Практическое задание №1

Задача №2

Студент 305 группы Менделевич Лев Владиславович

Преподаватель Митина Ирина Владимировна

Физический факультет

Московский государственный университет имени М. В. Ломоносова

1.Постановка задачи.

Используя схемvбегущего счета и итерационные методы, решить задачу:

$$\begin{cases} \dfrac{\partial u}{\partial t} - u\dfrac{\partial u}{\partial x} = 0, \quad -1 \le x < 0 \\ u(x,0) = 2 - \dfrac{4}{\pi} arctg(x+2) \\ u(0,t) = (2 - \dfrac{4}{\pi} arctg(2))e^{-t} \end{cases}$$

2.Характеристические уравнения.

Запишем характеристическое уравнение:

$$\frac{dt}{1} = -\frac{dx}{u(x_0,t_0)}$$

Преобразуем к виду:

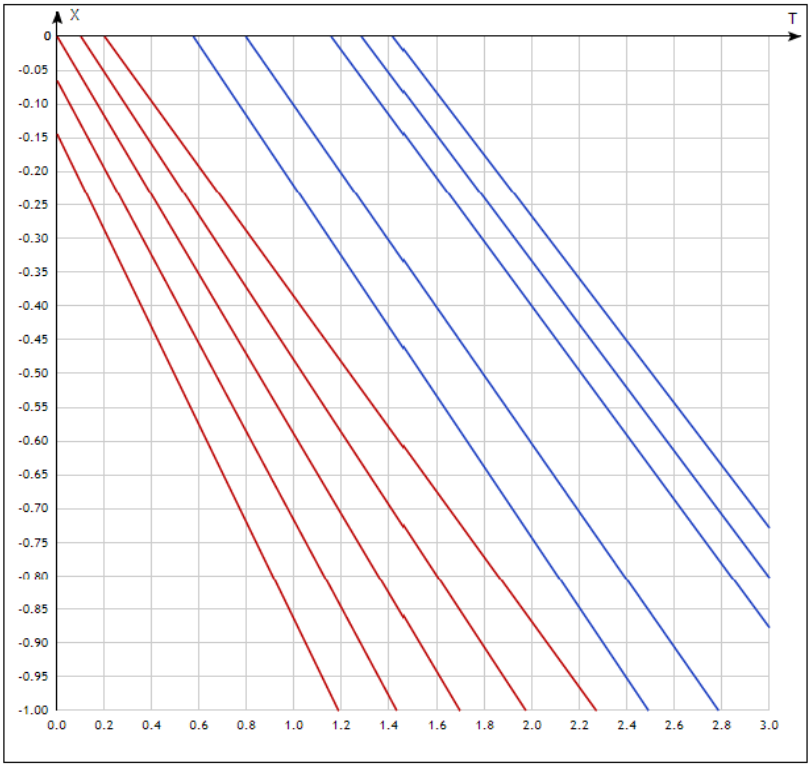
$$-u\,(t-t_0) = \; x - x_0$$

Отсюда можно получить два вида характеристик:

$t_0 = 0:$ $\quad x = -(2 - \frac{4}{\pi}\arctan(x_0+2))\ t$ (обозначены синим цветом на графике)

$x_0 = 0:$ $\quad x = -(t-t_0)(2 - \dfrac{4}{\pi}\arctan 2)e^{-t_0}$ (обозначены красным цветом на графике)

Построим соответствующие графики для $-1 \leqslant x < 0$



3.Метод решения.

Преобразуем исходное уравнение, написав его в следующей форме:

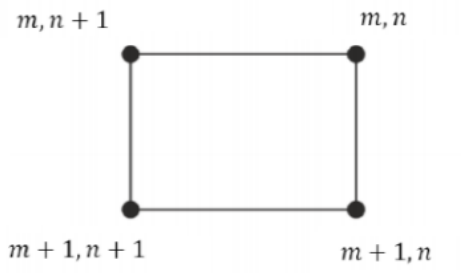
$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}(\frac{-u^2}{2}) = 0$$

Введем разностную сетку:

$$\omega = \{x_i = ih_x,\ i = 0, \dots, N,\ h_x = -\frac{1}{N};\ t_j = j\tau\},$$

N - число узлов вдоль оси x, τ - временной шаг

Для рассматриваемой задачи будем использовать четырехточечный шаблон. Он устойчив и аппроксимирует задачу как $O(h_x^2+h_y^2)$.



