FINITE DIFFERENCES METHOD part III

Elliptic equation

Introduction

Typical examples are the Laplace and Poisson equations.

$$\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}=0, \quad (x,y)\in D\subset \mathbf{R}^2$$

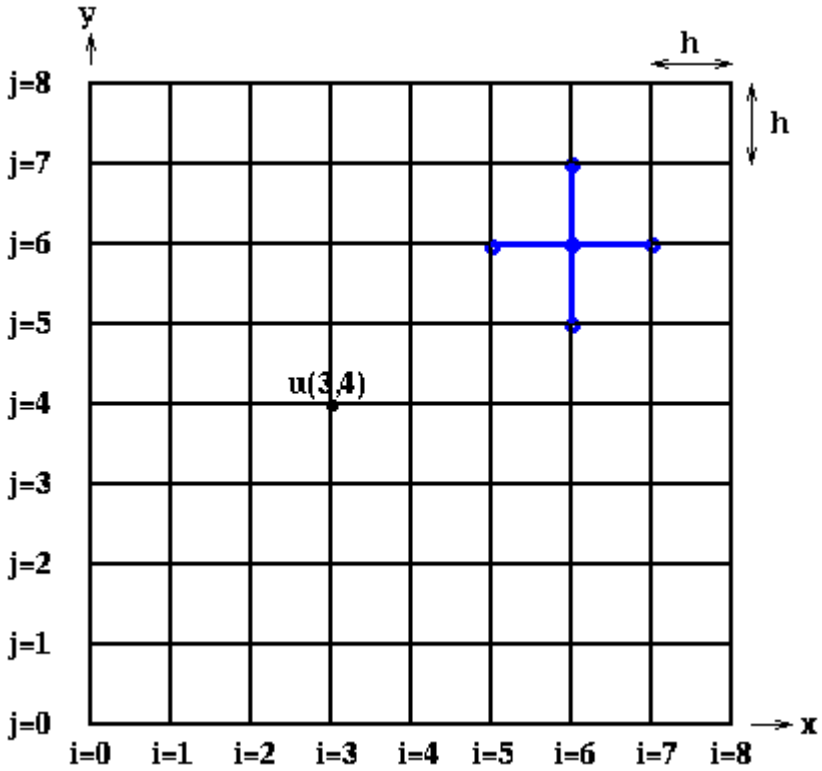
$$\left[\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right]=f(x,y), \quad (x,y)\in D\subset \mathbf{R}^2$$

Dirichlet type boundary conditions ($u(x,y)\big|_{\partial D}$ are known)

Neumann type boundary conditions ($\left.\frac{\partial u(x,y)}{\partial n}\right|_{\partial D}$ are known where **n** is the external normal to the boundary).

Finite differences scheme

Consider the mesh with the steps Δx and Δy in Ox and Oy directions, N_x and N_y being the number of the nodes in the two directions. We note the approximation of the solution in a point (i, j) of the mesh with



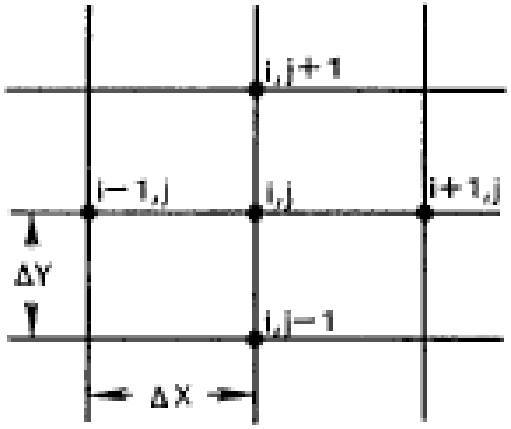
$$U_{i,j} \approx u\left(x_i, y_j\right),$$

$$i = 0,1,\ldots,N_x, \quad j = 0,1,\ldots,N_y.$$

We use an approximation with central finite differences and a scheme with 5 nodes:

$$\frac{U_{i+1,j} - 2U_{i,j} + U_{i-1,j}}{\Delta x^2} + \frac{U_{i,j+1} - 2U_{i,j} + U_{i,j-1}}{\Delta y^2} = 0 \ ,$$

$$i = 2,\ 3,\ ...,\ N_x - 1, \quad j = 2,\ 3,\ ...,\ N_y - 1 \qquad \text{(o)}$$



or

$$\frac{U_{i+1,j} - 2U_{i,j} + U_{i-1,j}}{\Delta x^2} + \frac{U_{i,j+1} - 2U_{i,j} + U_{i,j-1}}{\Delta y^2} = f\left(x_i, y_j\right) \ ,$$

