Московский авиационный институт
(национальный исследовательский университет)

Факультет информационных технологий и прикладной математики
Кафедра вычислительной математики и программирования

Отчет по лабораторной работе №6
по курсу «Численные методы»

Дата: 11.11.2023

Задание: Используя явную схему крест и неявную схему, решить начально-краевую задачу для дифференциального уравнения гиперболического типа. Аппроксимацию второго начального условия произвести с первым и со вторым порядком. 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:

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

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

$$u_x(\pi,t) - u(1,t) = 0,$$

$$u(x,0) = \sin x + \cos x;$$

$$u_t(x,0) = -a(\sin x + \cos x);$$

$$U = \sin(x - at) + \cos(x + at)$$

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 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

j

аргументов j, k в значения функции $u^k = u\,(x_j\ , t^k)$. 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 $\frac{\partial^2 u}{\partial t^2} = \frac{u_j^{k+1} - 2u_j^k + u_j^{k-1}}{\tau^2} + O(\tau^2)$, $\frac{\partial^2 u}{\partial x^2} = \frac{u_{j+1}^k - 2u_j^k + u_{j-1}^k}{h^2} + O(h^2)$. Подставляя, получим **явную конечно-разностную схему** для этой задачи в форме $\frac{u_j^{k+1} - 2u_j^k + u_j^{k-1}}{\tau^2} = a^2 * \frac{u_{j+1}^k - 2u_j^k + u_{j-1}^k}{h^2} + O(\tau^2 + h^2)$ 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 $\frac{u_j^{k+1} - 2u_j^k + u_j^{k-1}}{\tau^2} = a^2 * \frac{u_{j+1}^{k+1} - 2u_j^{k+1} + u_{j-1}^{k+1}}{h^2} + O(\tau^2 + h^2)$. Теперь сеточную функцию u_j^{k+1} на верхнем временном слое можно получить из решения СЛАУ с трехдиагональной матрицей, которую решаем методом прогонки. В обеих схемах необходимо знать значения u_j^k и u_j^{k-1} на нижних временных слоях. Для k =1 это делается следующим образом: $u^0{}_j = \sin x_j + \cos x_j$. Для определения $u^1{}_j$ можно воспользоваться простейшей аппроксимацией второго начального условия $\frac{u_j^1 - u_j^0}{\tau} = -a(\sin x_j + \cos x_j)$. Откуда для искомых значений u_j^1 получаем следующее выражение: $u_j^1 = \sin x_j + \cos x_j - \tau a(\sin x_j + \cos x_j)$. Недостатком такого подхода является первый порядок аппроксимации второго начального условия. Разложим u_j^1 в ряд Тейлора на точном решении по времени в окрестности t=0. В результате получаем искомую сеточную функцию $u^1{}_j$ со вторым порядком точности: $u_j^1 = \sin x_j + \cos x_j - \tau a(\sin x_j + \cos x_j) + \frac{a^2\tau^2}{2}(\sin x_j + \cos x_j)''$.

Аппроксимация производных направленными разностями первого порядка: $\frac{\partial u}{\partial x} = \frac{u_{j+1}^k - u_j^k}{h}$. Аппроксимации граничных условий второго порядка: $\frac{\partial u}{\partial x} = \frac{-3u_j^k + 4u_{j+1}^k - u_{j+2}^k}{2h}$.

Двухточечная аппроксимация со вторым порядком находится путем разложения в ряд тейлора до второго члена включительно, откуда потом и выражается производная $\frac{\partial u}{\partial x}$.