Разностная схема имеет вид:

$$\frac{y_n^{m+1} - y_n^m + y_{n+1}^{m+1} - y_{n+1}^m}{2h_t} + \frac{(y_n^{m+1})^2 - (y_n^m)^2 + (y_{n+1}^{m+1})^2 - (y_{n+1}^m)^2}{2 * 2h_x} = 0$$

Граничные условия имеют вид:

$$\begin{cases} y_n^0 = 2 - \frac{4}{\pi} arctg(-nh_x + 2) \\ y_0^m = (2 - \frac{4}{\pi} arctg(2))e^{-mh_t} \end{cases}$$

Данную задачу будем решать при помощи схемы бегущего счета. Значение сеточной функции $y_{m+1, n+1}$ неизвестно, но нам известны все значения, соответствующие начальному ($y_{0, n}$) и граничному ($y_{m, 0}$) условиям. Таким образом, зная значения в трех соседних точках: $y_{0,1}$, $y_{1,0}$, $y_{0,0}$ мы можем численно найти значение в четвертой точке $y_{1,1}$. Зная это значение, мы можем найти по трем известным точкам либо $y_{2,1}$, либо $y_{1,2}$. И так далее, заполняя найденными значениями сетку.

Это неявное уравнение относительно $y_{m+1, n+1}$. Будем решать его итерационным методом Ньютона.

$$y_{n+1}^{m+1(s+1)} = y_{n+1}^{m+1(s)} - \frac{f(y_{n+1}^{m+1(s)})}{f'(y_{n+1}^{m+1(s)})}$$

До достижения заданной точности ϵ:

$$|y_{n+1}^{m+1(s+1)} - y_{n+1}^{m+1(s)}| \leq \epsilon$$

4.Программа.

Компьютерная программа написана на языке С. Решение представлено на отрезке времени $0 \leqslant t \leqslant 3.$

Библиотеки:

```
#include <stdio.h>
#include <malloc.h>
#include <math.h>
#include <string.h>
#include <process.h>
#include <time.h>
```

Глобальные переменные (eps – заданная точность, h, tau – шаги по времени и по координате)

```
double eps = 0.01;
double h, tau;
```

Функция solve отвечает за реализацию итерационного метода Ньютона. Итерация происходит до тех пор, пока разность нового и старого (`y0 - y`) значения не будет меньше наперёд заданной точности (`eps`).

```
double solve(double y1, double y2, double y3,int i,int j)
{
	double y0 = y3;
	double y, f, dfdx, du;
	double d = eps + 1;
	while (d > eps)
	{
		y = y0;
		f =  (y*y) / (4 * h) + y/ (2 * tau) + (y3 - y1 - y2) / (2 * tau) +
			(y2*y2 - y1 * y1 - y3 * y3) / (4 * h);
		dfdx = y / (2 * h) + 1 / (2 * tau);
		y0 = y - f / dfdx;
		du = fabs(y0 - y);
		d = du;
	}
	return y0;
}
```

Создание всех необходимых переменных, файла куда будут записаны значения, создание двумерного массива, который и предстоит заполнить найденными значениями.

```
int main(void)
{

	FILE *fp1;
	fp1 = fopen("result.out", "w");
	double **a,*x,*t;
	int i, j, n=100, m=1000,NN=1000;
	double X = -1, T = 3;
	const double PI = 3.141592653589793;
	a = (double**)malloc(n * sizeof(double*));
	for (i = 0; i < n; i++)  // цикл по строкам
		a[i] = (double*)malloc(m * sizeof(double));
	x = (double*)malloc(NN * sizeof(double));
	t = (double*)malloc(NN * sizeof(double));
```

Присвоение шагам значений и заполнение координатно-временной сетки

```
	h = X / (1-n);
	tau = T / (m - 1);



	for (int j = 0; j < m; j++)
	{
		t[j] = T * j / m;
	}

	for (int k = 0; k < n; k++)
	{
		x[k] = (X * k) / n;
	}
```