$$i = 2,\ 3,\ ...,\ N_x - 1, \quad j = 2,\ 3,\ ...,\ N_y - 1$$

A system in the unknowns $U_{i,j}$ is obtained and the truncation error is given by

$$T_{r,s} = \frac{1}{12}(\Delta x)^2 \left(u_{xxxx} + u_{yyyy}\right)_{r,s} + o\left((\Delta x)^2\right).$$

and bounded by

$$|T_{r,s}| \leq T := \frac{1}{12}(\Delta x)^2 \left(M_{xxxx} + M_{yyyy}\right)$$

The system (o) can be written in the form:

$$U_{i+1,j} - 2U_{i,j} + U_{i-1,j} + \left(\frac{\Delta x}{\Delta y}\right)^2 \left(U_{i,j+1} - 2U_{i,j} + U_{i,j-1}\right) = 0 \ ,$$

$$i = 2,\ 3,\ ...,\ N_x - 1, \quad j = 2,\ 3,\ ...,\ N_y - 1$$

or

$$U_{i+1,j} + U_{i-1,j} + \beta^2 U_{i,j+1} + \beta^2 U_{i,j-1} - 2\left(1+\beta^2\right)U_{i,j} = 0 \ ,$$
$$i = 2,\ 3,\ ...,\ N_x - 1, \quad j = 2,\ 3,\ ...,\ N_y - 1 \qquad \text{(oo)}$$

The matrix of the system is sparse and can be written in a penta-diagonal form (Hoffmann and Chiang, 2000):

Example: We the Laplacian

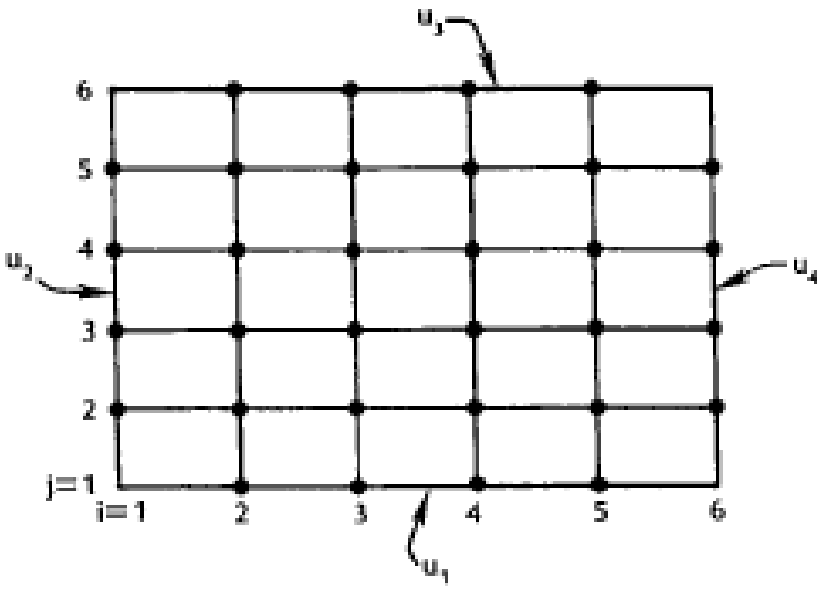
$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0, \quad (x, y) \in [0, L] \times [0, H]$$

along with the following boundary conditions

$$x = 0 \qquad u = u_2 \ , \quad y = 0 \qquad u = u_1$$

$$x = L \qquad u = u_4 \ , \quad y = H \qquad u = u_3$$

For a mesh of 6×6 nodes a system of 16 equations with 16 unknowns:



$$u_{3,4} + u_{1,4} + \beta^2 u_{2,5} + \beta^2 u_{2,3} - 2(1+\beta^2)u_{2,4} = 0$$

$$u_{4,4} + u_{2,4} + \beta^2 u_{3,5} + \beta^2 u_{3,3} - 2(1+\beta^2)u_{3,4} = 0$$

$$u_{5,4} + u_{3,4} + \beta^2 u_{4,5} + \beta^2 u_{4,3} - 2(1+\beta^2)u_{4,4} = 0$$

$$u_{6,4} + u_{4,4} + \beta^2 u_{5,5} + \beta^2 u_{5,3} - 2(1+\beta^2)u_{5,4} = 0$$

