Московский государственный университет им. М.В. Ломоносова
Механико-математический факультет
Кафедра математической теории интеллектуальных систем

Отчет по практикуму на ЭВМ

Задача 1 вариант 27

Выполнил студент 411 группы
Орлов Г.В.

Москва 2022

1 Постановка задачи

Для параболического уравнения в единичном квадрате Ω

$$\frac{\partial u}{\partial t} - \Delta u = f,$$

$$u|_{\partial\Omega\times[0,T]} = 0, u(0,x) = u_0(x)$$

Выписать схему метода дробных шагов и сравнить полученное решение с точным.

2 Метод решения

Метод дробных шагов заключается в том, чтобы с уровня τ по t перейти на уровень $\tau + \frac{1}{2}$ при помощи аппроксимации по x, а затем с уровня $\tau + \frac{1}{2}$ перейти на уровень $\tau + 1$ при помощи аппроксимации по y.

Строим сетку с шагом h по x и y, и с шагом τ по t. Разностная схема будет выглядеть так:

$$\begin{cases} \dfrac{u_{ij}^{\tau+1/2} - u_{ij}^{\tau}}{\tau} = \dfrac{1}{h^2}(u_{i+1j}^{\tau+1/2} - 2u_{ij}^{\tau+1/2} + u_{i-1j}^{\tau+1/2}) + \dfrac{1}{2}f_{ij}^{\tau} \quad (1) \\ \dfrac{u_{ij}^{\tau+1} - u_{ij}^{\tau+1/2}}{\tau} = \dfrac{1}{h^2}(u_{ij+1}^{\tau+1} - 2u_{ij}^{\tau+1} + u_{ij-1}^{\tau+1}) + \dfrac{1}{2}f_{ij}^{\tau+1} \quad (2) \end{cases}$$

Решаем систему уравнений (1), не забывая про граничные условия:

$$\begin{cases} \text{h}^2 u_{ij}^{\tau+1/2} - h^2 u_{ij}^{\tau} = \tau(u_{i+1j}^{\tau+1/2} - 2u_{ij}^{\tau+1/2} + u_{i-1j}^{\tau+1/2}) + \frac{\tau h^2}{2} f_{ij}^{\tau} \\ \text{u}_{0j}^{\tau+1/2} = u_{Nj}^{\tau+1/2} = 0 \end{cases}$$

Из этой системы вытекает такое матричное уравнение(при фиксированном j):

$$AX = D$$

где

$$A_{N+1,N+1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 \\ \tau & \text{-(h}^2 + 2\tau) & \tau & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 \\ 0 & \tau & \text{-(h}^2 + 2\tau) & \tau & 0 & \cdots & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & \tau & \text{-(h}^2 + 2\tau) & \tau \\ 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\text{X} = \begin{pmatrix} u_{0j}^{\tau+1/2} \\ u_{1j}^{\tau+1/2} \\ u_{2j}^{\tau+1/2} \\ \vdots \\ u_{N-2j}^{\tau+1/2} \\ u_{N-1j}^{\tau+1/2} \\ u_{Nj}^{\tau+1/2} \end{pmatrix} \text{ - матрица для ответа, } \text{D} = \begin{pmatrix} 0 \\ \text{-h}^2 u_{1j}^{\tau} - \frac{\tau h^2}{2} f_{1j}^{\tau} \\ \text{-h}^2 u_{2j}^{\tau} - \frac{\tau h^2}{2} f_{2j}^{\tau} \\ \vdots \\ \text{-h}^2 u_{N-2j}^{\tau} - \frac{\tau h^2}{2} f_{N-2j}^{\tau} \\ \text{-h}^2 u_{N-1j}^{\tau} - \frac{\tau h^2}{2} f_{N-1j}^{\tau} \\ 0 \end{pmatrix}$$

Для матрицы A проводится алгоритм Гаусса, по результату которого получается строка $(u_{0j}^{\tau+1/2}, u_{1j}^{\tau+1/2}, \cdots, u_{Nj}^{\tau+1/2})$ для фиксированного j. Этот шаг проводится для всех j от 0 до N, и в итоге мы получаем значение функции на слое $\tau + 1/2$.
Затем абсолютно аналогично решается система уравнений (2), после решения которой нам становится известен слой $\tau + 1$. Далее алгоритм повторяет это же действие, пока не дойдет до последнего слоя по t. Задача решена.

