Отчет по задаче практикума
«Уравнения с частными производными»

Тиунова Анастасия, 407 группа

10 апреля 2021 г.

1 Постановка задачи

Задача. Найти решение задачи (3.12)

$$\begin{cases} \frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} - \left(1 + x^2\right) u, \\ \alpha \in \{1.0; 0.01\}, \\ x = 0 : \ \frac{\partial u}{\partial x} = 0; \\ x = 1 : \ u = 1; \\ t = 0 : \ u = 1. \end{cases}$$

Воспользуемся методом четного продолжения относительно прямой $x = 0$ (такое допустимо т.к. в нуле $\frac{\partial u}{\partial x} = 0$), тогда задача сводится к:

$$\begin{cases} \frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} - \left(1 + x^2\right) u, \\ \alpha \in \{1.0; 0.01\}, \\ x = -1 : \ u = 1; \\ x = 1 : \ u = 1; \\ t = 0 : \ u = 1. \end{cases}$$

2 Метод решения

Будем использовать метод сеток. Приближенное решение задачи ищем в виде сеточной функции, т.е. функции, определенной в каждом узле сетки (N, M). Эта функция обозначается $\{u_m^n\}$.

Значение u_m^n будем трактовать как приближенное значение функции $u(t, x)$ в узле (t_n, x_m), т.е.

$$u_m^n \sim u(t_n, x_m).$$

Сеточную функцию получим как решение разностного уравнения. Принятый способ разностной аппроксимации называют *схемой*. Для решения нашей задачи будем использовать схему с весами, где вес $\delta \in [0, 1]$ будет выбран позднее:

$$\boxed{\frac{u_m^{n+1} - u_m^n}{\tau} = \alpha \left(\delta \frac{u_{m-1}^n - 2u_m^n + u_{m+1}^n}{h^2} + (1 - \delta) \frac{u_{m-1}^{n+1} - 2u_m^{n+1} + u_{m+1}^{n+1}}{h^2} \right) + \frac{f_m^n + f_m^{n+1}}{2}}.$$

Идея: последовательно выражать неизвестные сеточные функции из верхнего слоя через известные с нижних слоев с помощью разностного уравнения и граничных условий.

(I) Рассмотрим уравнение из условия

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} - \left(1 + x^2\right) u.$$

Запишем для него параметрическое семейство разностных схем с весом $\delta = \frac{1}{2}$:

$$\frac{u_m^{n+1} - u_m^n}{\tau} = \alpha \left(\frac{1}{2} \frac{u_{m-1}^n - 2u_m^n + u_{m+1}^n}{h^2} + (1 - \frac{1}{2}) \frac{u_{m-1}^{n+1} - 2u_m^{n+1} + u_{m+1}^{n+1}}{h^2} \right) - \left(1 + (hm)^2\right) \frac{u_m^n + u_m^{n+1}}{2} \tag{1}$$

(II) Изучим порядок аппроксимации в зависимости от веса δ:
Разложим имеющиеся функции в ряд Тейлора в окрестности точки (t_n, x_m):

$$u_m^{n+1} = u(t_n + \tau, x_m) = u(t_n, x_m) + \tau u_t(t_n, x_m) + \frac{1}{2}\tau^2 u_{tt}(t_n, x_m) + \frac{1}{6}\tau^3 u_{ttt}(t_n, x_m) + o(\tau^3)$$

$$u_{m\pm1}^n = u(t_n, x_m \pm h) = u(t_n, x_m) \pm h u_x(t_n, x_m) + \frac{1}{2}h^2 u_{xx}(t_n, x_m) \pm \frac{1}{6}h^3 u_{xxx}(t_n, x_m) + o(h^3)$$

$$\begin{aligned} u_{m\pm1}^{n+1} = u(t_n, x_m - h) = u(t_n, x_m) - h u_x(t_n, x_m) + \tau u_t(t_n, x_m) + \frac{1}{2}h^2 u_{xx}(t_n, x_m) + \\ + h\tau u_{xt}(t_n, x_m) + \frac{1}{2}\tau^2 u_{tt}(t_n, x_m) + \frac{1}{6} tau^3 u_{ttt}(t_n, x_m) \pm \frac{1}{2}\tau^2 h u_{ttx}(t_n, x_m) + \\ + \frac{1}{2}\tau h^2 u_{txx}(t_n, x_m) \pm \frac{1}{6}h^3 u_{xxx}(t_n, x_m) + o(h^3 + \tau^3) \end{aligned}$$