$$u_{3,5} + u_{1,5} + \beta^2 u_{2,6} + \beta^2 u_{2,4} - 2(1+\beta^2)u_{2,5} = 0$$

$$u_{3,2} + u_{1,2} + \beta^2 u_{2,3} + \beta^2 u_{2,1} - 2(1+\beta^2)u_{2,2} = 0$$

$$u_{4,2} + u_{2,2} + \beta^2 u_{3,3} + \beta^2 u_{3,1} - 2(1+\beta^2)u_{3,2} = 0$$

$$u_{5,2} + u_{3,2} + \beta^2 u_{4,3} + \beta^2 u_{4,1} - 2(1+\beta^2)u_{4,2} = 0$$

$$u_{6,2} + u_{4,2} + \beta^2 u_{5,3} + \beta^2 u_{5,1} - 2(1+\beta^2)u_{5,2} = 0$$

$$u_{3,3} + u_{1,3} + \beta^2 u_{2,4} + \beta^2 u_{2,2} - 2(1+\beta^2)u_{2,3} = 0$$

$$u_{4,3} + u_{2,3} + \beta^2 u_{3,4} + \beta^2 u_{3,2} - 2(1+\beta^2)u_{3,3} = 0$$

$$u_{5,3} + u_{3,3} + \beta^2 u_{4,4} + \beta^2 u_{4,2} - 2(1+\beta^2)u_{4,3} = 0$$

$$u_{6,3} + u_{4,3} + \beta^2 u_{5,4} + \beta^2 u_{5,2} - 2(1+\beta^2)u_{5,3} = 0$$

$$u_{4,5} + u_{2,5} + \beta^2 u_{3,6} + \beta^2 u_{3,4} - 2(1+\beta^2)u_{3,5} = 0$$

$$u_{5,5} + u_{3,5} + \beta^2 u_{4,6} + \beta^2 u_{4,4} - 2(1+\beta^2)u_{4,5} = 0$$

$$u_{6,5} + u_{4,5} + \beta^2 u_{5,6} + \beta^2 u_{5,4} - 2(1+\beta^2)u_{5,5} = 0$$

The system can be written in a matrix form having a band structure (5 diagonals are not zeros).

$$
\begin{bmatrix}
\alpha & 1 & 0 & 0 & \beta^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & \alpha & 1 & 0 & 0 & \beta^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & \alpha & 1 & 0 & 0 & \beta^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & \alpha & 0 & 0 & 0 & \beta^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\beta^2 & 0 & 0 & 0 & \alpha & 1 & 0 & 0 & \beta^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \beta^2 & 0 & 0 & 1 & \alpha & 1 & 0 & 0 & \beta^2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \beta^2 & 0 & 0 & 1 & \alpha & 1 & 0 & 0 & \beta^2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \beta^2 & 0 & 0 & 1 & \alpha & 0 & 0 & 0 & \beta^2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \beta^2 & 0 & 0 & 0 & \alpha & 1 & 0 & 0 & \beta^2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \beta^2 & 0 & 0 & 1 & \alpha & 1 & 0 & 0 & \beta^2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \beta^2 & 0 & 0 & 1 & \alpha & 1 & 0 & 0 & \beta^2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \beta^2 & 0 & 0 & 1 & \alpha & 0 & 0 & 0 & \beta^2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \beta^2 & 0 & 0 & 0 & \alpha & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \beta^2 & 0 & 0 & 1 & \alpha & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \beta^2 & 0 & 0 & 1 & \alpha & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \beta^2 & 0 & 0 & 1 & \alpha
\end{bmatrix}
\begin{bmatrix} u_{2,2} \\ u_{3,2} \\ u_{4,2} \\ u_{5,2} \\ u_{2,3} \\ u_{3,3} \\ u_{4,3} \\ u_{5,3} \\ u_{2,4} \\ u_{3,4} \\ u_{4,4} \\ u_{5,4} \\ u_{2,5} \\ u_{3,5} \\ u_{4,5} \\ u_{5,5} \end{bmatrix}
=
\begin{bmatrix} -u_{1,2} - \beta^2 u_{2,1} \\ -\beta^2 u_{3,1} \\ -\beta^2 u_{4,1} \\ -u_{6,2} - \beta^2 u_{5,1} \\ -u_{1,3} \\ 0 \\ 0 \\ -u_{6,3} \\ -u_{1,4} \\ 0 \\ 0 \\ -u_{6,4} \\ -u_{1,5} - \beta^2 u_{3,6} \\ -\beta^2 u_{3,6} \\ -\beta^2 u_{4,6} \\ -u_{6,5} - \beta^2 u_{5,6} \end{bmatrix}
$$

