Постановка задачи

Используя схему бегущего счета и итерационные методы, решить задачу для квазилинейного уравнения переноса:

$$\begin{cases} \dfrac{\partial u}{\partial t} + \dfrac{1}{1+u^2}\dfrac{\partial u}{\partial x} = 0, \ \ 0 < x \leq 1 \\ u(x,0) = x \\ u(0,t) = 0 \end{cases}$$

Исследование характеристик ¶

Посмотрим как будут вести себя проекции характеристик в заданных областях. Уравнение характеристик будет иметь вид: $dt = (1+u^2)dx$

Преобразуем его:

$\int_{t_o}^{t} dt = \int_{x_o}^{x} (1+u^2)dx$

$t = (1+u_0^2)(x-x_0)+t_0$

Воспользуемся начальным и граничным условиями для получения двух семейств кривых:

1) $t_0 = 0, u_0 = x_0: \quad t = (1+x_0^2)(x-x_0)$

2) $x_0 = 0, u_0 = 0: \quad t = x + t_0$

Импортируем необходимые библиотеки:

In [1]:

```
from math import *
import matplotlib.pyplot as plt
import numpy as np
from mpl_toolkits.mplot3d import axes3d
```

Определим функции характеристик через массивы, где итерирование будет идти по соответствующему неизвестному параметру: x_0 или t_0 . Здесь x_0 и t_0 взяты с определенными шагами с помощью np.arange.

In [2]:

```
def ch1(x):
    return [((1+x0*x0)*(x-x0)) for x0 in np.arange(0,1.1,0.1)]

def ch2(x):
    return [(x+t0) for t0 in np.arange(0,1.1,0.1)]
```

Создадим массив значений по x от 0 до 1 с определенным шагом и соответсвущие массивы для функций с итерированием уже по x.

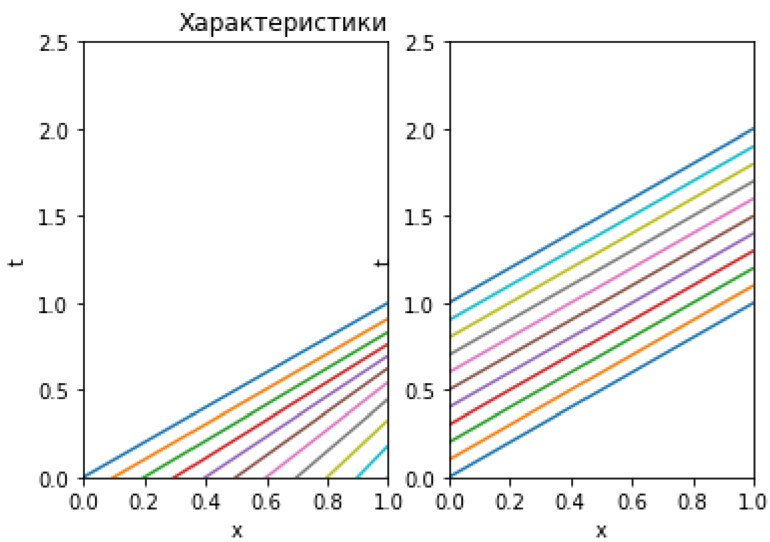
In [3]:

```
x_list = np.arange(0,1.1,0.1)

ch1_list = [ch1(x) for x in x_list]
ch2_list = [ch2(x) for x in x_list]
```

Построим соответствующие графики.

In [4]:

```
plt.subplot(1, 2, 1)
plt.ylim(0,2.5)
plt.xlim(0,1)
plt.plot(x_list, ch1_list)
plt.title('Характеристики',loc='right')
plt.ylabel('t')
plt.xlabel('x')
plt.subplot(1, 2, 2)
plt.ylim(0,2.5)
plt.xlim(0,1)
plt.plot(x_list, ch2_list)
plt.ylabel('t')
plt.xlabel('x')

plt.show()
```



Видно, что в заданной области характеристики не пересекаются. Следовательно, нет так называемого опрокидывания волны, и во всей области решение будет представимо через разностную схему.

