

## Лабораторная работа №5.

### Практическая часть.

Формат А.А.	Лабораторная работа №5	Зачет.
	Перенос примесей	
Тр 21-781	в атмосфере.	

Цель работы: численно смоделировать динамику ~~загрязнения~~ <sup>загрязнения</sup> примесей в приземном слое атмосферы.  
Теоретическая часть.

Уравнение диффузии:

$$\frac{dp}{dt} = - \frac{d}{dx} (C_x p) - \frac{d}{dy} (C_y p) + \frac{d}{dx} \left( D \frac{dp}{dx} \right) + \frac{d}{dy} \left( D \frac{dp}{dy} \right) + Q - \lambda p$$

$C_x$  и  $C_y$  - скорости ветра в разных направлениях.

Все слагаемые интегрирование по пространству дискретизированы в пространстве расчетной сеткой с неравномерными ячейками со сторонами в плоской декартовой системе координат  $\Delta x$  и  $\Delta y$ .

Явное разностное уравнение имеет вид:

$p$  в декартовой системе координат  
( $x, y$ ) для ячейки  $(i, j)$ :

$$\begin{aligned} p_{i,j}^{n+1} = & p_{i,j}^n - \frac{1}{\Delta x \Delta y} (\Delta M_{i+1/2,j}^n - \Delta M_{i-1/2,j}^n + \\ & + \Delta M_{i,j+1/2}^n - \Delta M_{i,j-1/2}^n) + \frac{\Delta t_n}{(\Delta x)^2} [p_{i+1,j}^n \cdot \\ & \cdot (p_{i,j}^n - p_{i-1,j}^n) - D_{i-1/2,j}^n (p_{i,j}^n - p_{i-1,j}^n)] + \\ & + \frac{\Delta t_n}{(\Delta y)^2} [D_{i,j+1/2}^n (p_{i,j}^n - p_{i,j-1}^n) - D_{i,j-1/2}^n \cdot \\ & \cdot (p_{i,j}^n - p_{i,j-1}^n)] + \Delta t_n (Q_{i,j}^n - 2p_{i,j}^n). \end{aligned}$$

Источники загрязнения определяются  
функцией  $Q$ , которая определяет времен-  
ную динамику их выбросов и их  
параметры.

$$D_{i \pm \frac{1}{2}}^n = D(|\bar{C}|_{i \pm \frac{1}{2}}^n), \quad |\bar{C}|_{i \pm \frac{1}{2}}^n = \frac{|\bar{C}|_{i,j}^n + |\bar{C}|_{i \pm 1,j}^n}{2}$$

$$D_{i,j \pm \frac{1}{2}}^n = D(|\bar{C}|_{i,j \pm \frac{1}{2}}^n), \quad |\bar{C}|_{i,j \pm \frac{1}{2}}^n = \frac{|\bar{C}|_{i,j}^n + |\bar{C}|_{i,j \pm 1}^n}{2}$$

$$\Delta M_{i+\frac{1}{2},j}^n = \Delta y \Delta t_n \begin{cases} p_{i,j}^n u_{i+\frac{1}{2},j}^n, & \text{если } u_{i+\frac{1}{2},j}^n > 0 \\ p_{i+1,j}^n u_{i+\frac{1}{2},j}^n, & \text{если } u_{i+\frac{1}{2},j}^n < 0 \end{cases}$$

$$\Delta M_{i-\frac{1}{2},j}^n = \Delta y \Delta t_n \begin{cases} p_{i,j}^n u_{i-\frac{1}{2},j}^n, & \text{если } u_{i-\frac{1}{2},j}^n < 0 \\ p_{i-1,j}^n u_{i-\frac{1}{2},j}^n, & \text{если } u_{i-\frac{1}{2},j}^n > 0 \end{cases}$$

$$\Delta M_{i,j+\frac{1}{2}}^n = \Delta x \Delta t \begin{cases} p_{i,j}^n v_{i,j+\frac{1}{2}}^n, & \text{если } v_{i,j+\frac{1}{2}}^n > 0 \\ p_{i,j+1}^n v_{i,j+\frac{1}{2}}^n, & \text{если } v_{i,j+\frac{1}{2}}^n < 0 \end{cases}$$

$$\Delta M_{i,j-\frac{1}{2}}^n = \Delta x \Delta t \begin{cases} p_{i,j}^n v_{i,j-\frac{1}{2}}^n, & \text{если } v_{i,j-\frac{1}{2}}^n < 0 \\ p_{i,j-1}^n v_{i,j-\frac{1}{2}}^n, & \text{если } v_{i,j-\frac{1}{2}}^n > 0 \end{cases}$$