where $\alpha = -2\left(1+\beta^2\right)$

Iterative methods for linear systems of equations

The iterative methods are based on the fixed point theory. Thus, we start from an initial solution that is iteratively improved. The simplest iterative method is the Jacobi method and for the system (oo) has the form:

$$u_{ij}^{k+1} = \frac{1}{2(1+\beta^2)}\left[u_{i+1j}^{k} + u_{i-1j}^{k} + \beta^2(u_{ij+1}^{k} + u_{ij-1}^{k})\right]$$

Initial solution: u^0

Improved approximations: $u^1, u^2, ..., u^n, u^{n+1}, ...$

An improvement of the Jacobi method is given by the **Gauss-Seidel** method in which an already calculated value is used to obtain the other values (i.e. $u_{i,j-1}^{k+1}$ is used to calculate the value of $u_{i,j}^{k+1}$). Thus, the Gauss-Seidel method has the form:

$$u_{ij}^{k+1} = \frac{1}{2(1+\beta^2)}\left[u_{i+1j}^{k} + u_{i-1j}^{k+1} + \beta^2(u_{ij+1}^{k} + u_{ij-1}^{k+1})\right]$$

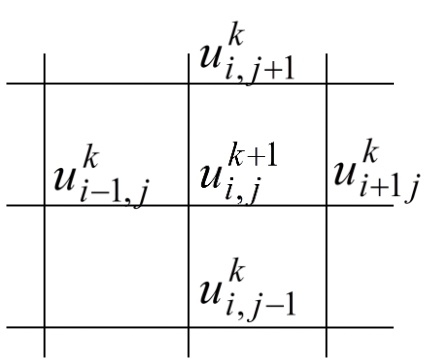
The Jacobi and Gauss-Seidel methods are convergent if the systems matrices are diagonal dominant. A matrix $A = (a_{ij})$ is diagonal dominant if:

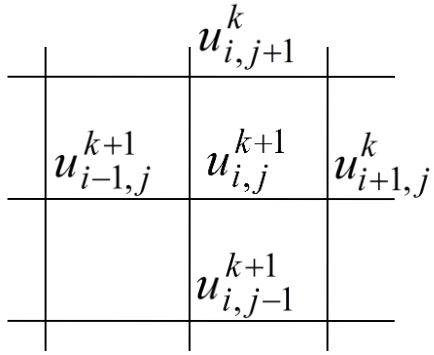
$$|a_{ii}| \geq \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|$$

The stop criteria for the both methods is given by

$$\left\|u^{k+1} - u^{k}\right\| < \varepsilon \qquad \text{or} \qquad \frac{\left\|u^{k+1} - u^{k}\right\|}{\left\|u^{k}\right\|} < \varepsilon$$

where ε is an accepted error.





In many cases the convergence of the Jacobi and Gauss-Seidel methods can be accelerated by introducing a relaxation parameter, ω. The algorithm is given by:

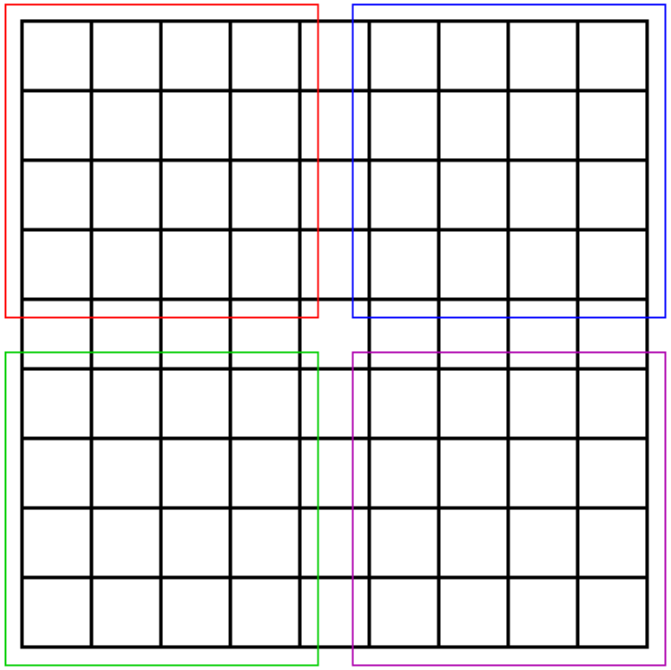
$$u_{i,j}^{k+1} = (1-\omega)u_{i,j}^{k} + \frac{\omega}{2(1+\beta^2)}\left[u_{i+1,j}^{k} + u_{i-1,j}^{k+1} + \beta^2(u_{i,j+1}^{k} + u_{i,j-1}^{k+1})\right]$$