Метод решения

Перепишем исходное уравнение, приведя его к дивергентному виду:

$$\frac{\partial u}{\partial t} + \frac{\partial(arctg(u))}{\partial x} = 0$$

Сетка

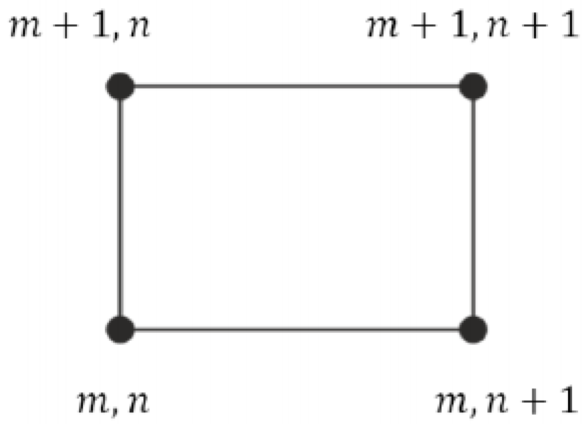
Введем в области $\Omega = \{(x,t) : 0 < x \leq 1,\ \ 0 < t < T\}$ сетку с шагом h_x по x и шагом h_t по t:

$$\omega_{h_x,\,h_t} = \begin{cases} x_n = n \cdot h_x,\ \ h_x = \frac{1}{N},\ \ n = \overline{0,N} \\ t_m = m \cdot h_t,\ \ h_t = \frac{1}{M},\ \ m = \overline{0,M} \end{cases} \qquad (1)$$

На $\omega_{h_x,\,h_t}$ будем рассматривать сеточную функцию $y_n^m = u(x_n, t_m)$

Схема бегущего счета. Шаблон

Для численного решения задачи будем использовать четырехточечный шаблон - "ящик". Он безусловно устойчив и аппроксимирует задачу как $O({h_x}^2 + {h_t}^2)$.



Данную задачу будем решать при помощи схемы бегущего счета. Значение сеточной функции y_{n+1}^{m+1} неизвестно, но нам известны все значения, соответствующие начальному(y_n^0) и граничному(y_0^m) условиям. Таким образом, зная значения в трех соседних точках: y_0^0, y_1^0, y_0^1, мы можем численно найти значение в четвертой точке y_1^1. Зная это значение, мы можем найти по трем известным точкам либо y_1^2, либо y_2^1. И так далее, заполняя найденными значениями сетку.

Таким образом, разностная схема задачи имеет вид:

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

Это неявное уравнение относительно y_{n+1}^{m+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:

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

Начальное и граничное условия примут вид:

$$\begin{cases} y_n^0 = x \\ y_0^m = 0 \end{cases}$$

Устойчивость

Критерий Неймана (необходимый)

Зафиксируем коэффициент перед $\frac{\partial u}{\partial x}$. Выберем произвольную точку (x_0, t_0) исследуемой области Ω и обозначим $\frac{1}{1+u^2(x_o, t_o)}$ за C. Теперь исследуемая схема приобретет вид:

$$\frac{U_{n+1}^{m+1} - U_{n+1}^{m} + U_{n}^{m+1} - U_{n}^{m}}{2h_t} + C\frac{U_{n+1}^{m+1} - U_{n}^{m+1} + U_{n+1}^{m} - U_{n}^{m}}{2h_x} = 0$$

Будем искать решение данного уравнения в виде $U_n^m = \lambda^m e^{i\alpha n}$. Подставив замену в уравнение, получим:

$$\lambda e^{i\alpha} - e^{i\alpha} + \lambda - 1 + \frac{Ch_t}{h_x}(\lambda e^{i\alpha} - \lambda + e^{i\alpha} - 1) = 0$$

Тогда для λ получим:

$$\lambda = \frac{e^{i\alpha} + 1 + \frac{Ch_t}{h_x}(1 - e^{i\alpha})}{e^{i\alpha} + 1 + \frac{Ch_t}{h_x}(e^{i\alpha} - 1)}$$

$$|\lambda| = 1$$

Из данного соотношения получаем, что условие $|\lambda(\alpha)| \leq 1$ справедливо для любых соотношений шагов по координате и времени, и, следовательно, спектральный критерий Неймана выполнен.