где  $\Delta t$  — время глобальной устойчивости:

$$\Delta t \leq \frac{|\Delta \tilde{r}|^2}{2D} + \frac{|\Delta \tilde{r}|}{|E|}$$

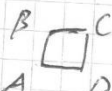
Крайние условия 1-го рода — заданные концентрации на границе одновременно

$$p(t, \tilde{r})|_S = \varphi(t)$$

Крайние условия 2-го рода — заданные

~~гидродинамические~~ гидродинамические условия на границе  $\frac{dp}{d\tilde{r}}|_S = \varphi(t)$

Крайние условия 2-го рода:



$$AB \quad p_{0,t}^{n+1} = p_{i,j}^{n+1} - \varphi \Delta x \quad CD \quad p_{n+1}^{n+1} = \varphi^{n+1} \Delta x - p_{i,j}^{n+1}$$

$$AD \quad p_{i,0}^{n+1} = p_{i,1}^{n+1} - \varphi^{n+1} \Delta y \quad BC \quad p_{1,n+1}^{n+1} = \varphi^{n+1} \Delta y - p_{i,m}^{n+1}$$

Роль: в течение многолетних  
динамиче запыляемости прилетел  
в приземной слое атмосферы.

## Практическая часть.

Код программы:

```
#include <iostream>
#include <locale>
#include <string>
#include <fstream>
#include <Windows.h>
#include <cmath>
#include <iomanip>
#define PI 3.1415926535
using namespace std;

//Функция записи в файл для визуализации
void In_VTK_file(double **P, int N, double dx, string filename)
{
    ofstream file(filename);
    //Шапка VTK файла
    file << "# vtk DataFile Version 5.0" << endl;
    file << "Example 3D regular grid VTK file." << endl;
    file << "ASCII" << endl;
    file << "DATASET UNSTRUCTURED_GRID" << endl;
    file << "POINTS " << (N + 1) * (N + 1) << " float" << endl;
    //Запись точек в файл
    for (int i = 0; i <= N; i++)
    {
        float x = i * dx;
        for (int j = 0; j <= N; j++)
        {
            float y = j * dx;
            file << x << " " << y << " 0" << endl;
        }
    }
    //Шапка для данных о точках
    file << "POINT_DATA " << (N + 1) * (N + 1) << endl;
    file << "SCALARS P float" << endl;
    file << "LOOKUP_TABLE default" << endl;
    //Запись значений в каждой точке
    for (int i = 0; i <= N; i++)
    {
        for (int j = 0; j <= N; j++)
        {
            file << fixed << setprecision(4) << P[i][j] << endl;
        }
    }
    file.close();
}

//Источник выброса
double Q(double x, double y)
{
    if (x == 100 && y == 50)
        return 2000; //Мощность выброса
    else
        return 0;
}

int main()
{
    setlocale(LC_ALL, "Rus");
    double dx = 10; //Шаг по x
    double dt; //Шаг по времени
    double t = 0; //Начальное время
    double t_end = 500; //Конечное время
```

```

double x_last = 3000;
double C = 10; //Ветер
double D = 100; //Диффузия
double alpha = 45.0 * PI / 180.0; //Угол направления ветра
double M1, M2, M3, M4;
M1 = M2 = M3 = M4 = 0;
double K1, K2, K3, K4, K5;
double k = 0.05; //Коэффициент Куранта
double Cx = C * cos(alpha);
double Cy = C * sin(alpha);
int i, j;
int N = x_last / dx;
//Плотность примеси в текущий момент времени
double** P = new double* [N + 2.0];
for (int i = 0; i < N + 2.0; i++)
{
    P[i] = new double[N + 2.0];
}
//Плотность примеси в текущий момент времени
double** P_next = new double* [N + 2.0];
for (int i = 0; i < N + 2.0; i++)
{
    P_next[i] = new double[N + 2.0];
}
//Начальные условия
for (i = 0; i <= N + 1; i++)
{
    for (j = 0; j <= N + 1; j++)
    {
        P[i][j] = 0;
        P_next[i][j] = 0;
    }
}
int n = 0;
//Условие устойчивости
dt = k * min((dx * dx) / (2 * D) + dx / abs(Cx), (dx * dx) / (2 * D) + dx /
abs(Cy));
t += dt;
//Основные расчеты
do
{
    for (i = 1; i <= N; i++)
    {
        for (j = 1; j <= N; j++)
        {
            if (Cx > 0)
            {
                M1 = P[i][j] * Cx * dx * dt;
                M2 = P[i - 1][j] * Cx * dx * dt;
            }
            else if (Cx < 0)
            {
                M1 = P[i + 1][j] * Cx * dx * dt;
                M2 = P[i][j] * Cx * dx * dt;
            }
            if (Cy > 0)
            {
                M3 = P[i][j] * Cy * dx * dt;
                M4 = P[i][j - 1] * Cy * dx * dt;
            }
            else if (Cy < 0)
            {
                M3 = P[i][j + 1] * Cy * dx * dt;
                M4 = P[i][j] * Cy * dx * dt;
            }
        }
    }
}