Код для явной схемы:

```
        private void Yav(bool flag, int apr)
        {
            double h = 1 / N;
            double tau = Math.Sqrt(sig * Math.Pow(h, 2) / Math.Pow(a,2));
            double[,] u = new double[K + 1, N + 1];
            for (int j = 0; j <= N; j++)
            {

                u[0, j] = Math.Sin(j * h) + Math.Cos(j *
                h); if (flag)
                    u[1, j] = Math.Sin(j * h) + Math.Cos(j * h) - a * (Math.Sin(j * h) +
* h)) *             Math.Cos(j
tau;
                else
                    u[1, j] = Math.Sin(j * h) + Math.Cos(j * h) - a * (Math.Sin(j * h) +
                    Math.Cos(j

* h)) * tau + Math.Pow(a, 2) * (-Math.Sin(j * h) - Math.Cos(j * h)) * Math.Pow(tau, 2) / 2;
            }
            for (int k = 1; k <= K - 1; k ++)
            {
                for (int j = 1; j <= N-1; j++)
                {

                    u[k + 1, j] = u[k, j + 1] * sig + u[k, j] * (-2 * sig + 2) + u[k, j - 1] *
u[k - 1,            sig -
j];

                }

                if (apr == 0)
                {
                    u[k + 1, 0] = u[k + 1, 1] / (h + 1);
                    u[k + 1, N] = u[k + 1, N - 1] / (1 - h);
                }
                else if (apr == 1)
                {
                    u[k + 1, 0] = (-4 * u[k + 1, 1] + u[k + 1, 2]) / (-2 * h - 3);
                    u[k + 1, N] = (4 * u[k + 1, N - 1] - u[k + 1, N - 2]) / (-2 * h + 3);
                }
                else if (apr == 2)
                {

                    u[k + 1, 0] = (u[k + 1, 1] + 1 / (2 * sig) * (2 * u[k, 0] - u[k - 1,
1 + 1 / (2 *        0])) / (h +
sig));

                    u[k + 1, N] = (u[k + 1, N - 1] + 1 / (2 * sig) * (2 * u[k, N] - u[k - 1,
                    N])) /

(-h + 1 + 1 / (2 * sig));
                }
            }
            if (apr == 0)
            {
                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));
                    }
                }
            }
```

```
else if (apr == 1)
{
    for (int k = 0; k <= K; k++)
    {
        for (int j = 0; j <= N; j++)
        {
            U2[k, j] = u[k, j];
            dt2[k] += Math.Abs(u[k, j] - f(h * j, tau * k));
        }
```

