Московский авиационный институт
(национальный исследовательский университет)

Факультет информационных технологий и прикладной математики
Кафедра вычислительной математики и программирования

Отчет по лабораторной работе №5
по курсу «Численные методы»

____

Дата: 14.10.2023

Задание: Используя явную и неявную конечно-разностные схемы, а также схему Кранка - Николсона, решить начально-краевую задачу для дифференциального уравнения параболического типа. Осуществить реализацию трех вариантов аппроксимации граничных условий, содержащих производные: двухточечная аппроксимация с первым порядком, трехточечная аппроксимация со вторым порядком, двухточечная аппроксимация со вторым порядком. В различные моменты времени вычислить погрешность численного решения путем сравнения результатов с приведенным в задании аналитическим решением U(x,t) . Исследовать зависимость погрешности от сеточных параметров τ ,h.

Вариант:

$$\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2}, a > 0;$$

$$u(0,t) = 0;$$

$$u(1,t) = 0,$$

$$u(x,0) = \sin(2\pi x);$$

$$U = \exp(-4\pi^2 at)\sin(2\pi x)$$

Решение: Нанесем на пространственно-временную область $0 \le x \le 1$, $0 \le t \le T$ конечноразностную сетку $\omega_{h\tau}$: $\omega_{h\tau} = \{x_j = jh,\ j = \overline{0,N},\ t^k = k\tau,\ k = \overline{0,K}\}$ с пространственным шагом h=1/N и шагом по времени τ=T/K. Введем два временных слоя: нижний $t^k = k\tau$, на котором распределение искомой функции $u(x_j, t^k)$, $j = \overline{0,N}$, известно (при k=0 распределение определяется начальным условием $u(x_j, t^0) = \sin(2\pi x_j)$) и верхний временной слой $t^{k+1} = (k+1)\tau$, на котором распределение искомой функции $u(x_j, t^{k+1})$, $j = \overline{0,N}$ подлежит определению. Сеточной функцией задачи назовем однозначное отображение целых аргументов j, k в значения функции $u_j^k = u(x_j, t^k)$. На введенной сетке введем сеточные функции u_j^k, u_j^{k+1}, первая из которых известна, вторая – подлежит определению. Для ее определения в задаче заменим (аппроксимируем) дифференциальные операторы отношением конечных разностей, получим $\frac{\partial u}{\partial t} = \frac{u_j^{k+1} - u_j^k}{\tau} + O(\tau)$, $\frac{\partial^2 u}{\partial x^2} = \frac{u_{j+1}^k - 2u_j^k + u_{j-1}^k}{h^2} + O(h^2)$. Подставляя, получим **явную конечно-разностную схему** для этой задачи в форме $\frac{u_j^{k+1} - u_j^k}{\tau} = a * \frac{u_{j+1}^k - 2u_j^k + u_{j-1}^k}{h^2} + O(\tau + h^2), u_0^k = 0; u_N^k = 1, k = \overline{0, K}; u_j^0 = \sin(2\pi x_j), j = \overline{0, N}$ где для каждого j -го уравнения все значения сеточной функции известны, за исключением одного - u_j^{k+1}, которое может быть определено явно из соотношений. Если дифференциальный оператор по пространственной переменной аппроксимировать отношением конечных разностей на верхнем временном слое $\frac{\partial^2 u}{\partial x^2} = \frac{u_{j+1}^{k+1} - 2u_j^{k+1} + u_{j-1}^{k+1}}{h^2} + O(h^2)$ то после подстановки, получим **неявную конечноразностную схему** для этой задачи $\frac{u_j^{k+1} - u_j^k}{\tau} = a * \frac{u_{j+1}^{k+1} - 2u_j^{k+1} + u_{j-1}^{k+1}}{h^2} + O(\tau + h^2), u_0^k = 0; u_N^k = 1, k = \overline{0, K}; u_j^0 = \sin(2\pi x_j)$, $j = \overline{0, N}$. Теперь сеточную функцию u_j^{k+1} на верхнем временном слое можно получить из решения СЛАУ с трехдиагональной матрицей, которую решаем методом прогонки. Рассмотрим неявно-явную схему с весами для простейшего уравнения теплопроводности $\frac{u_j^{k+1} - u_j^k}{\tau} = \theta a * \frac{u_{j+1}^{k+1} - 2u_j^{k+1} + u_{j-1}^{k+1}}{h^2} + (1 - \theta) a * \frac{u_{j+1}^k - 2u_j^k + u_{j-1}^k}{h^2}$,

где θ - вес неявной части конечно-разностной схемы, 1−θ - вес для явной части, причем 0≤θ≤1. При θ=1 имеем полностью неявную схему, при θ=0 - полностью явную схему, и при θ=1/2 - **схему Кранка-Николсона.** Так как в задаче нет производных в краевых условиях, то аппроксимация не требуется.