where $0 < \omega < 2$. For $\omega < 1$ we have under-relaxation, for $\omega = 1$ the Gauss-Seidel method is obtained and otherwise we have over relaxation. In some particular cases the optimum parameter ω can be calculated, but generally is obtained by numerical experiments.

Parallelization

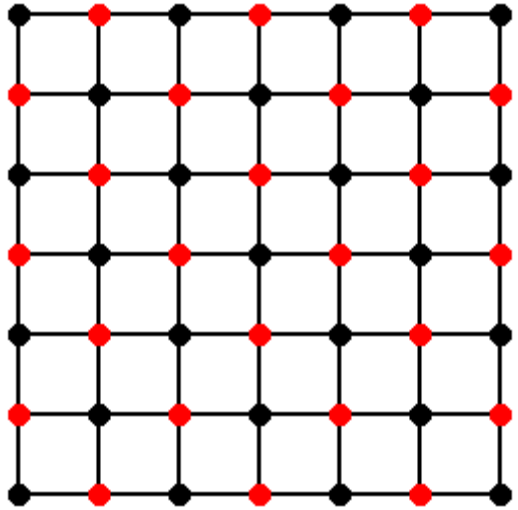
Jacobi

Partitioning of the 2D Poisson Equation



Gauss Seidel

```
for all black (i,j) grid points
U(i,j,m+1) = (U(i-1,j,m) + U(i+1,j,m) +
      U(i,j-1,m) + U(i,j+1,m) + b(i,j))/4
  end for

for all red (i,j) grid points
U(i,j,m+1) = (U(i-1,j,m+1) + U(i+1,j,m+1) +
      U(i,j-1,m+1) + U(i,j+1,m+1) + b(i,j))/4
  end for
```



Black points have only Red neighbors

Red points have only Black neighbors

(https://people.eecs.berkeley.edu/~demmel/cs267/lecture24/lecture24.html)

Alternating direction implicit (ADI) method for elliptic equations

Consider the parabolic equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

with the boundary conditions $u(x, y, t)\big|_{\partial D}$ and the initial condition $u(x, y, 0)\big|_{D}$.

The explicit scheme for a grid with the steps $\Delta x = \Delta y$ has the form:

$$u_{ij}^{n+1} = u_{ij}^{n} + \frac{\Delta t}{(\Delta x)^2}\left[u_{i+1j}^{n} + u_{i-1j}^{n} - 4u_{ij}^{n} + u_{ij+1}^{n} + u_{ij-1}^{n}\right]$$

If we choose $\Delta t/(\Delta x)^2 = 1/4$ it becomes

$$u_{ij}^{n+1} = \frac{1}{4}\left(u_{i+1j}^{n} + u_{i-1j}^{n} + u_{ij+1}^{n} + u_{ij-1}^{n}\right) \qquad (*)$$

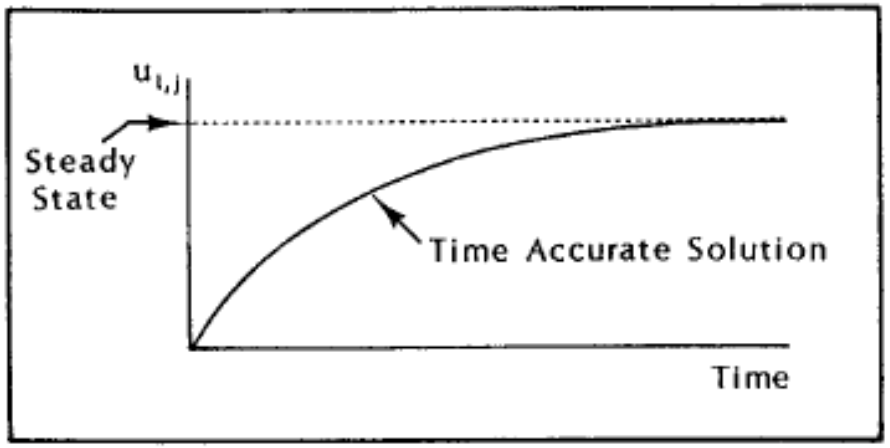
For $\beta = 1$ the Jacobi method is obtained:

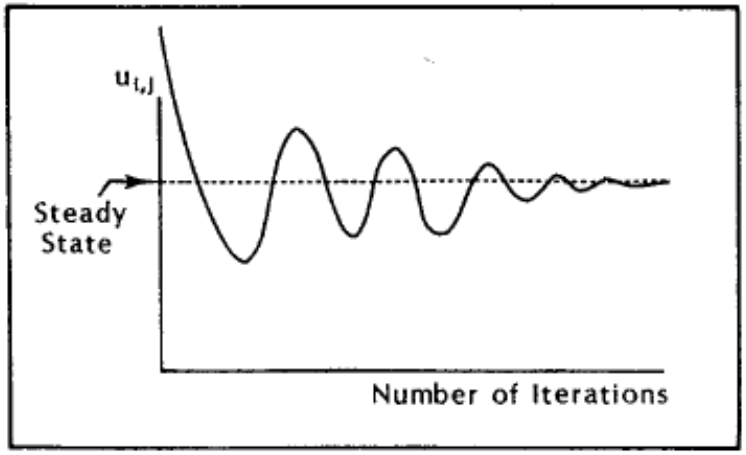
$$u_{ij}^{k+1} = \frac{1}{4}\left(u_{i+1j}^{k} + u_{i-1j}^{k} + u_{ij+1}^{k} + u_{ij-1}^{k}\right) \qquad (**)$$

We notice that the explicit scheme for the heat equation is equivalent with the Jacobi iteration for the elliptic equation even if the two equations model different physical phenomena.

This analogy suggests us that the numerical methods used for parabolic equations can be used to solve elliptic equations.

We say that a time depending solution reach the steady state if after a long time interval there are no differences in time for the solution.





It is worth mentioning that the solution of the parabolic equation using (*) has a precise physical signification, the distribution of u at the corresponding time step, while the intermediary solution obtained using (**) has no physical signification.

Further, we use the ADI method for the elliptic problem discretized by:

$$\frac{U_{r+1,s}-2U_{r,s}+U_{r-1,s}}{(\Delta x)^2}+\frac{U_{r,s+1}-2U_{r,s}+U_{r,s-1}}{(\Delta y)^2}=0$$

An iteration parameter τ is introduced:

$$\frac{U_{r,s}^{n+1*}-U_{r,s}^{n}}{\tau}=-\frac{1}{2}\left[\frac{U_{r+1,s}^{n+1*}-2U_{r,s}^{n+1*}+U_{r-1,s}^{n+1*}}{(\Delta x)^2}+\frac{U_{r,s+1}^{n}-2U_{r,s}^{n}+U_{r,s-1}^{n}}{(\Delta y)^2}\right] \quad \text{(a)}$$

$$\frac{U_{r,s}^{n+1}-U_{r,s}^{n+1*}}{\tau}=-\frac{1}{2}\left[\frac{U_{r+1,s}^{n+1}-2U_{r,s}^{n+1}+U_{r-1,s}^{n+1}}{(\Delta y)^2}+\frac{U_{r,s+1}^{n+1*}-2U_{r,s}^{n+1*}+U_{r,s-1}^{n+1*}}{(\Delta x)^2}\right] \quad \text{(b)}$$

If we know the values of the right hand part of the Eq. (a) it is possible to calculate the intermediary values U^{n+1*}, and then from Eq, (b) we find U^{n+1}. We can notice that the iteration parameter τ can be variable and the scheme is convergent for every $\tau > 0$ (see Morega, 1998), The iterative process ends when the following condition is fulfilled (see Mitchell şi Griffiths, 1980):

$$\left|U_{r,s}^{n+1} - U_{r,s}^{n}\right| < \varepsilon$$

Example

Heat generation in a rectangular square:

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + 1 = 0, \quad (x, y) \in D = [0,1] \times [0,1]$$
$$T(x, y)\big|_{\partial D} = 0 \qquad \text{(III.87)}$$

Discretization:

$$\frac{U_{i+1,j} - 2U_{i,j} + U_{i-1,j}}{\Delta x^2} + \frac{U_{i,j+1} - 2U_{i,j} + U_{i,j-1}}{\Delta y^2} + 1 = 0 \ ,$$
$$i = 2,\ 3,\ ...,\ N_x - 1, \quad j = 2,\ 3,\ ...,\ N_y - 1$$