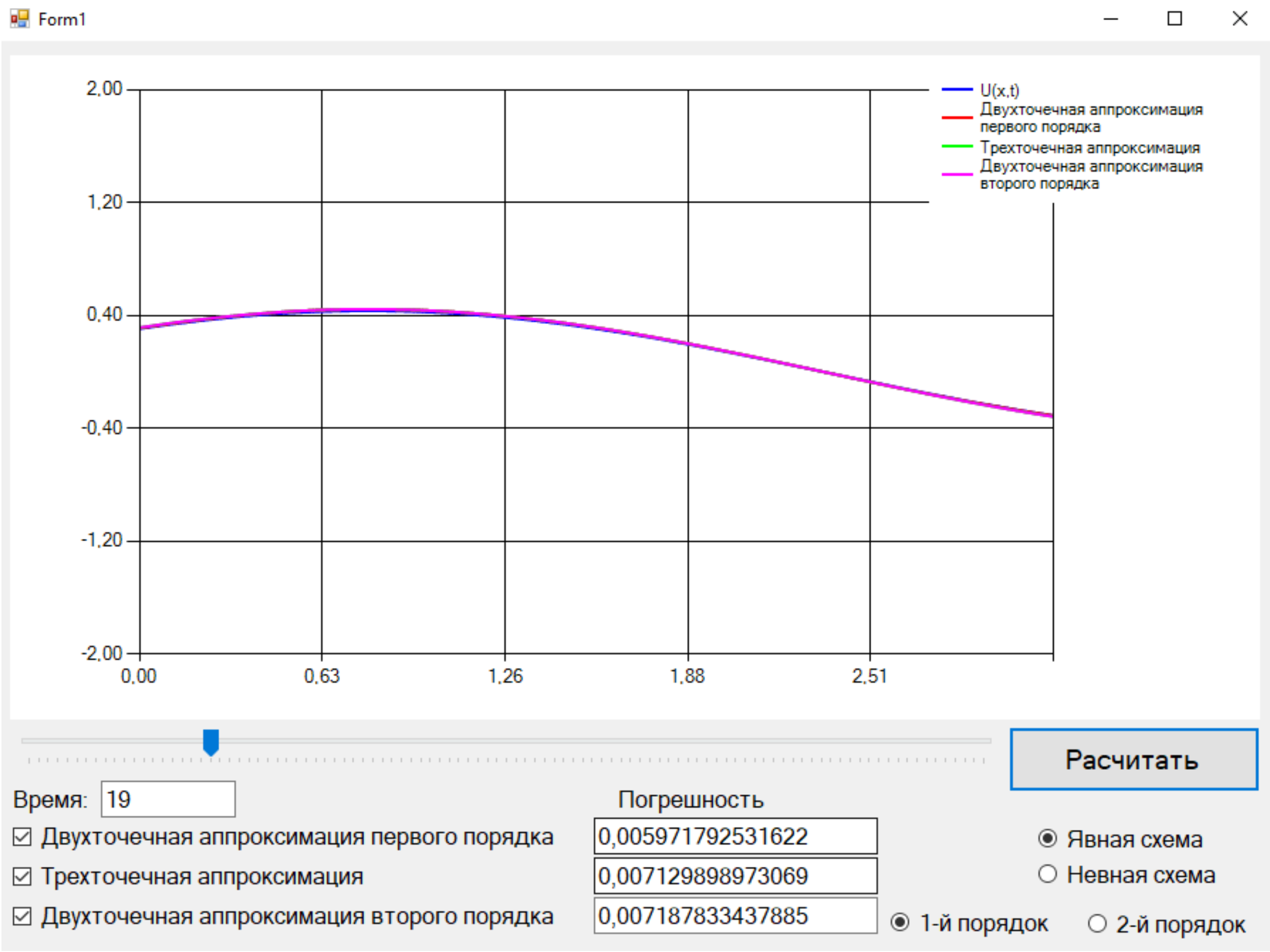
```
            }
        }
        else if (apr == 2)
        {
            for (int k = 0; k <= K; k++)
            {
                for (int j = 0; j <= N; j++)
                {
                    U3[k, j] = u[k, j];
                    dt3[k] += Math.Abs(u[k, j] - f(h * j, tau * k));
                }
            }
        }
    }
```