Код для явной схемы:

```
private void Yav2D()
{
    double h = l / N;
```

```
double tau = sig * Math.Pow(h, 2) / a;
double[,] u = new double[K + 1, N + 1];
for (int j = 0; j <= N; j++) u[0, j] = j * h + Math.Sin(Math.PI * j * h);
for (int k = 0; k <= K - 1; k ++)
{
   for (int j = 1; j <= N-1; j++)
   {
      u[k + 1, j] = sig * u[k, j + 1] + (1 - 2 * sig) * u[k, j] + sig * u[k, j - 1];
   }
   u[k + 1, 0] = 0;
   u[k + 1, N] = 1;
}
```

```
        for (int k = 0; k <= K; k++)
        {
            for (int j = 0; j <= N; j++)
            {
                U[k, j] = u[k, j];
                dt[k] += Math.Abs(u[k, j] - f(h * j, tau * k));
            }
        }
    }
```

Код для неявной схемы:

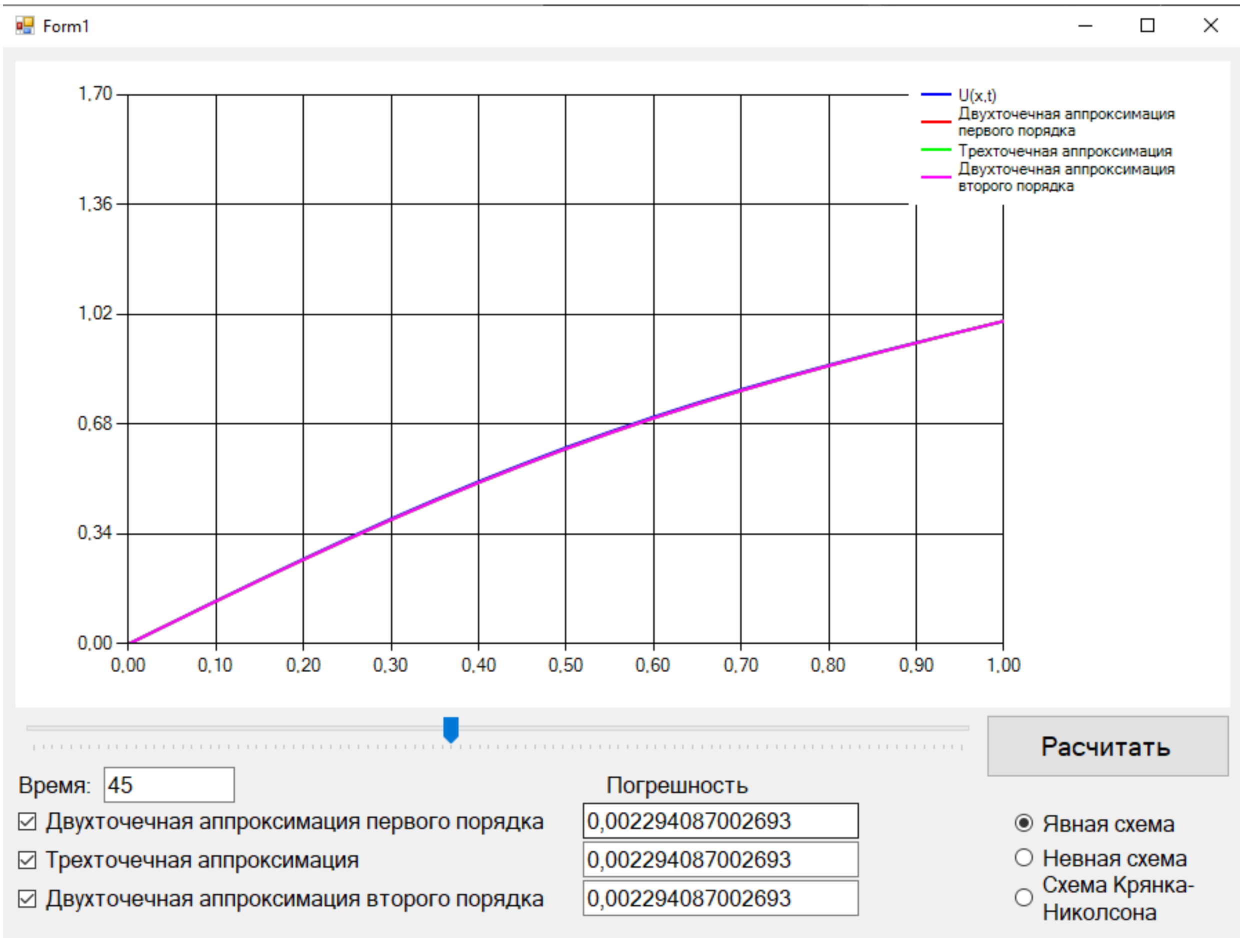
```
    private void NeYav2D()
    {
        double h = l / N;
        double tau = sig * Math.Pow(h, 2) / this.a;
        double[,] u = new double[K + 1, N + 1];
        double[] b = new double[N + 1];
        double[] a = new double[N];
        double[] c = new double[N];
        double[] d = new double[N + 1];
        double[] x;
        for (int j = 0; j <= N; j++) u[0, j] = j * h + Math.Sin(Math.PI * j * h);
        for (int k = 0; k <= K - 1; k++)
        {
            for (int j = 0; j <= N - 1; j++)
            {
                a[j] = sig; b[j] = -(1 + 2 * sig); c[j] = sig; d[j] = -u[k, j];
            }
            b[0] = 1; c[0] = 0; d[0] = 0;
            a[N - 1] = 0; b[N] = 1; d[N] = 1;
            x = Progon(a,b,c,d).ToArray();
            for (int j = 0; j <= N; j++)
            {
                u[k + 1, j] = x[j];
            }
        }
        for (int k = 0; k <= K; k++)
        {
            for (int j = 0; j <= N; j++)
            {
                U[k, j] = u[k, j];
                dt[k] += Math.Abs(u[k, j] - f(h * j, tau * k));
            }
        }
    }
```

