МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет прикладной математики и информатики

Кафедра математического моделирования и анализа данных

Лабораторная работа 2

Численные методы решения задачи Коши для уравнения колебаний

Вариант К-3

Выполнил
Цуранов Никита Васильевич
3 курс, 7 группа

Преподаватель
Радкевич Елена Владимировна

Минск, 2020

Содержание

Постановка задачи **3**

Построение разностной схемы **3**

Исследование устойчивости **4**

Реализация построенной схемы **5**

Вывод программы **5**

Листинг программы: **6**

Выводы **7**

Лист оценивания **7**

Постановка задачи

$$\begin{cases} \frac{\partial^2 u}{\partial t^2} - \frac{\partial^2 u}{\partial x^2} = (x^2 - t^2)e^{-xt}, 0 \le x, t \le 1, \\ u(x,0) = 1, \\ \frac{\partial u(x,0)}{\partial t} = -x \\ u(0,t) = 1, \\ u(1,t) = e^{-t} \end{cases}$$

Необходимо:

1. На сетке $\omega_{h,\tau}$ построить разностную схему с весами при $\sigma = \frac{1}{4} - \frac{h^2}{12\tau^2}$ с погрешностью аппроксимации не ниже $O(h^4 + \tau^2)$
2. Показать, что построенная схема имеет заданный порядок аппроксимации
3. Исследовать устойчивость полученной разностной схемы по начальным данным, используя принцип максимума
4. Реализовать разностную схему при $h = 0.05$ и τ, выбранное из условия устойчивости
5. Оценить приближенное решение, анализируя погрешность аппроксимации при разных шагах

Построение разностной схемы

В области $\{0 \le x \le 1, 0 \le t \le 1\}$ введем сетку $\omega_{h\tau} = \omega_h \times \omega_\tau$. Для аппроксимации исходного уравнения используем девятиточечный шаблон.

$$\begin{cases} y_{\bar{t}t}(x,t) = \Lambda\big(\sigma\hat{y} + (1-2\sigma)y + \sigma\check{y}\big) + \varphi, (x,t) \in \omega_{h\tau}, \\ \left.\begin{matrix} y(x,0) = 1, \\ y(x,0)_t = -x \end{matrix}\right. \quad x \in \overline{\omega_h} \\ \left.\begin{matrix} y(0,t) = 1, \\ y(1,t) = e^{-t} \end{matrix}\right. \quad t \in \overline{\omega_t} \end{cases}$$

Тогда из 13 лекции, пункта 3.2 имеем, что при $\sigma = \frac{1}{4} - \frac{h^2}{12t^2}$ и $\varphi = f + \frac{h^2}{12}\frac{\partial^2 f}{\partial x^2}$, то погрешность аппроксимации будет $O(h^4 + \tau^2)$.

Тогда $\varphi = e^{-tx}(x^2 - t^2 + \frac{(t^4 + (2-tx)^2 - 2)h^2}{12})$.

Рассмотрим краевое условие $u(x,0)_t = \mu_0(x) = -x$. Для достижения нужного порядка необходимо построить:

$$\tilde{\mu}_0(x) = \mu_0(x) + \frac{\tau}{2}f(x,0) + \frac{\tau^2}{6}\dot{f}(x,0) + \frac{\tau^8}{24}\ddot{f}(x,0) + \frac{\tau^2}{12}\mu_0''(x) = -x + \frac{\tau}{2}x^2 - \frac{\tau^2}{6}x^3 + \frac{\tau^8}{24}(x^4 - 2)$$

Из того, что $\hat{y} + \check{y} = 2y + \tau^2 y_{\bar{t}t}$ следует, что $\Lambda y + \sigma\tau^2\Lambda y_{\bar{t}t} - y_{\bar{t}t} = -\varphi$

$$\Lambda y_{\bar{t}t} = \frac{\hat{y}_{\bar{x}x} - 2y_{\bar{x}x} + \check{y}_{\bar{x}x}}{\tau^2} \Rightarrow y_{\bar{x}x} + \sigma(\hat{y}_{\bar{x}x} - 2y_{\bar{x}x} + \check{y}_{\bar{x}x}) - y_{\bar{t}t} = -\varphi$$

В индексном виде:

$$\sigma\frac{y_{i-1}^{j-1}-2y_i^{j-1}+y_{i+1}^{j-1}}{h^2}+(1-2\sigma)\frac{y_{i-1}^{j}-2y_i^{j}+y_{i+1}^{j}}{h^2}+$$
$$+\sigma\frac{y_{i-1}^{j+1}-2y_i^{j+1}+y_{i+1}^{j+1}}{h^2}-\frac{y_i^{j-1}-2y_i^{j}+y_i^{j+1}}{\tau^2}=\varphi_i$$

Запишем коэффициенты перед $y_{i\pm1}^{j\pm1}$ умноженные на $h^2\tau^2$:

y	$i-1$	i	$i+1$
$j+1$	$\sigma\tau^2$	$-h^2-2\sigma\tau^2$	$\sigma\tau^2$
j	$(1-2\sigma)\tau^2$	$2h^2-2(1-2\sigma)\tau^2$	$(1-2\sigma)\tau^2$
$j-1$	$\sigma\tau^2$	$-h^2-2\sigma\tau^2$	$\sigma\tau^2$