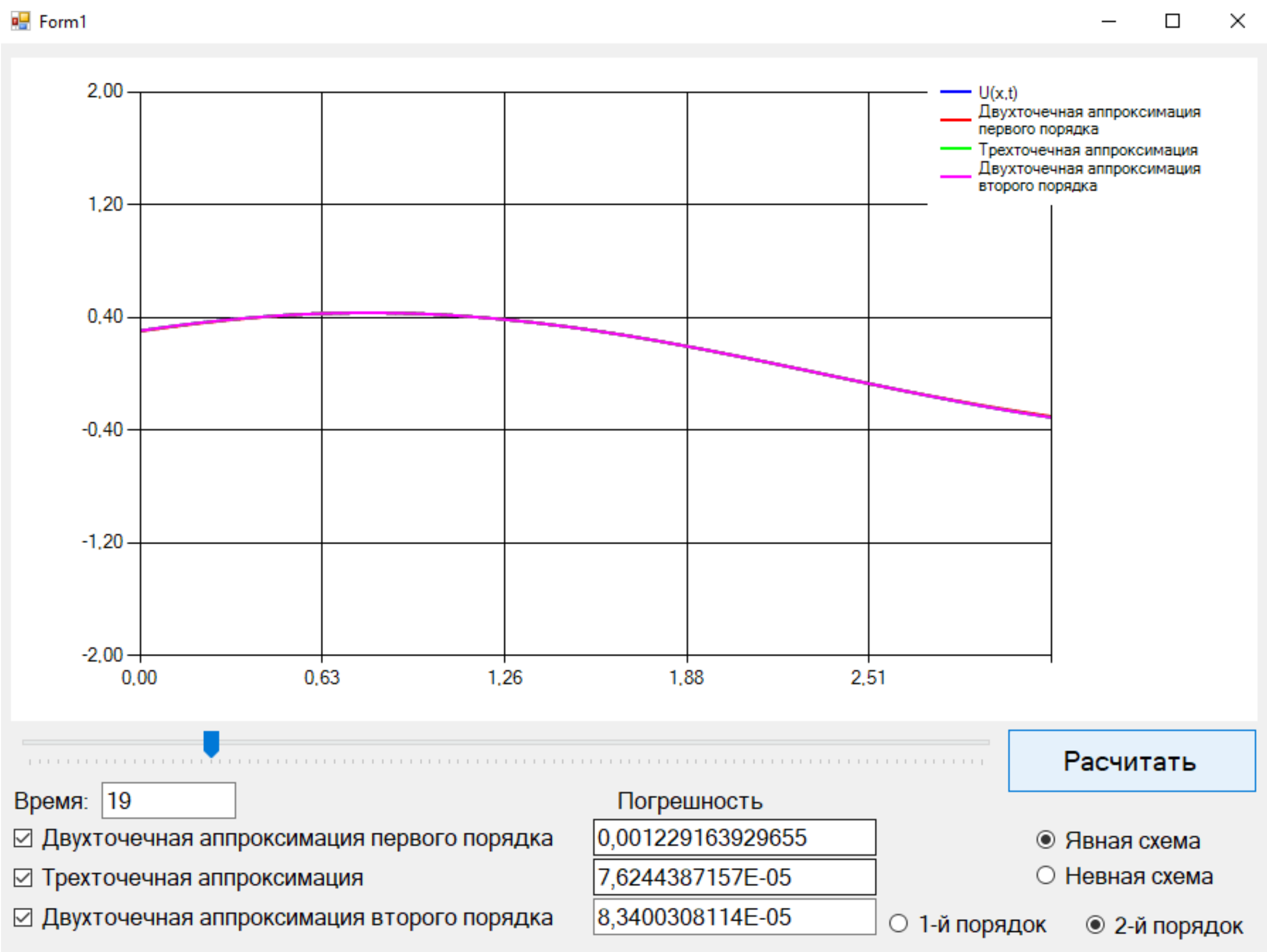
Код для неявной схемы:

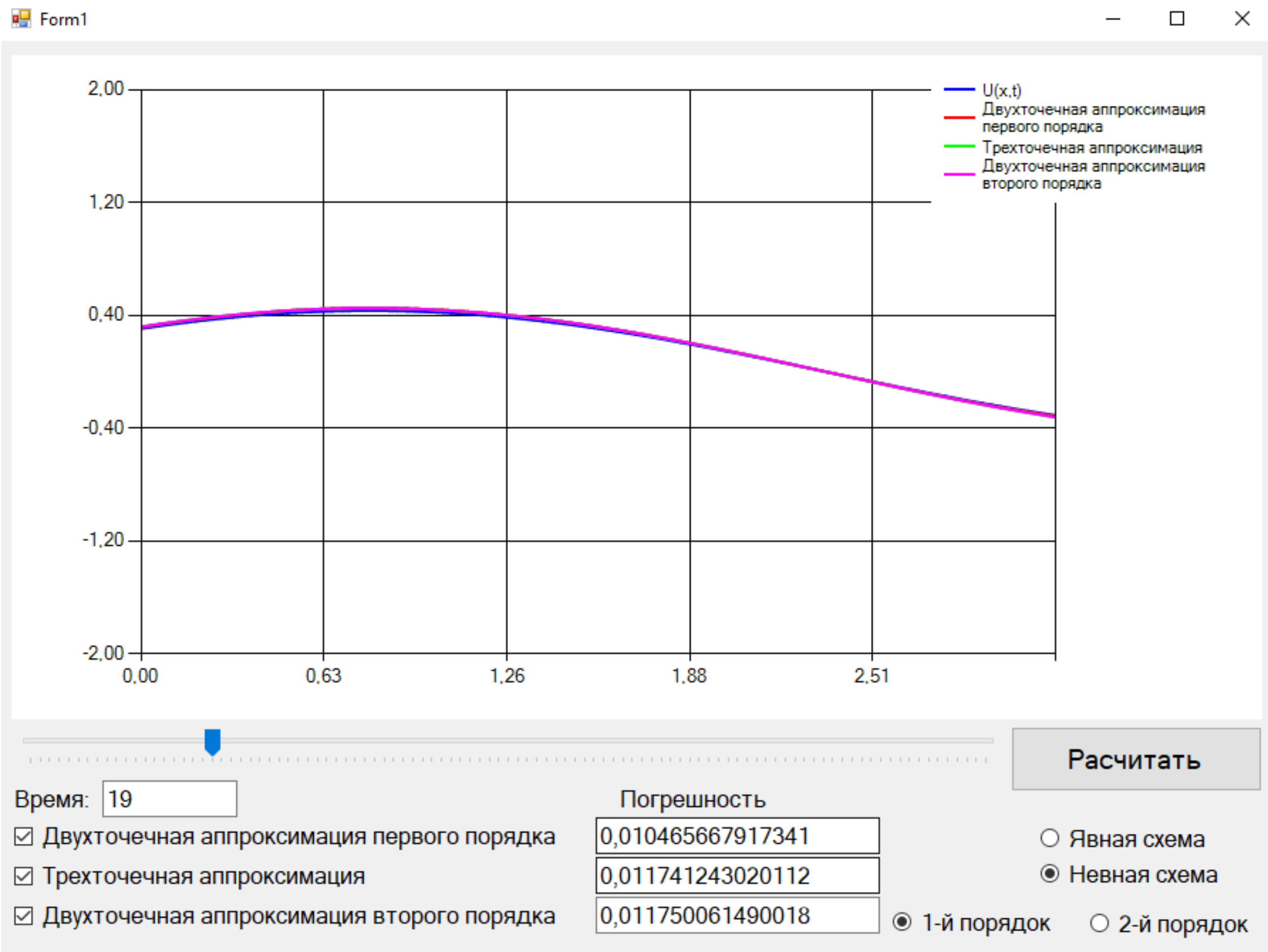
```
    private void NeYav(bool flag, int apr) //Неявная схема 2Т1П
    {
        double h = 1 / N;
        double tau = Math.Sqrt(sig * Math.Pow(h, 2) / Math.Pow(this.a,
        2)); double[,] u = new double[K + 1, N + 1];
        double[] b = new double[N + 1];
        double[] a = new double[N];
        double[] c = new double[N];
        double[] d = new double[N + 1];
        double[] x;
        double p;
        for (int j = 0; j <= N; j++)
        {
            u[0, j] = Math.Sin(j * h) + Math.Cos(j *
            h); if (flag)
                u[1, j] = Math.Sin(j * h) + Math.Cos(j * h) - this.a * (Math.Sin(j * h) +
Math.Cos(j * h)) * tau;
            else
                u[1, j] = Math.Sin(j * h) + Math.Cos(j * h) - this.a * (Math.Sin(j * h) +
Math.Cos(j * h)) * tau + Math.Pow(this.a, 2) * (-Math.Sin(j * h) - Math.Cos(j * h)) *
Math.Pow(tau, 2) / 2;
        }
        for (int k = 1; 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] = 2 * u[k, j] - u[k -
                1, j];
            }
            if (apr == 0)
            {
                b[0] = -1 / h - 1; c[0] = 1/h; d[0] = 0;
                a[N - 1] = -1 / h; b[N] = 1 / h - 1; d[N] = 0;
            }
            else if (apr == 1)
            {

                p = 1 / (2 * h * sig);
                b[0] = -3 / (2 * h) -1 - p * a[0]; c[0] = 4 / (2 * h) - p * b[1]; d[0] = -
d[1];           p *

                a[N - 1] = -4 / (2 * h) + p * b[N - 1]; b[N] = 3 / (2 * h) - 1 + p * c[N -
                1];

d[N] = p * d[N - 1];
            }
            else if (apr == 2)
            {

u[k - 1, 0]);

- u[k - 1, N]);
           }
```