Критерий Куранта (достаточный)

Перепишем исследуемую разностную схему, поставив для нее задачу в виде:

$$\begin{cases} \frac{U_{n+1}^{m+1} - U_{n+1}^{m} + U_{n}^{m+1} - U_{n}^{m}}{2h_t} + C\frac{U_{n+1}^{m+1} - U_{n}^{m+1} + U_{n+1}^{m} - U_{n}^{m}}{2h_x} = \epsilon_n^m \\ U_n^0 = \phi_n \\ U_0^m = \mu^m \end{cases}$$

Преобразуем уравнение к виду:

$$U_{n+1}^{m+1}(1+\frac{Ch_t}{h_x})+U_n^{m+1}(1-\frac{Ch_t}{h_x})=U_{n+1}^{m}(1-\frac{Ch_t}{h_x})+U_n^{m}(1+\frac{Ch_t}{h_x})+2h_t\epsilon_n^m\ ,$$
где ϵ_n^m - некоторое возмущение исходной схемы.

Оценим данное соотношение по равномерной норме:

$$\|U^{m+1}\|(1+\frac{Ch_t}{h_x})+\|U^{m+1}\|(1-\frac{Ch_t}{h_x})\le\|U^{m}\|(1-\frac{Ch_t}{h_x})+\|U^{m}\|(1+\frac{Ch_t}{h_x})+2h_t\|\epsilon^m\|$$

$$2\|U^{m+1}\|\le 2\|U^m\|+2h_t\|\epsilon\|$$

Тогда по индукции получаем:

$$\|U^m\|\le\|\phi\|+mh_t|\epsilon\|$$
$$\|U^m\|\le\|\phi\|+T\|\epsilon\|$$

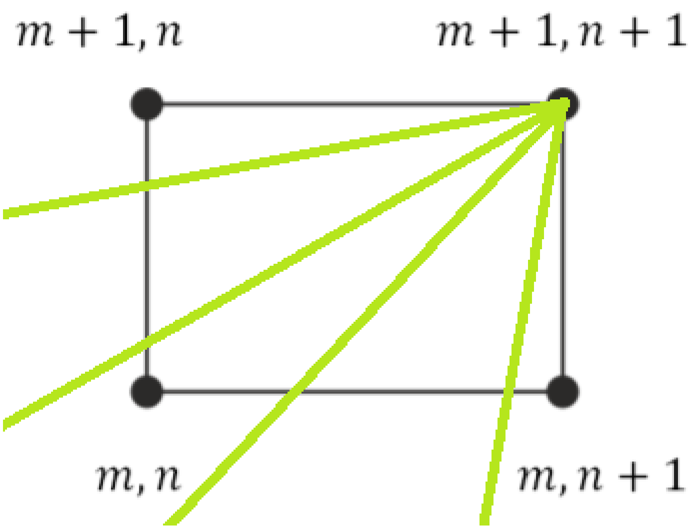
Переобозначая:

$$\|U^m\|\le\ M|\phi\|+N\|\epsilon\|\ ,$$
где T - величина интервала времени, на котором мы ищем решение, а M и N - константы, не зависящие от шагов сетки.

Таким образом получаем, что и критерий Куранта выполнен для рассматриваемой схемы для любых соотношений шагов по времени и координате, то есть фактически выполнено определение устойчивости.

Геометрический критерий

Проведём характеристику через точку (x_{n+1},t_{m+1}), в которой ищется решение. Если характеристика пересекает отрезок, соединяющий точки шаблона, в которых решение известно, то схема устойчива, иначе неустойчива.



Видно, что любая характеристика, проходящая через эту точку, пересекает отрезки шаблона, значения функции на которых известны. Критерий выполнен. Схема устойчива безусловно.

Порядок аппроксимации

Вычислим порядок аппроксимации. Для этого разложим значения функции U в узлах сетки в ряд до члена третьего порядка включительно в точке $(x_n + \frac{h_x}{2}; t_m + \frac{h_t}{2})$:

$$
\begin{aligned}
U^{m+1} &= U^{m+0.5} + \frac{h_t}{2}(U^{m+0.5}) + \frac{1}{2}\frac{h_t^2}{4}(U^{m+0.5})'' + \frac{1}{6}\frac{h_t^3}{8}(U^{m+0.5})''' + O(h_t^4) \\
U^{m} &= U^{m+0.5} - \frac{h_t}{2}(U^{m+0.5})' + \frac{1}{2}\frac{h_t^2}{4}(U^{m+0.5})'' - \frac{1}{6}\frac{h_t^3}{8}(U^{m+0.5})''' + O(h_t^4) \\
U_{n+1} &= U_{n+0.5} + \frac{h_x}{2}U'_{n+0.5} + \frac{1}{2}\frac{h_x^2}{4}U''_{n+0.5} + \frac{1}{6}\frac{h_x^3}{8}U'''_{n+0.5} + O(h_x^4) \\
U_{n} &= U_{n+0.5} - \frac{h_x}{2}U'_{n+0.5} + \frac{1}{2}\frac{h_x^2}{4}U''_{n+0.5} - \frac{1}{6}\frac{h_x^3}{8}U'''_{n+0.5} + O(h_x^4)
\end{aligned}
$$