Подставим результаты в схему:

$$u_t + o(\tau^2) = \alpha \left(\frac{1}{2}(u_{xx} + o(h^2)) + (1 - \frac{1}{2})(u_{xx} + o(h^2)) \right) - (1 + x^2)u;$$

$$u_t = \alpha u_{xx} - (1 + x^2)u + o(h^2) + o(\tau^2);$$

(III) Изучим устойчивость схемы для $\delta = \frac{1}{2}$:
Варьированием (1) по u_m^n получаем линейное уравнение на Δu_m^n:

$$\begin{aligned} \frac{\Delta u_m^{n+1} - \Delta u_m^n}{\tau} = \alpha \left(\frac{1}{2} \frac{\Delta u_{m-1}^n - 2\Delta u_m^n + \Delta u_{m+1}^n}{h^2} + (1 - \frac{1}{2}) \frac{\Delta u_{m-1}^{n+1} - 2\Delta u_m^{n+1} + \Delta u_{m+1}^{n+1}}{h^2} \right) - \\ - (1 + (hm)^2) \frac{\Delta u_m^n + \Delta u_m^{n+1}}{2}. \end{aligned}$$

Замораживая коэффициенты в нем, приходим к линейному уравнению с постоянными коэффициентами, решение всегда имеет вид:

$$\Delta u_m^n = \lambda^n e^{ik\phi} u_0^0 \tag{2}$$

Подставим (3) в (2):

$$\frac{\lambda - 1}{\tau} = \alpha \left(\frac{1}{2} \frac{\cos(\phi) - 1}{h^2} + (1 - \frac{1}{2})\lambda \frac{\cos(\phi) - 1}{h^2} \right) - (1 + x^2);$$

$$\lambda = \frac{1 + \tau\alpha \frac{\cos(\phi) - 1}{h^2}}{1 - \tau\left(\alpha \frac{\cos(\phi) - 1}{h^2}\right)} - \frac{\tau(1 + x^2)}{1 - \tau\left(\alpha \frac{\cos(\phi) - 1}{h^2}\right)} \Rightarrow |\lambda| \leq 1 + o(\tau)$$

$\Rightarrow$ схема безусловно устойчива.

(IV) Из показанного выше, следует,что предложенная схема безусловно устойчива и имеет порядок аппроксимации $o(\tau^2+h^2)$.

$$\boxed{\frac{u_m^{n+1}-u_m^n}{\tau}=\alpha\left(\frac{1}{2}\frac{u_{m-1}^n-2u_m^n+u_{m+1}^n}{h^2}+(1-\frac{1}{2})\frac{u_{m-1}^{n+1}-2u_m^{n+1}+u_{m+1}^{n+1}}{h^2}\right)-\left(1+(hm)^2\right)\frac{u_m^n+u_m^{n+1}}{2}}.$$

Идея: составить систему линейных уравнений с трехдиагональной матрицей на неизвестные $u_{m-1}^{n+1},u_m^{n+1},u_{m+1}^{n+1}$ с коэффициентами и свободным членом, вычисленными с нижнего слоя с использованием граничных условий.
Из граничных условий:

1) $x=-1:\ u_m^n=1$
2) $x=1:\ u_m^n=1$
3) $t=0:\ u_m^n=1$

Тогда:

$$m=1:\ \ u_m^{n+1}\left(\frac{1}{\tau}-\frac{\alpha}{2h^2}+\frac{1}{2}(1+(hm)^2)\right)-\frac{\alpha}{2h^2}u_{m+1}^{n+1}=$$
$$=\frac{u_m^n}{\tau}+\frac{\alpha}{2}\frac{u_{m-1}^n-2u_m^n+u_{m+1}^n}{h^2}-(1+(hm)^2)\frac{u_m^n}{2}$$