```

```

        //Основная формула
        //Разбита на 5 частей, чтобы удобней было с ней работать
        K1 = P[i][j];
        K2 = 1.0 / (dx * dx) * (M1 - M2 + M3 - M4);
        K3 = (dt / (dx * dx)) * (D * (P[i + 1][j] - P[i][j]) - D *
(P[i][j] - P[i - 1][j]));
        K4 = (dt / (dx * dx)) * (D * (P[i][j + 1] - P[i][j]) - D *
(P[i][j] - P[i][j - 1]));
        K5 = Q(i * dx, j * dx) * dt;

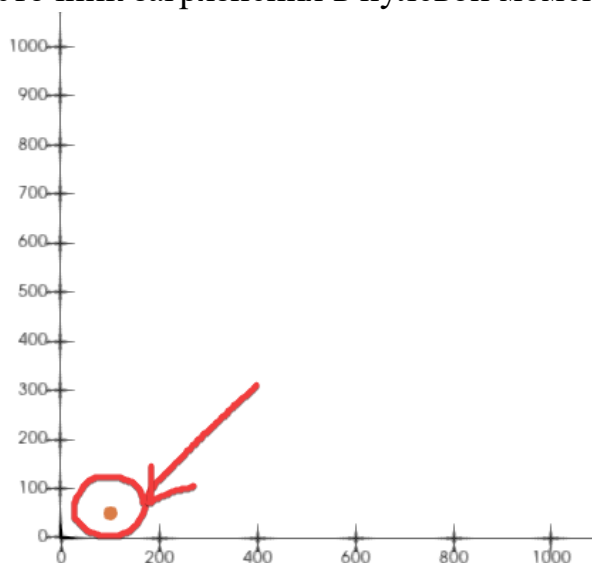
        P_next[i][j] = K1 - K2 + K3 + K4 + K5;
    }
}
//Граничные условия
for (j = 0; j <= N + 1; j++)
{
    P_next[0][j] = P_next[1][j];
    P_next[N + 1][j] = P_next[N][j];
}
for (i = 0; i <= N + 1; i++)
{
    P_next[i][0] = P_next[i][1];
    P_next[i][N + 1] = P_next[i][N];
}
//Меняем массивы местами
for (i = 0; i <= N + 1; i++)
{
    for (j = 0; j <= N + 1; j++)
    {
        P[i][j] = P_next[i][j];
    }
}
//Запись в файл
if (n % (int)(0.5 * N) == 0)
{
    auto s = to_string(n);
    //Строка названия файла с данными
    string filename = "test\\p" + s + ".vtk";
    In_VTK_file(P, N, dx, filename);
}
n++;
t += dt;
}
while (t <= t_end);
//Очистка памяти
for (int i = 0; i < N + 2.0; i++)
{
    delete[] P[i];
}
for (int i = 0; i < N + 2.0; i++)
{
    delete[] P_next[i];
}
system("pause");
return 0;
}