We apply further the Jacobi and Gauss-Seidel methods. The Matlab programs are:

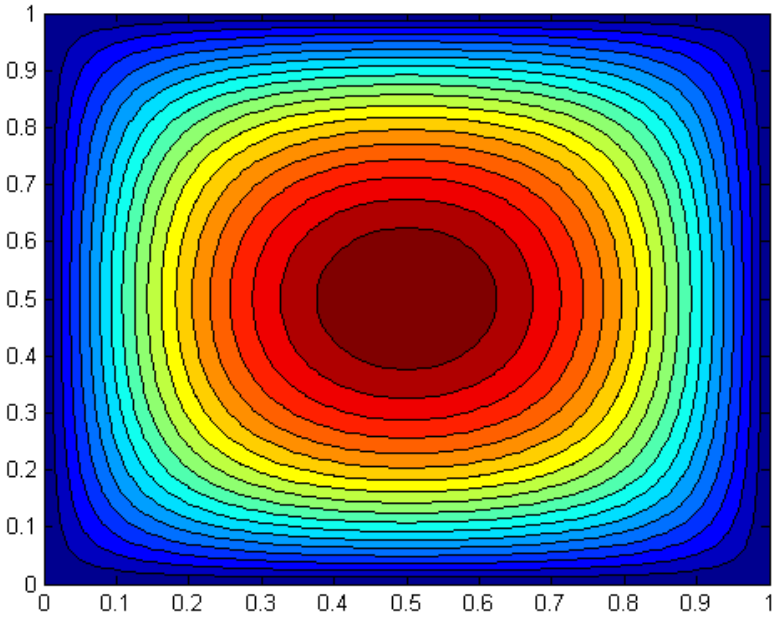
```
%Jacobi
%discretization and initialization%%%%%%%%%%
tic
N=101; %number of nodes in x- direction
h=1/(N-1);
T=zeros(N,N);Tnew=T;

nr_it=0;stop=0;
while (stop~=1)
    nr_it=nr_it+1; errT=0;
    for i=2:N-1
        for j=2:N-1
           Tnew(i,j)=0.25*(T(i+1,j)+T(i-1,j)+T(i,j+1)+T(i,j-1)+h*h);

           if abs(T(i,j)-Tnew(i,j))>errT
                errT=abs(T(i,j)-Tnew(i,j));

           end
        end
    end;

    if errT<1e-6
           stop=1;
    end


     if mod(nr_it,250)==0
```

```
        fprintf('nr_it=%g err=%g\n', nr_it,errT));
    end

T=Tnew;

end %while

x=0:h:(N-1)*h;y=x;
contour(x,y,T',20)
axis equal
axis([0,1,0,1])
T_max=max(max(T))
nr_it
toc


%Gauss-Seidel
%discretization and initialization%%%%%%%%%%
tic
N=101; %number of nodes in x- direction
h=1/(N-1);
T=zeros(N,N);
Tnew=T;

nr_it=0;
stop=0;
while (stop~=1)
    nr_it=nr_it+1; errT=0;
    errT=0;
```

```
    for i=2:N-1
        for j=2:N-1
            Tnew(i,j)=0.25*(T(i+1,j)+T(i-1,j)+T(i,j+1)+T(i,j-1)+h*h);
            if abs(T(i,j)-Tnew(i,j))>errT
                errT=abs(T(i,j)-Tnew(i,j));
            end

            T(i,j)=Tnew(i,j);
        end
    end;
     if errT<1e-6
            stop=1;
     end


 if mod(nr_it,250)==0
       fprintf('nr_it=%g err=%g\n', errT);
    end
T=Tnew;
end

x=0:h:(N-1)*h;y=x;
contour(x,y,T',20)
axis equal
axis([0,1,0,1])
T_max=max(max(T))
nr_it
toc
```

We notice that Gauss-Seidel method is faster and needs less iteration.