Из данных соотношений получим:

$$
\frac{U_{n+1}^{m+1} - U_{n+1}^{m} + U_{n}^{m+1} - U_{n}^{m}}{2h_t} + C\frac{U_{n+1}^{m+1} - U_{n}^{m+1} + U_{n+1}^{m} - U_{n}^{m}}{2h_x} - \frac{\partial U}{\partial t} - C\frac{\partial U}{\partial x} = O(h_t^2 + h_x^2
$$

Аналитическое решение

В области $x \le t$ функция $u(x,t)$ остается равной 0, а в области $x > t$ решение $u(x,t)$ определяется начальным условием. Выберем произвольную точку (x,t) в области $t < x$ и выразим через x и t координату x_0 пересечения проекции характеристики с осью Ox, на которой задано начальное условие:

$$t = (x - x_0)(1 + x_0^2)$$

Перепишем в виде:

$$x_0^3 - x x_0^2 + x_0 - x + t = 0$$

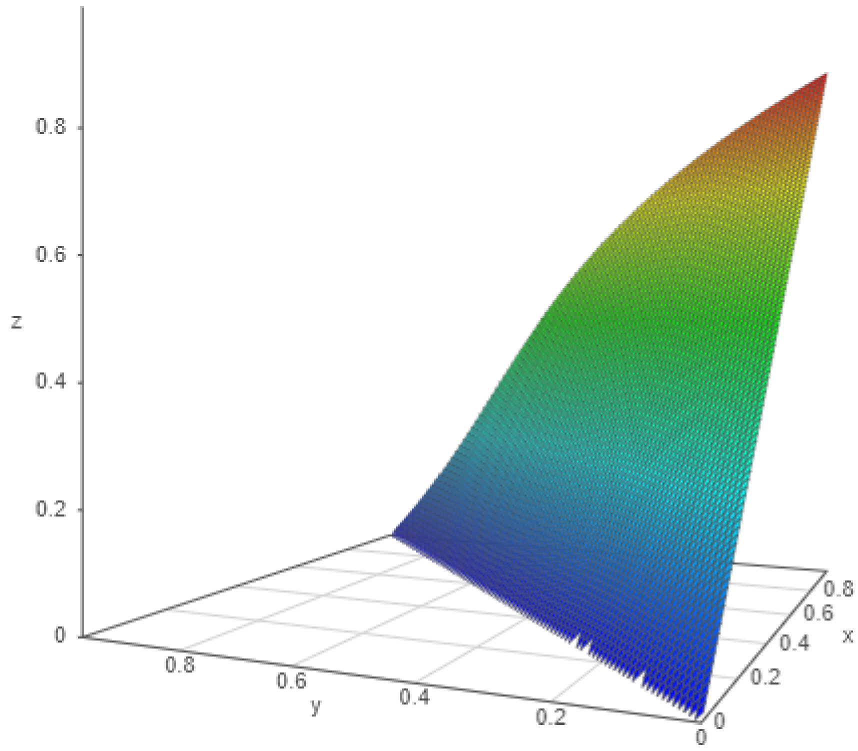
При $x > t$ решение имеет вид:

$$u(x,t) = x_0$$

Найдём действительный корень уравнения:

$$
u = \frac{\sqrt[3]{3\sqrt{3}\,\sqrt{27t^2 - 4tx^3 - 36tx + 4x^4 + 8x^2 + 4} - 27t + 2x^3 + 18x}}{3\sqrt[3]{2}} - \frac{\sqrt[3]{2}\,(3 - x^2)}{3\sqrt[3]{3\sqrt{3}\,\sqrt{27t^2 - 4tx^3 - 36tx + 4x^4 + 8x^2 + 4} - 27t + 2x^3 + 18x}} + \frac{x}{3}
$$

График аналитического решения в области $x > t$:



Численное решение

Зададим: ϵ - точность в методе Ньютона, N - количество шагов по x, M - количество шагов по y, а также границы нашей сетки.

