:

```
d = C * tau / h
y[0] = np.vectorize(u0)(x)
for k in range(m):
    for i in range(1, n + 1):
        y[k+1][i] = y[k][i] - d * (y[k][i] - y[k][i-1
```

Точное решение

```
def solution(x, t):
    return f(x - C * t)

vsolution = np.vectorize(solution, excluded=['t'])
u = np.zeros((m + 1, n + 1))
for k in range(m):
    u[k] = vsolution(x, tau * k)
```

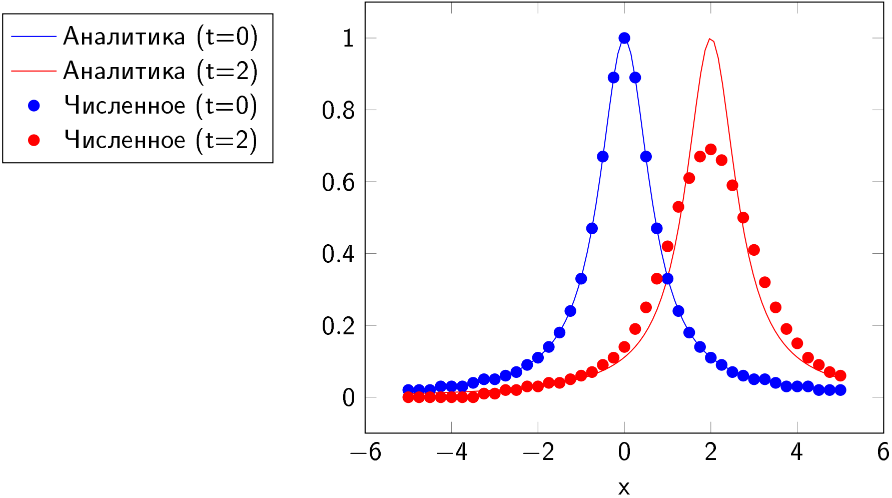
Уравнение переноса

Анимация

```python
import matplotlib.pyplot as plt
import matplotlib.animation as animation

def animate(k):
    plt.clf()
    plt.ylim(0, 1)
    plt.title(time = " + str(tau * k))
    plt.plot(x, y[k], marker='o')
    plt.legend("Numerical")
    plt.plot(x, u[k], marker='*')
    plt.legend("Analytical")

ani = animation.FuncAnimation(plt.figure(0), animate,
                              frames=y.shape[0],
                              interval=100)
# ani.save('transfer.mp4')
plt.show()
```

Уравнение переноса

Например, $u_0(x) = \frac{1}{2x^2+1}$, $C = 1$, то есть $u(x,t) = \frac{1}{2(x-t)^2+1}$.



Погрешность и устойчивость

Погрешность аппроксимации (оператора):

$$O(\tau + h)$$

Условие устойчивости:

$$\tau < \frac{h}{2C}$$

Погрешность решения:

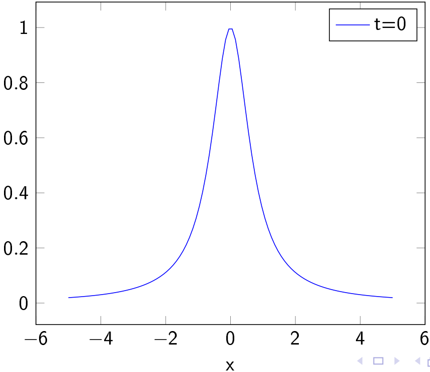
$$O(\tau + h)$$

Уравнение теплопроводности

Дана функция начального профиля $u_0(x)$ и коэффициент теплопроводности μ. Найти профиль в произвольный момент времени T.

$$\begin{cases} \frac{\partial u(x,t)}{\partial t} - \mu \frac{\partial^2 u(x,t)}{\partial x^2} = 0, x \in R, t > 0 \\ u(x,0) = u_0(x), x \in R \end{cases}$$

Например, $u_0(x) = \frac{1}{2x^2+1}$.



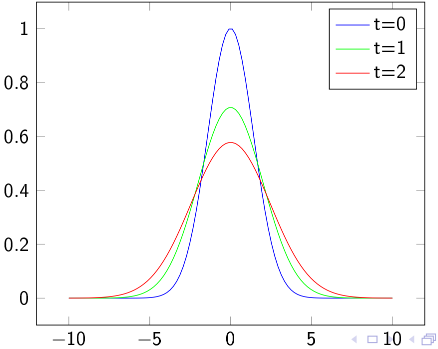
Уравнение теплопроводности

$$\frac{\partial u(x,t)}{\partial t} - \mu \frac{\partial^2 u(x,t)}{\partial x^2} = 0$$

Аналитическое решение

$$u(x,t) = \frac{1}{2\sqrt{\mu \pi t}} \int_R e^{-\frac{(x-y)^2}{4\mu t}} u_0(y) dy$$

Например, $u_0(x) = e^{-\frac{x^2}{4}}$, $\mu = 1$, то есть $u(x,t) = \frac{1}{\sqrt{t+1}} e^{-\frac{x^2}{4(t+1)}}$.