Заполнение значений исходя из граничных условий

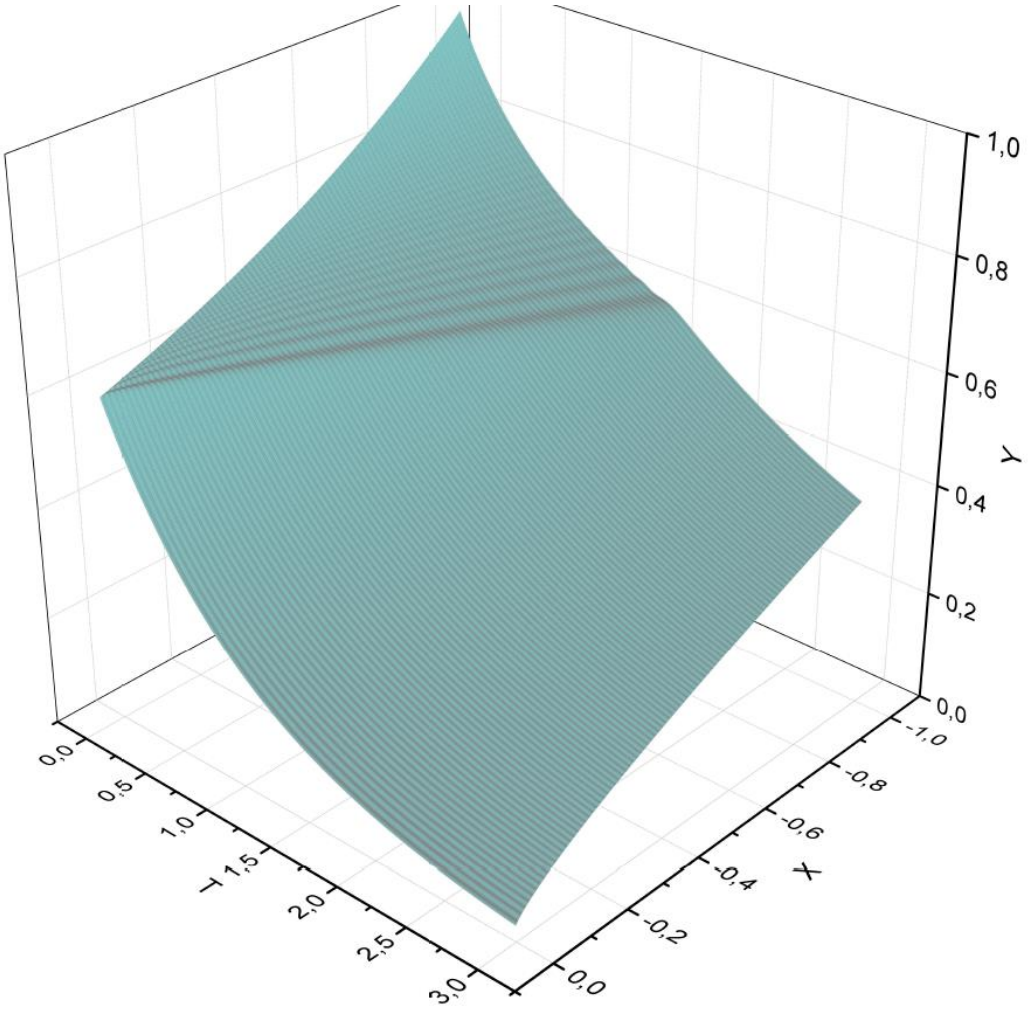
```
	for (i = 0; i < n; i++)
			{
				for (j = 0; j < m; j++)
				{
					if ((i == 0) && (j == 0))
						a[i][j] = (2 - 4 * atan(2) / PI);
					else
					{
						if (i == 0)
						a[i][j] = (2 -4*atan(2)/PI)*exp(-t[j]);
						if (j == 0)
						a[i][j] = (2- 4 * atan(x[i]+2) / PI);
					}
				}
			}
```