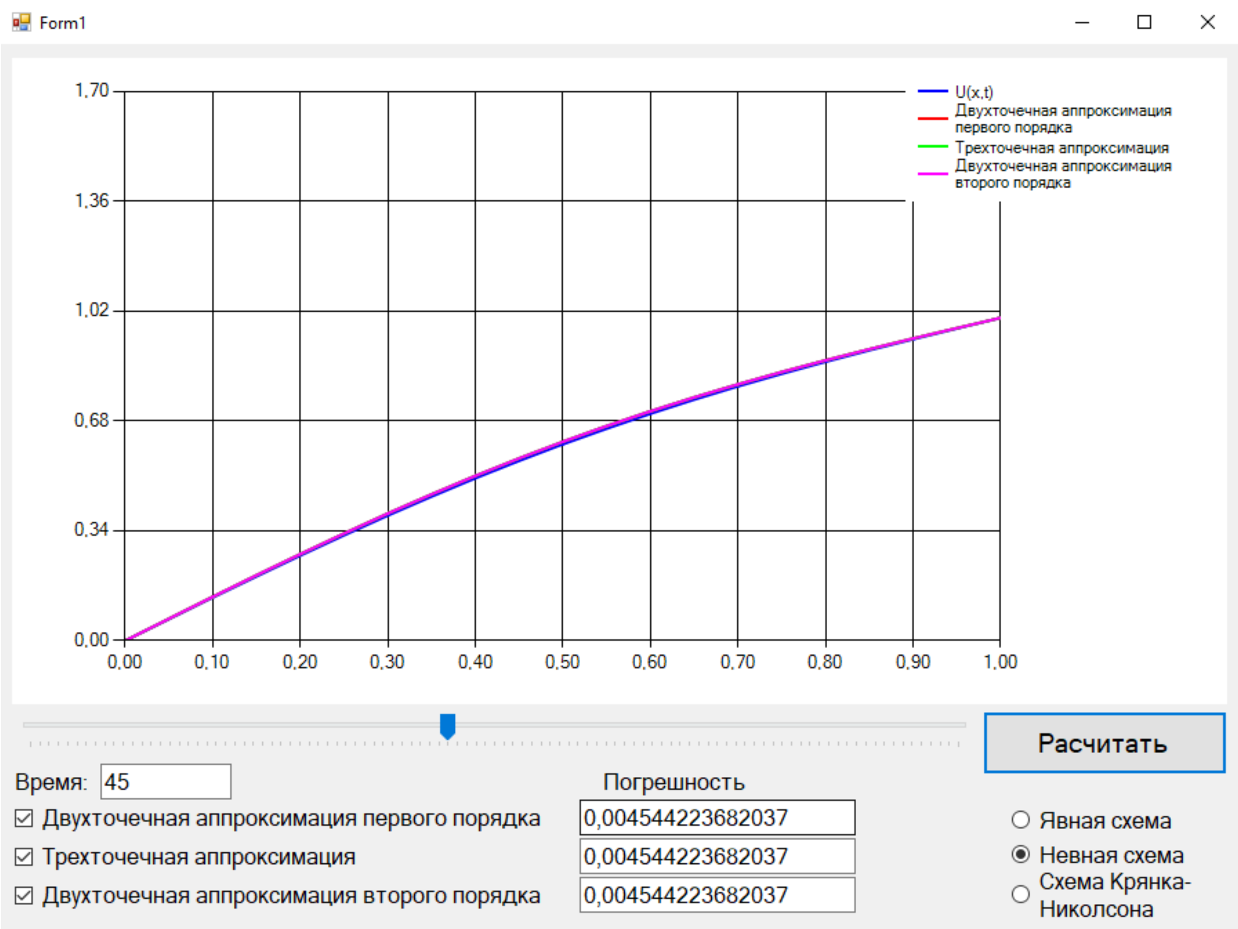
Код для схемы Кранка-Николсона:

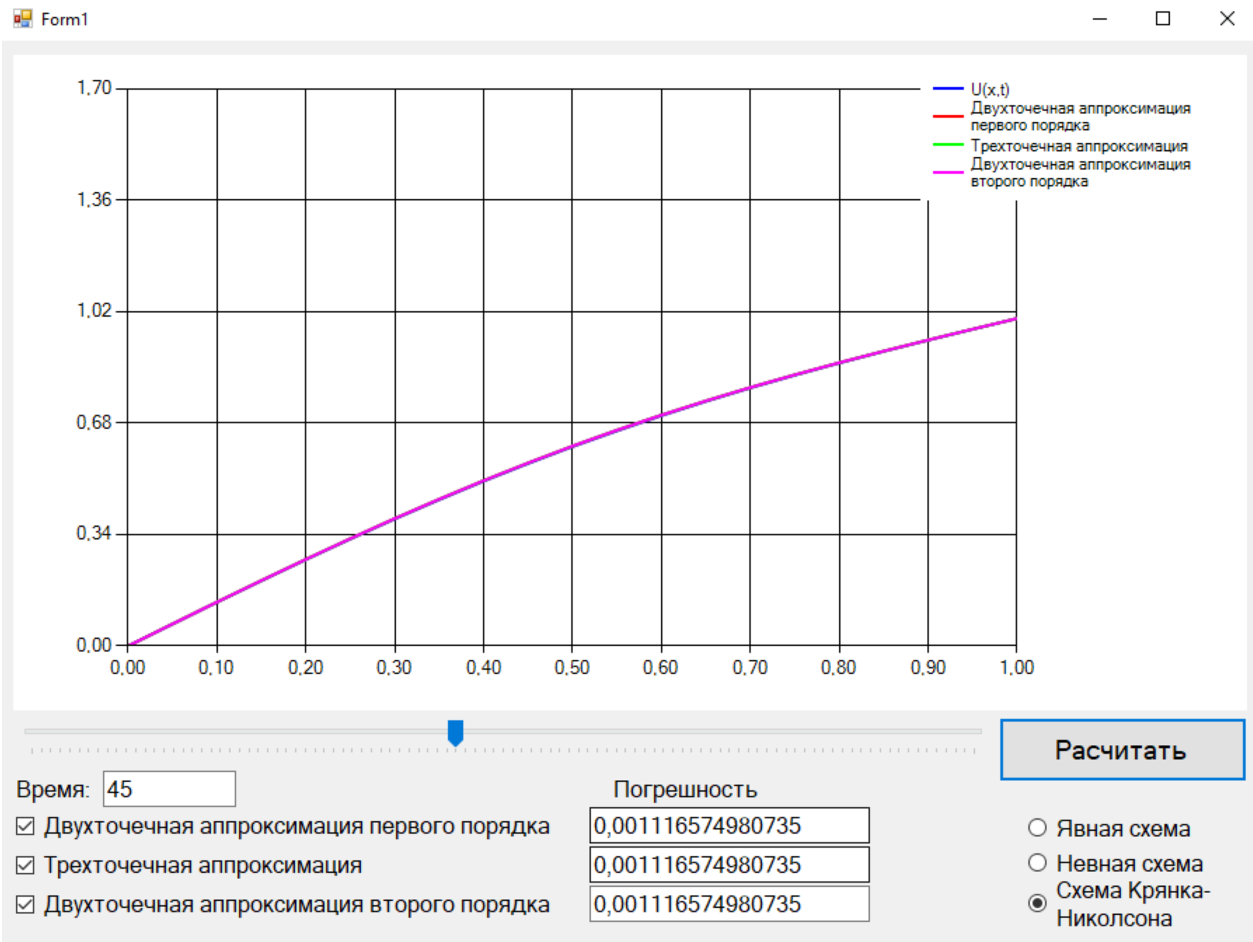
```
    private void KN2D()
    {
        double h = l / N;
        double tau = sig * Math.Pow(h, 2) / this.a;
```

```
    double[,] u = new double[K + 1, N + 1];
    double[] b = new double[N + 1];
    double[] a = new double[N];
    double[] c = new double[N];
    double[] d = new double[N + 1];
    double[] x;
    double r = this.a * tau / (h * h);
    for (int j = 0; j <= N; j++) u[0, j] = j * h + Math.Sin(Math.PI * j * h);
    for (int k = 0; k <= K - 1; k++)
    {
      for (int j = 1; j <= N - 1; j++)
      {
        a[j] = -r/2; b[j] = r+1; c[j] = -r/2; d[j] = r / 2 * (u[k,j - 1] + u[k,j + 1]) + u[k,j] * (1 - r);
      }
      b[0] = 1; c[0] = 0; d[0] = 0; a[0] = -r / 2;
      a[N - 1] = 0; b[N] = 1; d[N] = 1; c[N - 1] = -r / 2;
      x = Progon(a, b, c, d).ToArray();
      for (int j = 0; j <= N; j++)
      {
        u[k + 1, j] = x[j];
      }
    }
    for (int k = 0; k <= K; k++)
    {
      for (int j = 0; j <= N; j++)
      {
        U[k, j] = u[k, j];
        dt[k] += Math.Abs(u[k, j] - f(h * j, tau * k));
      }
    }
}
```