```

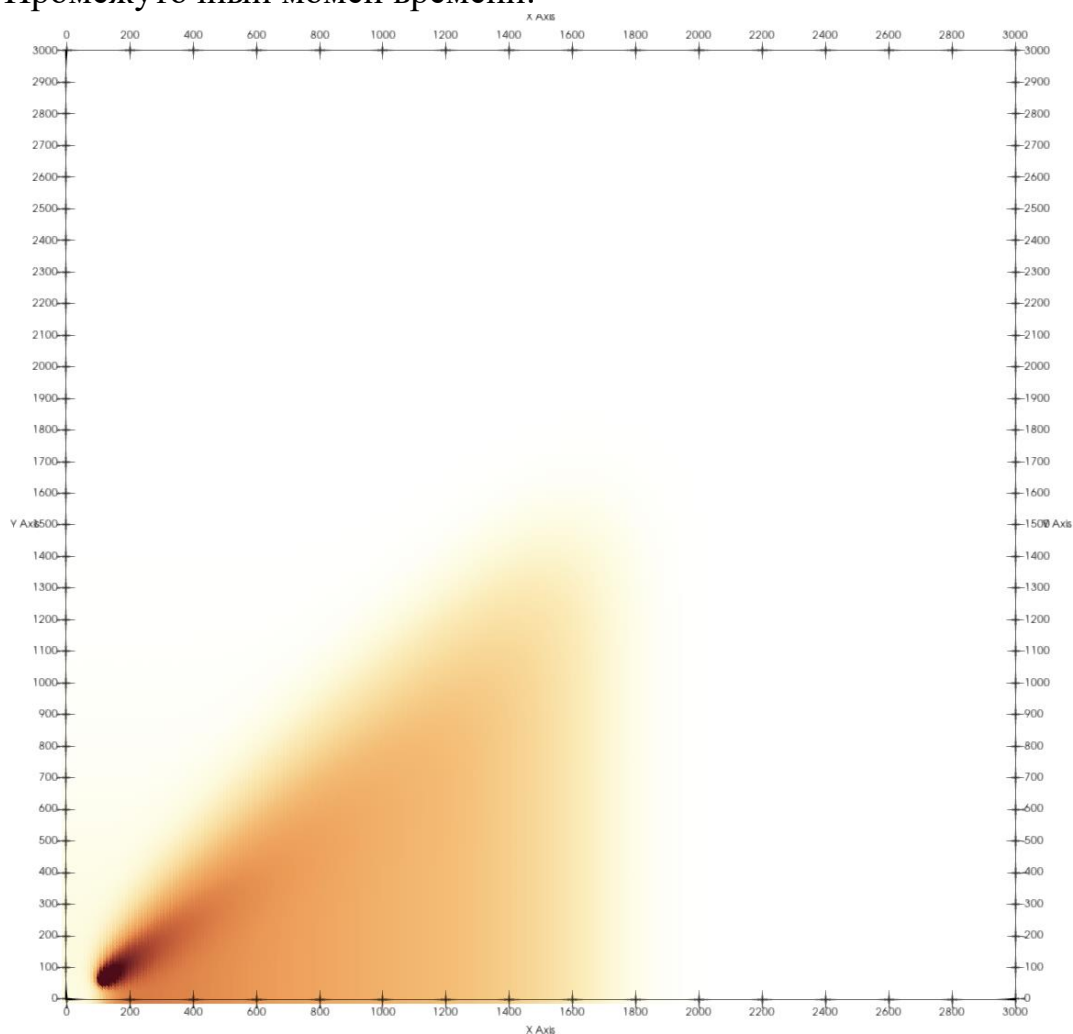
## Графики.

Была создана анимация переноса примесей в атмосфере.

Источник загрязнения в нулевой момент времени:



Промежуточный момент времени:





Конечный момент времени (500).