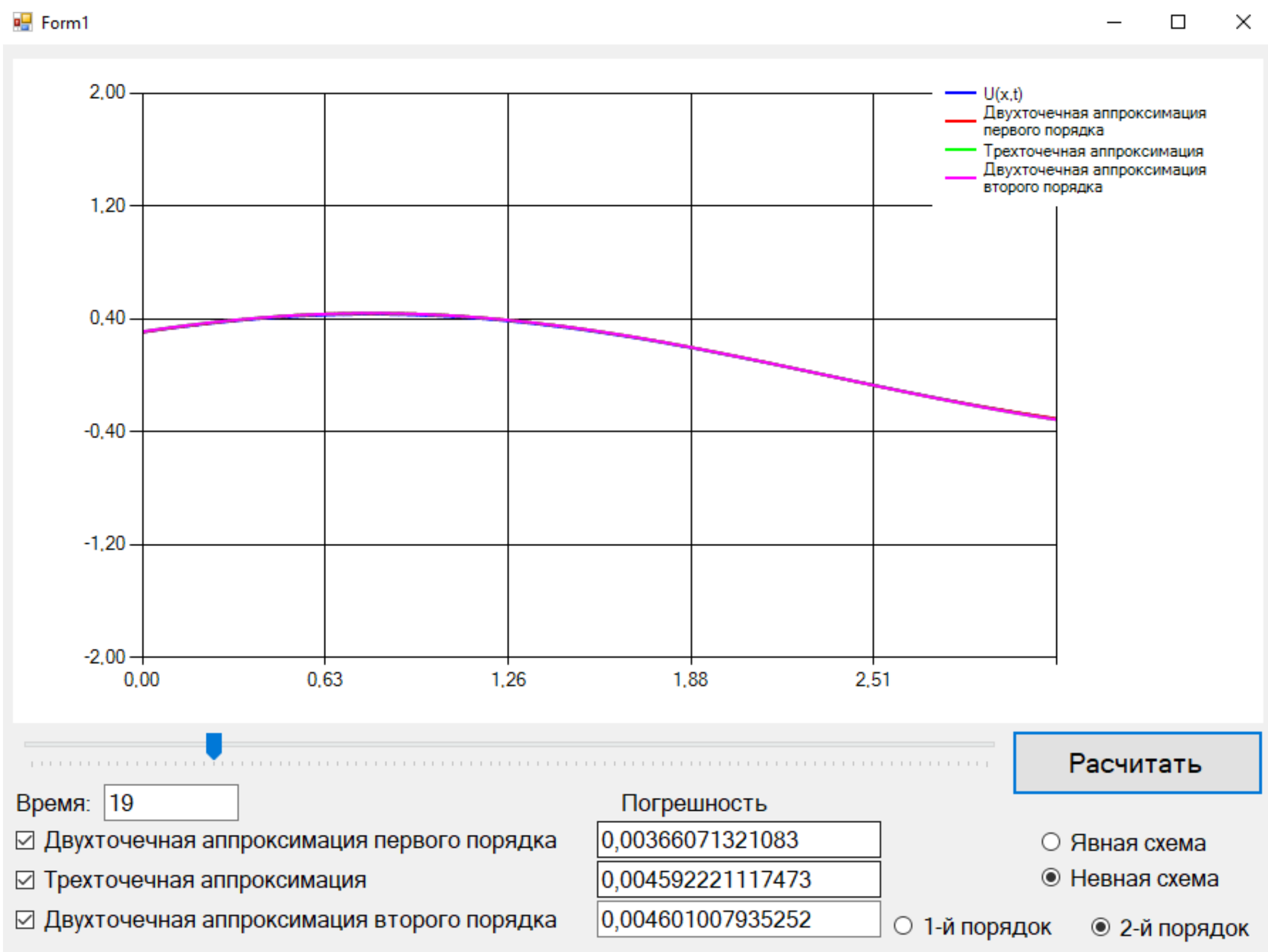
```
b[0] = 1 + 1 /
(2 * sig) + h;     a[N - 1] = -1; b[N] = 1 + 1 / (2 * sig) - h; d[N] = 1 / (2 * sig) * (2 *
c[0] = -1; d[0]    u[k, N]
= 1 / (2 * sig)
* (2 * u[k, 0] -
               x =
               Progon(a,b,c,d).ToArray();
               for (int j = 0; j <= N; j++)
```

```
            {
                u[k + 1, j] = x[j];
            }
        }
        if (apr == 0)
        {
            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));
                }
            }
        }
        else if (apr == 1)
        {
            for (int k = 0; k <= K; k++)
            {
                for (int j = 0; j <= N; j++)
                {
                    U2[k, j] = u[k, j];
                    dt2[k] += Math.Abs(u[k, j] - f(h * j, tau * k));
                }
            }
        }
        else if (apr == 2)
        {
            for (int k = 0; k <= K; k++)
            {
                for (int j = 0; j <= N; j++)
                {
                    U3[k, j] = u[k, j];
                    dt3[k] += Math.Abs(u[k, j] - f(h * j, tau * k));
                }
            }
        }
    }
```
Результаты:


Form1
U(x,t)
Двухточечная аппроксимация первого порядка
Трехточечная аппроксимация
Двухточечная аппроксимация второго порядка
2,00
1,20
0,40
-0,40
-1,20
-2,00
0,00
0,63
1,26
1,88
2,51
Расчитать
Время: 19
Погрешность
Двухточечная аппроксимация первого порядка
0,005971792531622
Трехточечная аппроксимация
0,007129898973069
Двухточечная аппроксимация второго порядка
0,007187833437885
Явная схема
Невная схема
1-й порядок
2-й порядок


Form1
U(x,t)
Двухточечная аппроксимация первого порядка
Трехточечная аппроксимация
Двухточечная аппроксимация второго порядка
2,00
1,20
0,40
-0,40
-1,20
-2,00
0,00
0,63
1,26
1,88
2,51
Расчитать
Время: 19
Погрешность
Двухточечная аппроксимация первого порядка
0,001229163929655
Трехточечная аппроксимация
7,6244387157E-05
Двухточечная аппроксимация второго порядка
8,3400308114E-05
Явная схема
Невная схема
1-й порядок
2-й порядок


Form1
2,00
1,20
0,40
-0,40
-1,20
-2,00
0,00
0,63
1,26
1,88
2,51
U(x,t)
Двухточечная аппроксимация первого порядка
Трехточечная аппроксимация
Двухточечная аппроксимация второго порядка
Расчитать
Время: 19
Погрешность
Двухточечная аппроксимация первого порядка
0,010465667917341
Трехточечная аппроксимация
0,011741243020112
Двухточечная аппроксимация второго порядка
0,011750061490018
Явная схема
Невная схема
1-й порядок
2-й порядок


Form1
2,00
1,20
0,40
-0,40
-1,20
-2,00
0,00
0,63
1,26
1,88
2,51
U(x,t)
Двухточечная аппроксимация первого порядка
Трехточечная аппроксимация
Двухточечная аппроксимация второго порядка
Расчитать
Время: 19
Погрешность
Двухточечная аппроксимация первого порядка
0,00366071321083
Трехточечная аппроксимация
0,004592221117473
Двухточечная аппроксимация второго порядка
0,004601007935252
Явная схема
Невная схема
1-й порядок
2-й порядок

Вывод: Мной было реализовано 2 схемы решения УРЧП гиперболического типа 1D, аппроксимация второго начального условия с первым и со вторым порядком, в каждой схеме по 3 метода аппроксимации производной в краевых условиях.