In [5]:

```
epsilon = 0.00001
N = 100; M = 100
T_begin = 0; T_end = 1
X_begin = 0; X_end = 1
```

Соответственно, элементарные шаги.

In [6]:

```
h_x=(X_end - X_begin)/(N-1)
h_t=(T_end - T_begin)/(M-1)
```

Создадим двумерный массив размерами с нашу сетку($N \times M$), в ячейках которого будут храниться соответствующие искомые значения.

In [7]:

```
y=np.zeros((M,N))
```

Начнем заполнять его начальным и граничным значениями.

In [8]:

```
for n in np.arange(N):
        y[0][n] = h_x*n
for m in np.arange(M):
        y[m][0] = 0
```

Определим вспомогательные функции.

In [9]:

```
def F(m,n):
    return atan(y[m][n])

def df(mp1,np1):
    return (0.5/h_t + 0.5/(h_x*((y[mp1][np1])*(y[mp1][np1])+1)))
```

Разностная схема будет иметь вид.

In [10]:

```
def f(mp1, np1):
    n = np1-1
    m = mp1-1
    return (y[mp1][n]-y[m][n] + y[mp1][np1]-y[m][np1]) / (2.*h_t) + (F(mp1, np1)-F(mp1
```

Перейдем к методу Ньютона, пробегая по всей сетке.

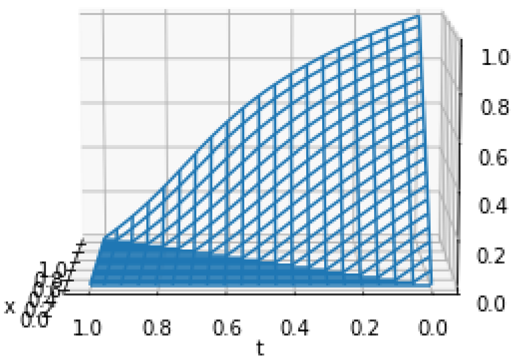
In [11]:

```
eps = epsilon + 1;
while eps > epsilon:
    eps = 0
    for m in np.arange(M)[0:M-1]:
        for n in np.arange(N)[0:N-1]:
            ep = f(m+1, n+1) / df( m+1, n+1)
            y[m+1][n+1] = y[m+1][n+1] - ep
            if abs(ep) > eps:
                eps = abs(ep)
```

Построим график решения.

In [12]:

```
tm = np.linspace(T_begin,T_end, num=M)
xn = np.linspace(X_begin, X_end, num=N)

X, T = np.meshgrid(xn, tm)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
surf1 = ax.plot_wireframe(X, T, y, rstride=5,cstride=5)
plt.xlabel('x')
plt.ylabel('t')
ax.view_init(10,180)
plt.show()
```



Погрешности

Рассчитаем велиины погрешностей в области $x > t$, вычитая аналитическое решение из численного.

In [15]:

```
MM=M//8
NN=N//4
def fan(m,n):
    return ((2*(xn[n])**3 + 3*sqrt(3)*sqrt(4*(xn[n])**4 - 4*(xn[n])**3*(tm[m]) + 8*(xn
    + 27*(tm[m])**2 + 4) + 18*(xn[n]) - 27*(tm[m]))**(1/3)/(3*2**(1/3)) - (2**(1/3)*(3
    3*sqrt(3)*sqrt(4*(xn[n])**4 - 4*(xn[n])**3*(tm[m]) + 8*(xn[n])**2 - 36*(tm[m])*(xn
    + 27*(tm[m])**2+4)+18*(xn[n])-27*(tm[m]))**(1/3))+(xn[n])/3)
```

Информация о точности решения приведена в таблице:

M,N	error
10	0.000399
20	0.000072
40	0.000024
60	0.0000094

M,N	error
80	0.0000058
100	0.0000035

Построим график зависимости погрешности от величины шага:

In [14]:

```
MN=[10,20,40,60,80,100]
error=[0.000399,0.000072,0.000024,0.0000094,0.0000058,0.0000035]
plt.plot(MN,error,'bd')
plt.ylabel('error')
plt.xlabel('M,N')
plt.grid(True)
```