3 Сравнение с точным решением

Для сравнения с точным решением точное решение необходимо найти. Пусть

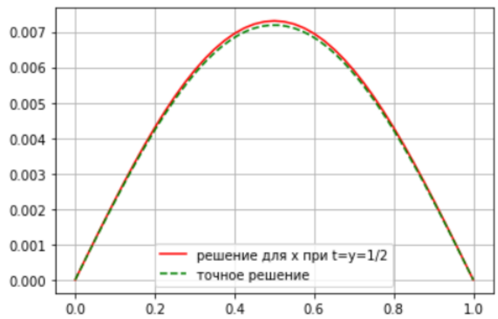
$$u(t, x, y) = e^{-\pi^2 t} sin(\pi x) * sin(\pi y)$$

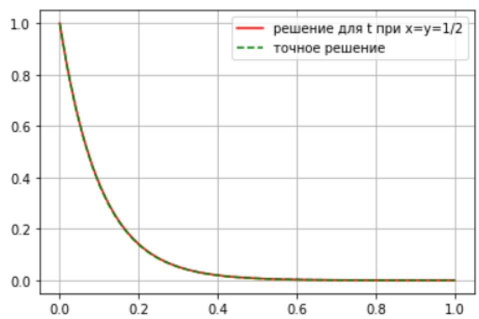
С помощью исходного уравнения находим f:

$$f(t, x, y) = \pi^2 e^{-\pi^2 t} sin(\pi x) * sin(\pi y)$$

Такая пара функций подходит в качестве точного решения задачи. Решим задачу численно с этими начальными данными и сравним полученный результат с точным.

Программа запоняет 3 файла: в фале x.txt находятся все точки функции u. В файле datax.txt находятся координаты функции $u|_{y=t=\frac{1}{2}}(x)$.В файле datat.txt находятся координаты функции $u|_{y=x=\frac{1}{2}}(t)$. В прграммах "графикх.ipynb"и "графикt.ipynb"рисуются графики для сравнения с точными значениями. Для $u|_{y=t=\frac{1}{2}}(x)$ это $e^{-\frac{\pi^2}{2}} sin(\pi x)$, для $u|_{y=x=\frac{1}{2}}(t)$ это $e^{-\pi^2 t}$. Привожу графики для параметров n=40, t=320.





Слева график для $u|_{y=t=\frac{1}{2}}(x)$, справа - для $u|_{y=x=\frac{1}{2}}(t)$.

Теперь рассмотрим погрешности: порядок аппроксимации функции - $O(\tau + h^2)$, а значит погрешность может убывать и по квадрату, и линейно. Для линейного убывания погрешности параметры t и n дожны быть одного порядка, тогда при увеличении t погрешность будет убывать линейно. Привожу таблицу:

err	n	t
$5.18 * 10^{-2}$	20	20
$2.9 * 10^{-2}$	20	40
$1.54 * 10^{-2}$	20	80
$8.38 * 10^{-3}$	20	160

```
[orlov_g@925e624d717f task3]$ ./a.out 20 20
LeakTracer 3.0.0 (shared library) -- LGPLv2
5.179184e-02
[orlov_g@925e624d717f task3]$ ./a.out 20 40
LeakTracer 3.0.0 (shared library) -- LGPLv2
2.885071e-02
[orlov_g@925e624d717f task3]$ ./a.out 20 80
LeakTracer 3.0.0 (shared library) -- LGPLv2
1.543610e-02
[orlov_g@925e624d717f task3]$ ./a.out 20 160
LeakTracer 3.0.0 (shared library) -- LGPLv2
8.380865e-03
```

Для квадратичного убывания погрешности параметр t дожнен быть на несколько порядков больше n. Тогда при увеличении n погрешность убывает квадратично:

err	n	t
$1.51 * 10^{-2}$	5	10000
$4.24 * 10^{-3}$	10	10000
$1.15 * 10^{-3}$	20	10000

```
[orlov_g@925e624d717f task3]$ ./a.out 5 10000
LeakTracer 3.0.0 (shared library) -- LGPLv2
1.508393e-02
[orlov_g@925e624d717f task3]$ ./a.out 10 10000
LeakTracer 3.0.0 (shared library) -- LGPLv2
4.236010e-03
[orlov_g@925e624d717f task3]$ ./a.out 20 10000
LeakTracer 3.0.0 (shared library) -- LGPLv2
1.148711e-03
```

Я использую достаточно небольшие значения параметров, так как метод дробных шагов не очень эффективен по времени, и при больших значения n программа будет работать очень долго.