$$m\in\{2,..,M-3\}:\ \ -\frac{\alpha}{2h^2}u_{m-1}^{n+1}+u_m^{n+1}\left(\frac{1}{\tau}+\frac{\alpha}{h^2}+\frac{1}{2}(1+(hm)^2)\right)-\frac{\alpha}{2h^2}u_{m+1}^{n+1}=$$
$$=\frac{u_m^n}{\tau}+\frac{\alpha}{2}\frac{u_{m-1}^n-2u_m^n+u_{m+1}^n}{h^2}-(1+(hm)^2)\frac{u_m^n}{2}$$

$$m=M-2:\ \ -\frac{\alpha}{2h^2}u_{m-1}^{n+1}+u_m^{n+1}\left(\frac{1}{\tau}+\frac{\alpha}{h^2}+\frac{1}{2}(1+(hm)^2)\right)=$$
$$=\frac{\alpha}{2h^2}+\frac{u_m^n}{\tau}+\frac{\alpha}{2}\frac{u_{m-1}^n-2u_m^n+u_{m+1}^n}{h^2}-(1+(hm)^2)\frac{u_m^n}{2}$$

3 Вычислительный эксперимент

Ниже представлено вычесленное значение в точке максимума и минимума производной potприразныxn,при $\alpha=1$

$t \backslash x$	1	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{1}{5}$
0.1	0.89823584	0.89406673	0.89654779	0.89786238	0.89598834
$t_{MaxD_t}0$	0.99981994	0.99984659	0.99986214	0.99986658	0.99985992
0.9	0.73141937	0.66655632	0.63197447	0.62251293	0.63677409

Таблица 1: $\tau=\frac{h^2}{3}, m=50$

$t \backslash x$	1	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{1}{5}$
0.1	0.89881706	0.89418373	0.89649535	0.89790958	0.89622441
$t_{MaxD_t}0$	0.99995483	0.99996149	0.99996544	0.99996666	0.99996517
0.9	0.73379455	0.66863746	0.63340792	0.62295348	0.63576145

Таблица 2: $\tau=\frac{h^2}{3}, m=100$

$t \backslash x$	1	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{1}{5}$
0.1	0.89896643	0.89420539	0.89647078	0.89791855	0.89629163
$t_{MaxD_t}0$	0.99997990	0.99998286	0.99998462	0.99998518	0.99998454
0.9	0.73448739	0.66924910	0.63381622	0.62304101	0.63537489

Таблица 3: $\tau = \frac{h^2}{3}, m = 150$

Изучим главный член аппроксимации в опорных точках, ниже представлены значения Δuв них:

$t \backslash x$	1	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{1}{5}$
0.1	-0.00058123	-0.00011700	0.00005245	-0.00004720	-0.00023607
t_{MaxD_t}	-0.00013489	-0.00011490	-0.00010330	-0.00010008	-0.00010524
0.9	-0.00237518	-0.00208114	-0.00143344	-0.00044055	0.00101264

Таблица 4: $\tau = \frac{h^2}{3}, \Delta u$ при $m = 50- > m = 100$

$t \backslash x$	1	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{1}{5}$
0.1	-0.00014937	-0.00002166	0.00002457	-0.00000897	-0.00006722
t_{MaxD_t}	-0.00002507	-0.00002137	-0.00001919	-0.00001852	-0.00001938
0.9	-0.00069283	-0.00061164	-0.00040831	-0.00008752	0.00038656

Таблица 5: $\tau = \frac{h^2}{3}, \Delta u$ при $m = 100- > m = 150$

Итого, для $\frac{u(m=50)-u(m=100)}{u(m=100)-u(m=150)}$ в опорных точках:

$t \backslash x$	1	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{1}{5}$
0.1	3.89112728	5.40256135	2.13508599	5.26259202	3.51178444
t_{MaxD_t}	5.37980063	5.37667767	5.38377453	5.40297738	5.43101757
0.9	3.42821751	3.40254641	3.51069205	5.03363792	2.61960198

Аналогично для $\alpha = 0.01$:

$t \backslash x$	1	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{1}{5}$
0.1	0.87359323	0.89123356	0.90168780	0.90469718	0.90018686
$t_{MaxD_t}0$	0.99981996	0.99984661	0.99986216	0.99986660	0.99985994
0.9	0.29535552	0.35272414	0.39132466	0.40311003	0.38556182