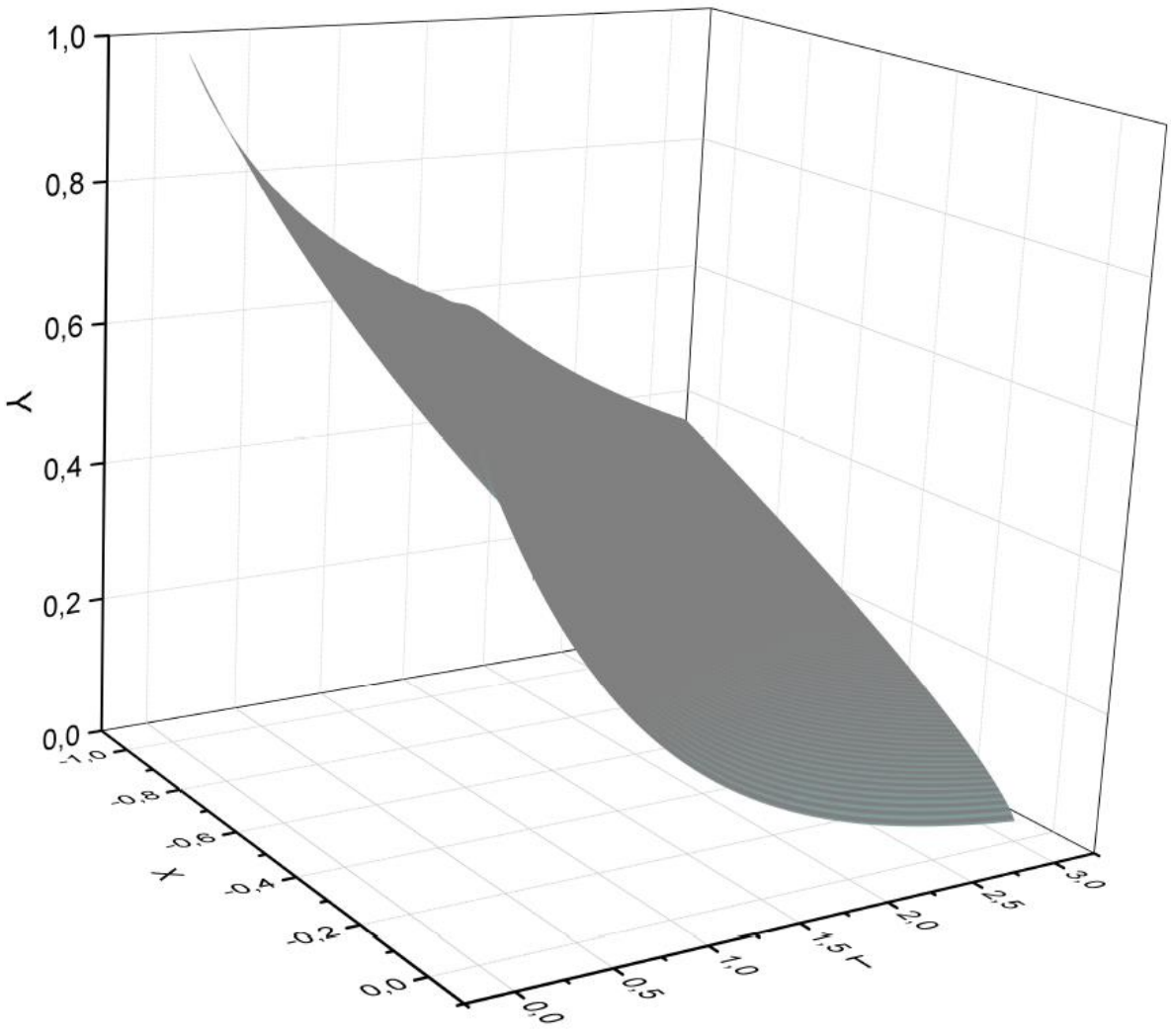
Заполнение значений в каждом узле сетки с помощью функции solve и вывод результата в файл

```
        for (i = 0; i < n-1; i++)
        {
                for (j = 0; j < m-1; j++)
                {
                        a[i+1][j+1] = solve(a[i][j], a[i+1][j], a[i][j+1], i, j);
                        fprintf(fp1, "%lf %lf %lf\n", x[i], t[j] , a[i][j]);
                }
    }
        fclose(fp1);

}
```

5.Графики.




1,0
0,8
0,6
0,4
0,2
0,0
Y
-1,0
-0,8
-0,6
-0,4
-0,2
0,0
X
0,0
0,5
1,0
1,5
2,0
2,5
3,0
T