Тогда система примет вид:

$$\begin{cases}\sigma\tau^2(y_{i-1}^{j-1}+y_{i-1}^{j+1}+y_{i+1}^{j-1}+y_{i+1}^{j+1})-(h^2+2\sigma\tau^2)(y_i^{j-1}+y_i^{j+1})+\\ \quad+(1-2\sigma)\tau^2(y_{i-1}^{j}+y_{i+1}^{j})+(2h^2-2(1-2\sigma)\tau^2)y_i^j=\varphi h^2\tau^2,(x,t)\in\omega_{h\tau},\\ y_i^0=1, \\ y_i^1=\tilde{\mu}_{0,i}\tau+1 \quad i=\overline{0,N}\\ y_0^j=1, \\ y_N^j=e^{-t_j} \quad j=\overline{0,N}\end{cases}$$

Исследование устойчивости

Пусть c_m^k — коэффициент перед y_m^k, тогда рассмотрим принцип максимума относительно y_i^j, перенеся y_i^j в одну сторону, остальные в другую:

$$\begin{cases}\varphi\geq 0,\\ c_k^m\geq 0, i-1\leq k\leq i+1, j-1\leq m\leq j+1,\\ c_i^j\geq c_{i-1}^{j-1}+c_{i-1}^{j}+c_{i-1}^{j+1}+c_i^{j-1}+c_i^{j+1}+c_{i+1}^{j-1}+c_{i+1}^{j}+c_{i+1}^{j+1}\end{cases}$$

Из второго условия следует, что

$$\begin{cases}-\sigma\tau^2\geq 0,\\ (2\sigma-1)\tau^2\geq 0,\\ h^2+2\sigma\tau^2\geq 0,\\ 2h^2-2(1-2\sigma)\tau^2\geq 0.\end{cases}$$

Можно заметить, что $\sigma\leq 0.25\Rightarrow(2\sigma-1)\leq 0$, а значит по принципу максимума схема не устойчива.

Но из 13 лекции и неравенства 14 можем сделать вывод, что схема устойчива для любых h и τ.

Реализация построенной схемы

Заполняем начальную сетку из краевых условий построенной схемы. Для нахождения следующих узлов нам понадобится решить $N_h - 2$ систем линейных уравнений.

Для этого выразим следующий временной уровень:

$$
\begin{gathered}
y_0^{j+1} = 1, \\
\tfrac{\sigma}{h^2} y_{i-1}^{j+1} - (\tfrac{2\sigma}{h^2} + \tfrac{1}{\tau^2}) y_{i-1}^{j+1} + \tfrac{\sigma}{h^2} y_{i-1}^{j+1} = -b_i, i = \overline{1, N-1}, \qquad j = \overline{1, N-1} \\
y_N^{j+1} = e^{-t_j}.
\end{gathered}
$$

$$
\text{Где } b_i = \frac{2y_i^j - y_i^{j+1}}{\tau^2} + (1 - 2\sigma)\frac{y_{i-1}^j - 2y_i^j + y_{i+1}^j}{h^2} + \sigma \frac{y_{i-1}^{j-1} - 2y_i^{j-1} + y_{i+1}^{j-1}}{h^2} + \varphi_i^j
$$

Вывод программы

```
Solution with steps h = 0.05 and t = 0.1:
1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
1.0000 0.9950 0.9901 0.9852 0.9803 0.9754 0.9706 0.9657 0.9609 0.9561 0.9514
1.0000 0.9900 0.9802 0.9705 0.9609 0.9514 0.9420 0.9326 0.9234 0.9142 0.9051
1.0000 0.9851 0.9705 0.9561 0.9420 0.9280 0.9142 0.9006 0.8873 0.8741 0.8610
1.0000 0.9802 0.9608 0.9419 0.9233 0.9051 0.8873 0.8698 0.8526 0.8357 0.8191
1.0000 0.9753 0.9513 0.9279 0.9051 0.8828 0.8611 0.8399 0.8192 0.7990 0.7793
1.0000 0.9704 0.9418 0.9141 0.8872 0.8611 0.8357 0.8111 0.7872 0.7639 0.7413
1.0000 0.9656 0.9324 0.9005 0.8696 0.8398 0.8111 0.7833 0.7564 0.7304 0.7052
1.0000 0.9608 0.9232 0.8870 0.8524 0.8191 0.7871 0.7564 0.7268 0.6983 0.6709
1.0000 0.9560 0.9140 0.8738 0.8355 0.7989 0.7639 0.7304 0.6983 0.6676 0.6382
1.0000 0.9512 0.9049 0.8608 0.8190 0.7792 0.7413 0.7053 0.6710 0.6383 0.6071
1.0000 0.9465 0.8959 0.8480 0.8027 0.7599 0.7194 0.6810 0.6447 0.6102 0.5775
1.0000 0.9418 0.8870 0.8354 0.7868 0.7411 0.6981 0.6576 0.6194 0.5833 0.5493
1.0000 0.9371 0.8781 0.8229 0.7712 0.7228 0.6774 0.6349 0.5950 0.5576 0.5225
1.0000 0.9324 0.8694 0.8107 0.7560 0.7050 0.6574 0.6130 0.5716 0.5331 0.4970
1.0000 0.9277 0.8607 0.7986 0.7410 0.6875 0.6379 0.5919 0.5492 0.5096 0.4728
1.0000 0.9231 0.8522 0.7867 0.7263 0.6705 0.6190 0.5715 0.5276 0.4871 0.4497
1.0000 0.9185 0.8437 0.7750 0.7119 0.6539 0.6007 0.5518 0.5068 0.4656 0.4277
1.0000 0.9139 0.8353 0.7634 0.6978 0.6377 0.5829 0.5327 0.4869 0.4450 0.4067
1.0000 0.9094 0.8270 0.7520 0.6839 0.6219 0.5656 0.5143 0.4677 0.4254 0.3868
1.0000 0.9048 0.8187 0.7408 0.6703 0.6065 0.5488 0.4966 0.4493 0.4066 0.3679

Max difference between true and numerical values:
h\tau    0.100000    0.010000    0.001000    0.000100
0.100    6.53e-04    4.44e-05    3.90e-05    3.89e-05
0.010    6.50e-04    6.92e-06    4.48e-07    3.95e-07
0.001    6.50e-04    6.83e-06    6.97e-08    5.36e-09
```