Таблица 6: $\tau = \frac{h^2}{3}, m = 50$

$t \backslash x$	1	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{1}{5}$
0.1	0.87317738	0.89081313	0.90141794	0.90473469	0.90068254
$t_{MaxD_t}0$	0.99995483	0.99996149	0.99996544	0.99996666	0.99996517
0.9	0.29413794	0.35124527	0.39028283	0.40325867	0.38745649

Таблица 7: $\tau = \frac{h^2}{3}, m = 100$

$t \backslash x$	1	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{1}{5}$
0.1	0.87303833	0.89067196	0.90132521	0.90474153	0.90083822
$t_{MaxD_t}0$	0.99997990	0.99998286	0.99998462	0.99998518	0.99998454
0.9	0.29372975	0.35074990	0.38992539	0.40328576	0.38805324

Таблица 8: $\tau = \frac{h^2}{3}, m = 150$

Изучим главный член аппроксимации в опорных точках, ниже представлены значения Δuв них:

$t \backslash x$	1	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{1}{5}$
0.1	0.00041586	0.00042043	0.00026985	-0.00003751	-0.00049568
t_{MaxD_t}	-0.00013487	-0.00011488	-0.00010328	-0.00010006	-0.00010523
0.9	0.00121758	0.00147887	0.00104183	-0.00014865	-0.00189467

Таблица 9: $\tau = \frac{h^2}{3}, \Delta u$ при $m = 50- > m = 100$

$t \backslash x$	1	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{1}{5}$
0.1	0.00013905	0.00014117	0.00009274	-0.00000683	-0.00015567
t_{MaxD_t}	-0.00002507	-0.00002137	-0.00001919	-0.00001852	-0.00001938
0.9	0.00040819	0.00049537	0.00035744	-0.00002709	-0.00059675

Таблица 10: $\tau = \frac{h^2}{3}, \Delta u$ при $m = 100- > m = 150$

Итого, для $\frac{u(m=50)-u(m=100)}{u(m=100)-u(m=150)}$ в опорных точках:

$t \backslash x$	1	$\frac{4}{5}$	$\frac{3}{5}$	$\frac{2}{5}$	$\frac{1}{5}$
0.1	2.99071876	2.97815568	2.90984076	5.48961893	3.18411450
t_{MaxD_t}	5.37933200	5.37612763	5.38316217	5.40234397	5.43041343
0.9	2.98285475	2.98536772	2.91470846	5.48778344	3.17494935

Нетрудно заметить, что вычислительный эксперимент соответствует теоретическим результатам, а именно:

1) увеличение числа шагов приводит к более точному, (ограниченному) результату, что соответствует устойчивости схемы и граничных условий.

2) изменение функции, при увеличении числа шагов, уменьшается пропорционально квадрату длине малых отрезков, на которые разбивается изначальный, что соотносится с порядком аппроксимации, полученным теоретически.

4 Листинг программы

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
void RESHI(double *a, double *x, double *b,int N);
int Max(double *p,int nn, int M);
int Min(double *p,int w,int nn, int M);
void makeAandB(double *p,double *A, double *B, double t, double h,double a,int j,int M);


double diff (double a, double b, double p[], int nn, int M, int T,double strMax[],int numM
{
    double t, n1, M1, h;
    n1=nn;
    M1=M;
    t=T/(n1-1);
    h=2/(M1-1);
    for(int i=M; i<=M*2-1; i++){p[i]=1;}
    for(int j=nn-2; j>=0; j--)
    {
        double *A,*B,*X;
        B = (double *) malloc ((M - 2) * sizeof (double));
        X = (double *) malloc ((M - 2) * sizeof (double));
        A = (double *) malloc ((M - 2) * 3 * sizeof (double));

        makeAandB(p,A,B,t,h,a,j,M);
        RESHI(A,X,B,M-2);
        for(int i=1;i<=M-2;i++)
        {
            p[i]=X[i-1];
        }
        p[0]=1;
        p[M-1] = 1;
        if(j==(nn-1)/10){
            for(int i=0;i<5;i++)
                str09[i]=p[(i*M)/10+(M/5) ];
            // printf("%d////", j);
            // printf("\n");
        }
        if(j==9*(nn-1)/10){
            for(int i=0;i<5;i++)
                str01[i]=p[(i*M)/10+(M/5)];
            //printf("%d////", j);
        }
        for( int i=0;i<M-1;i++)