Результаты:


Form1
1,70
1,36
1,02
0,68
0,34
0,00
0,00
0,10
0,20
0,30
0,40
0,50
0,60
0,70
0,80
0,90
1,00
U(x,t)
Двухточечная аппроксимация первого порядка
Трехточечная аппроксимация
Двухточечная аппроксимация второго порядка
Расчитать
Время: 45
Погрешность
Двухточечная аппроксимация первого порядка
0,002294087002693
Трехточечная аппроксимация
0,002294087002693
Двухточечная аппроксимация второго порядка
0,002294087002693
Явная схема
Невная схема
Схема Крянка-Николсона


Form1
1,70
1,36
1,02
0,68
0,34
0,00
0,00
0,10
0,20
0,30
0,40
0,50
0,60
0,70
0,80
0,90
1,00
U(x,t)
Двухточечная аппроксимация первого порядка
Трехточечная аппроксимация
Двухточечная аппроксимация второго порядка
Расчитать
Время: 45
Погрешность
Двухточечная аппроксимация первого порядка
0,004544223682037
Трехточечная аппроксимация
0,004544223682037
Двухточечная аппроксимация второго порядка
0,004544223682037
Явная схема
Невная схема
Схема Крянка-Николсона



Вывод: Мной было реализовано 3 схемы решения УРЧП параболического типа 1D, в каждом из которых по 3 метода аппроксимации производной в краевых условиях, однако так как в варианте нет производных в краевых условиях, то ничего не аппроксимируется, а следовательно точность при всех аппроксимациях будет одинаковая, как это видно на скриншотах выше.