Для оценивания рассмотрел целую сетку параметров и максимум модуля разности искомой функции и численного решения, т.к. на мой взгляд он лучше показывает порядок чем какая-либо норма.

Использовался метод прогонки из библиотеки numpy. На вход принимает матрицу из диагоналей и их количество (в нашем случае одна над и одна под)

Листинг программы:

```
import numpy as np
from scipy.linalg import solve_banded

def true_ans(h, t):
    w_h = np.linspace(0, 1, round(1 / h) + 1)
    w_t = np.linspace(0, 1, round(1 / t) + 1)
    return np.exp(-np.dot(w_h[:, np.newaxis], w_t[np.newaxis, :]))

def solve(h, t):
    w_h = np.linspace(0, 1, round(1 / h) + 1)
    w_t = np.linspace(0, 1, round(1 / t) + 1)
    fi = lambda x, t: np.exp(-t * x) * (x**2 - t**2 + (t**4 + (2 - t*x)**2 - 2) * h**2 / 12)
    mu_0 = lambda x, tau: -x + tau / 2 * x**2 - tau**2 / 6 * x**3 + tau**8 / 24 * (x**4 - 2)
    sigma = 0.25 - h**2 / 12 / t**2

    y = np.zeros((w_h.size, w_t.size))
    y[:, 0] = 1
    y[:, 1] = mu_0(w_h, t) * t + 1
    y[0, :] = 1
    y[-1, :] = np.exp(-w_t)

    A = np.array([
        [0] * 2 + [sigma / h ** 2] * (w_h.size - 2),
        [1] + [-(2 * sigma / h ** 2 + 1 / t ** 2)] * (w_h.size - 2) + [1],
        [sigma / h ** 2] * (w_h.size - 2) + [0] * 2
    ])

    for j in range(1, w_t.size - 1):
        b = (2 * y[1:-1, j] - y[1:-1, j - 1]) / t**2 + \
                (1 - 2 * sigma) * (y[:-2, j] - 2 * y[1:-1, j] + y[2:, j]) / h**2 + \
                sigma * (y[:-2, j - 1] - 2 * y[1:-1, j - 1] + y[2:, j - 1]) / h**2 + \
                fi(w_h[1:-1], w_t[j])
        y[:, j + 1] = solve_banded((1, 1), A, [1, *-b, np.exp(-w_t[j + 1])])
    return y

print('Solution with steps h = 0.05 and t = 0.1:')
for row in solve(0.05, 0.1):
    for num in row:
        print('%7.4f' % num, end='')
    print()
print('\nMax difference between true and numerical values:')
print('h\\t   %10f %10f %10f %10f' % tuple(np.power(10., [-1, -2, -3, -4])))
for h in np.power(10., [-1, -2, -3]):
    print('%5.3f' % h, end='')
    for t in np.power(10., [-1, -2, -3, -4]):
        print('%11.2e' % np.max(np.abs(true_ans(h, t) - solve(h, t))), end='')
    print()
```

Выводы

Таким образом мы получили точное (было найдено с помощью Wolfram) и приближенное решение для уравнения колебания.

Хоть и система оказалась не устойчивой по принципу максимума, она все же оказалась устойчивой по материалам из лекций.

Построенная схема имеет порядок $O(h^4 + \tau^2)$, что подтверждается на практике. Убедится в этом можно посмотрев на матрицу невязок и теоретические расчеты.

Лист оценивания

№	Параметры оценивания	Оценка
1	Оформление отчета: необходимые формулы набраны и получены самостоятельно	3
2	Программирование: алгоритм закодирован и отлажен самостоятельно	3
3	Анализ результатов: полученные результаты оценены самостоятельно	3
	Баллы преподавателя	