        {
            if(fabs(p[i]-p[M+i])>=fabs(s)){s=p[i]-p[M+i];numMax=j;
```

```
                //printf("%d////", numMax);
                //printf("%.8f     ", p[i]-p[M+i]);
                for(int k=0; k<5;k++){
                    strMax[k]=p[(k*M)/10+(M/5)];
                }
            }
        }
        for (int i = 0; i <= M - 1; i++){

            //printf("%.8f     ", p[i+M]);
            p[i+M]=p[i];
        }
        //printf("\n");


    }


    /*for(int i=0; i<=2*M-1; i++)
    {
        if (i%M==0) printf("\n");
        printf("%.8f     ", p[i]);
    }
    printf("\n");
    printf("\n");/*
       for(int i=0; i<=nn*M-1; i++)
       {
       if (i%M==0) printf("\n");
       printf("%.8f     ", Dx[i]);
       }
       printf("\n");*/
    return 0;
}


int main ()
{
    int nn=10,M=10,T=1,numMax,M1,nn1, numMax1, nn2, numMax2, M2;
    double  s=0,b = 0.1, a = 0.01,strMax[5],str01[5],str09[5],strMax1[5],str011[5],str091[5
    for(int j=0; j<1;j++){
        M = 50*(j+1);
        nn=M*M*3;
        double *p,*p1,*p2;
        p = (double *)malloc((2*M)* sizeof (double));
        diff(a,b,p,nn,M,T,strMax,numMax,str01,str09,s);
        /*printf ( "0.1 & %.8f & %.8f  & %.8f & %.8f & %.8f",str01[0],str01[1],str01[2],str
        printf("\\\\");
        printf("\n");
        printf ( "$t_{Max D_t} %d$ & %.8f & %.8f  & %.8f & %.8f & %.8f",numMax, strMax[0],
        printf("\\\\");
        printf("\n");
        printf ( "0.9 & %.8f & %.8f  & %.8f & %.8f & %.8f",str09[0],str09[1],str09[2],str09
        printf("\\\\");
        printf("\n");*/
```

```
M1 = 100*(j+1);
nn1=M1*M1*3;
p1 = (double *)malloc((2*M1)* sizeof (double));
diff(a,b,p1,nn1,M1,T,strMax1,numMax1,str011,str091,s);


M2 = 150*(j+1);
nn2=M2*M2*3;
p2 = (double *)malloc((2*M2)* sizeof (double));
diff(a,b,p2,nn2,M2,T,strMax2,numMax2,str012,str092,s);

printf ( "0.1 & %.8f & %.8f  & %.8f & %.8f & %.8f",str01[0]-str011[0],str01[1]-str0
         str01[2]-str011[2],str01[3]-str011[3],str01[4]-str011[4]);
printf("\\\\");
printf("\n");
printf ( "$t_{Max D_t}$ & %.8f & %.8f  & %.8f & %.8f & %.8f", strMax[0]-strMax1[0]
         strMax[2]-strMax1[2],strMax[3]-strMax1[3],strMax[4]-strMax1[4]) ;
printf("\\\\");
printf("\n");
printf ( "0.9 & %.8f & %.8f  & %.8f & %.8f & %.8f",str09[0]-str091[0],str09[1]-str0
         str09[2]-str091[2],str09[3]-str091[3],str09[4]-str091[4]);
printf("\\\\");

printf("\n");
printf("\n");
printf("\n");

printf ( "0.1 & %.8f & %.8f  & %.8f & %.8f & %.8f",str011[0]-str012[0],str011[1]-st
         str011[3]-str012[3],str011[4]-str012[4]);
printf("\\\\");
printf("\n");
printf ( "$t_{Max D_t}$ & %.8f & %.8f  & %.8f & %.8f & %.8f", strMax1[0]-strMax2[0]
         strMax1[2]-strMax2[2],strMax1[3]-strMax2[3],strMax1[4]-strMax2[4]) ;
printf("\\\\");
printf("\n");
printf ( "0.9 & %.8f & %.8f  & %.8f & %.8f & %.8f",str091[0]-str092[0],str091[1]-st
         str091[3]-str092[3],str091[4]-str092[4]);
printf("\\\\");

printf("\n");
printf("\n");
printf("\n");