Уравнение теплопроводности

Построим численное решение на двумерной равномерной сетке $n \times m$:

$$x_i = L + i \cdot h,$$

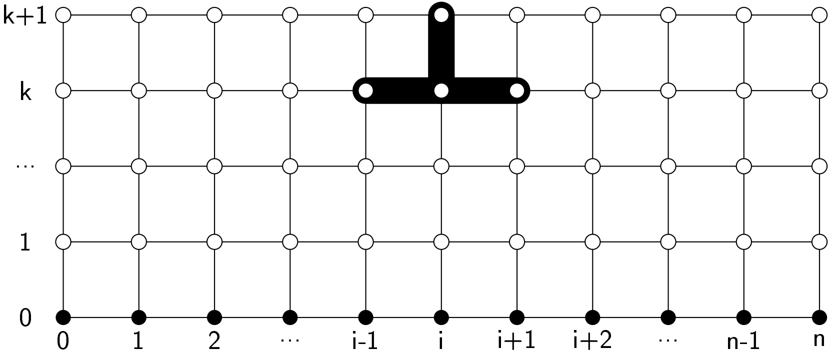
$$t^k = k \cdot \tau,$$

где $h = \frac{R-L}{n}$, $\tau = \frac{T}{m}$.
Аппроксимация решения $y_i^k \approx u(x_i, t^k)$.

$$\begin{cases} \frac{\partial u(x,t)}{\partial t} - \mu \frac{\partial^2 u(x,t)}{\partial x^2} = 0, x \in R, t > 0 \\ u(x,0) = u_0(x), x \in [L,R] \end{cases}$$

$$\begin{cases} \frac{y_i^{k+1} - y_i^k}{\tau} - \mu \frac{y_{i+1}^k - 2y_i^k + y_{i-1}^k}{h} = 0, i = 1..n-1 \\ u_i^k = u_0(x_i) \end{cases}$$

Уравнение теплопроводности



Уравнение теплопроводности

Параметры задачи:

```
def u0(x):
    return np.exp(-x*x/4)

mu = 1.0
T = 2.0
L, R = -10.0, 10.0
```

Параметры метода:

```
n = 40
m = 40
h = (R - L) / n
tau = T / m
```

Сетки:

```
x = np.linspace(L, R, n + 1)
t = np.linspace(0.0, T, m + 1)
y = np.zeros((m + 1, n + 1))
```

Уравнение теплопроводности

$$\begin{cases} y_i^{k+1} = y_i^k + \frac{\mu\tau}{h^2}(y_{i+1}^k - 2y_i^k + y_{i-1}^k), i = 1..n-1 \\ y_i^0 = u_0(x_i) \end{cases}$$

Метод:

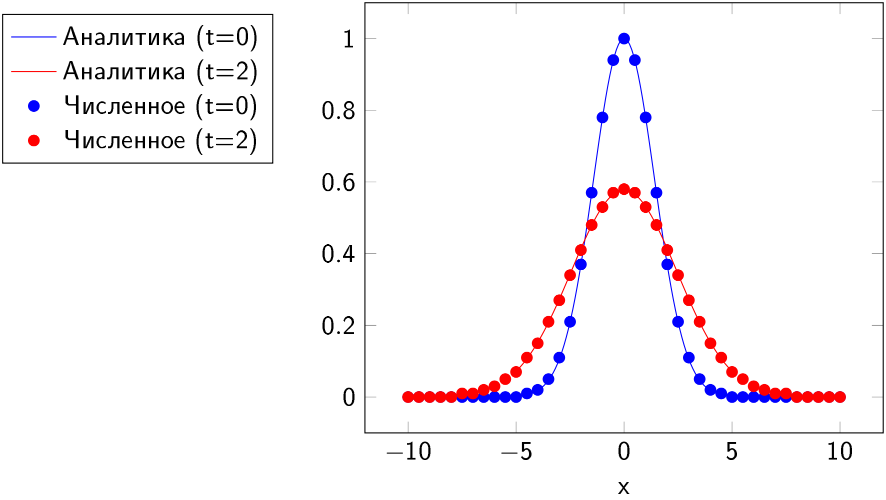
```
d = mu * tau / (h * h)
y[0] = np.vectorize(u0)(x)
for k in range(m):
    for i in range(1, n):
        y[k+1][i] = y[k][i] + d * (y[k][i-1] - 2 * y[
```

Точное решение

```
def solution(x, t):
    return 1 / np.sqrt(t + 1) * np.exp(- x * x / 4 /

vsolution = np.vectorize(solution, excluded=['t'])
u = np.zeros((m + 1, n + 1))
for k in range(m):
    u[k] = vsolution(x, tau * k)
```

Уравнение теплопроводности

Например, $u_0(x) = e^{-\frac{x^2}{4}}$, $\mu = 1$, то есть $u(x,t) = \frac{1}{\sqrt{t+1}} e^{-\frac{x^2}{4(t+1)}}$.



Погрешность и устойчивость

Погрешность аппроксимации (оператора):

$$O(\tau + h^2)$$

Условие устойчивости:

$$\tau < \frac{h^2}{2\mu}$$

Погрешность решения:

$$O(\tau + h^2)$$