printf ( "0.1 & %.8f & %.8f  & %.8f & %.8f & %.8f",(str01[0]-str011[0])/(str011[0]-
         (str01[1]-str011[1])/(str011[1]-str012[1]),(str01[2]-str011[2])/(str011[2]
         (str01[3]-str011[3])/(str011[3]-str012[3]),(str01[4]-str011[4])/(str011[4]
printf("\\\\");
printf("\n");
printf ( "$t_{Max D_t}$ & %.8f & %.8f  & %.8f & %.8f & %.8f", (strMax[0]-strMax1[0]
         (strMax[1]-strMax1[1])/(strMax1[1]-strMax2[1]),(strMax[2]-strMax1[2])/(str
         (strMax[3]-strMax1[3])/(strMax1[3]-strMax2[3]),(strMax[4]-strMax1[4])/(str
printf("\\\\");
printf("\n");
```

```
        printf ( "0.9 & %.8f & %.8f  & %.8f & %.8f & %.8f",(str09[0]-str091[0])/(str091[0]-
                  (str09[1]-str091[1])/(str091[1]-str092[1]),(str09[2]-str091[2])/(str091[2]
                  (str09[3]-str091[3])/(str091[3]-str092[3]),(str09[4]-str091[4])/(str091[4]
        printf("\\\\");


    }

    return 0;
}



int Max(double *p,int nn, int M)
{
    double s=0.0;
    int i,k;
    for(i=0;i<nn*M;i++)
    {
        if(p[i]>=s && !(p[i]>=100 && p[i]<=100)){s=p[i];k=i;}
    }
    return k;
}
int Min(double *p,int w,int nn, int M)
{
    double s=2.0;
    int i,k;
    for(i=0;i<nn*M;i++)
    {
        if(p[i]<s && i>=w*M){s=p[i];k=i;}
    }
    return k;
}
void RESHI(double *a, double *x, double *b,int N)
{
    double *s,*p,*y;
    s = (double *) malloc (N * sizeof (double));
    p = (double *) malloc (N* sizeof (double));
    y = (double *) malloc (N* sizeof (double));
    y[0]=a[0];
    s[0]=-a[1]/y[0];
    p[0]=b[0]/y[0];
    for(int i=1;i<N-1;i++)
    {
        y[i]=a[3*i]+a[3*i+2]*s[i-1];
        s[i]=-a[3*i+1]/y[i];
        p[i]=(b[i]-a[3*i+2]*p[i-1])/y[i];
    }
    y[N-1]=a[(N-1)*3]+a[(N-1)*3+2]*s[N-2];
    p[N-1]=(b[N-1]-a[(N-1)*3+2]*p[N-2])/y[N-1];
    x[N-1]=p[N-1];
    for(int i=N-2;i>=0;i--)
    {
        x[i]=s[i]*x[i+1]+p[i];
```

```
    }
    return;
}
void makeAandB(double *p,double *A, double *B, double t, double h,double a, int j, int M)
{
    A[0]=1/t+a/(h*h)+(1-h)*(1-h);
    A[1]=-a/(2*h*h);
    A[2] = 0;
    A[(M-2)*3-2] = 0;
    A[(M-2)*3-1]=-a/(2*h*h);
    A[(M-2)*3-3]=1/t+a/(h*h)+(1-h)*(1-h);

    B[0]=a/(2*h*h)+p[(1)*(M)+1]*(1/t-(1+(1-h)*(1-h)))+a*(p[(1)*M]-2*p[(1)*M+1]+p[(1)*M+2])/
    B[M-3]= a/(2*h*h)+a*(p[(1)*M+M-3]-2*p[(1)*M+M-2]+p[(1)*M+M-1])/(2*h*h)+p[(1)*(M)+M-2]*

    for(int i=1; i<M-3; i++)
    {
        A[3*i+2]=-a/(2*h*h);
        A[3*i]=1/t+a/(h*h)+(1+(h*(i+1)-1)*(h*(i+1)-1))/2;
        A[3*i+1]=-a/(2*h*h);
        B[i]=p[(1)*(M)+1+i]*(1/t-(1+(h*(i+1)-1)*(h*(i+1)-1))/2)+a*(p[(1)*M+i]-2*p[(1)*M+1+
    }
    return;
}
```