Grid	Jacobi			Gauss-Seidel		
	T_{max}	Number of iterations	Computing time (seconds)	T_{max}	Number of iterations	Computing time (seconds)
51 x 51	0.073142	2577	0.625	0.073396	1465	0.422
101 x 101	0.071640	7502	6.969	0.0726535	4454	4.594



Temperature distribution for the example

Hyperbolic equations (one space variable) (Wave equation)

Characteristics

Consider the simplest equation of this kind (see, Morton and Mayers, 2005)

$$\frac{\partial u}{\partial t} + a(x,t)\frac{\partial u}{\partial x} = 0, \quad x \in [a,b],\ t \geq 0 \qquad (\#)$$

$$u(x,0) = u^0(x)$$

It is possible to study this equation by using the *method of characteristics.* Recall that for an equation of the form (see Smith, 1985):

$$a(x,t)\frac{\partial u}{\partial t} + b(x,t)\frac{\partial u}{\partial x} = c(x,t)$$

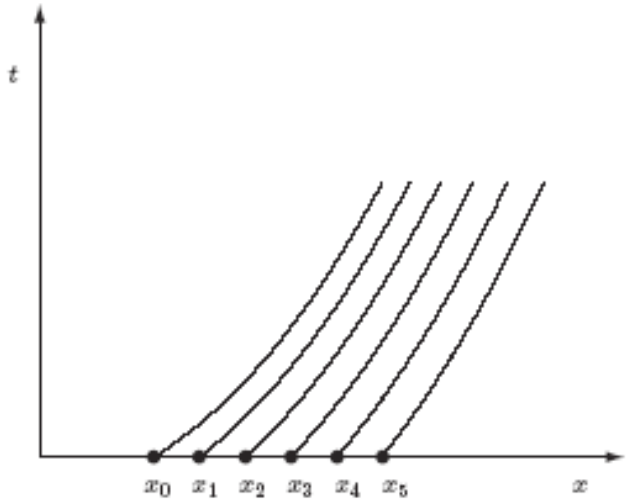
The characteristics are the solutions of the equation

$$a\,dy - b\,dt = 0$$

and one can find $u(x,t)$ from

$$c\,dt - a\,du = 0.$$

A general formulation of the method given before an be written in a compact way:



Typical characteristics of the wave equation

$$\frac{dt}{a} = \frac{dy}{b} = \frac{dU}{c}$$

For eq. (#) ($c = 0$), along a charcteristic $u(x,t)$ satisfies

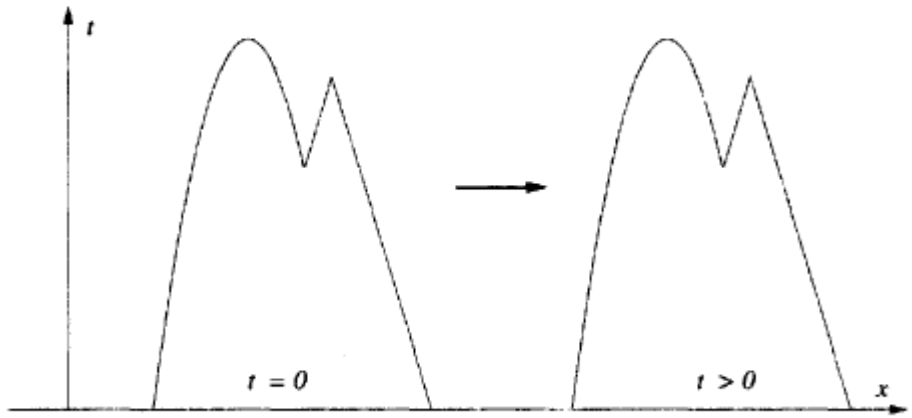
$$\frac{du}{dt}=\frac{\partial u}{\partial t}-\frac{\partial u}{\partial x}\frac{\partial x}{\partial t}=0$$

and then, from the boundary condition (#), we have $u(x,t)=u^0(x_j)$ for the characteristic starting in the point x_j (see figure).

If $a(x,t) \underset{notation}{=} a=const.$ then the characteristics are $x-at=const.$, and the solution of eq. (#) is:

$$u(x,t)=u^0(x-at)$$

We notice that this solution is the initial solution shifted to the left if $a<0$ or to the right if $a>0$ (vezi, Strikwerda, 2004)). Thus, the solution of the wave equation (#) is a wave propagated with the speed a without changing its shape.



Solution shifting for the 1D wave equation.

Courant, Friedrichs and Lewy condition (*CFL*)

In order to solve numerically eq. (#) we can use forward finite difference in time and backward finite difference in space:

$$\frac{U_i^{n+1} - U_i^n}{\Delta t} + a\frac{U_i^n - U_{i-1}^n}{\Delta x} = 0$$

or

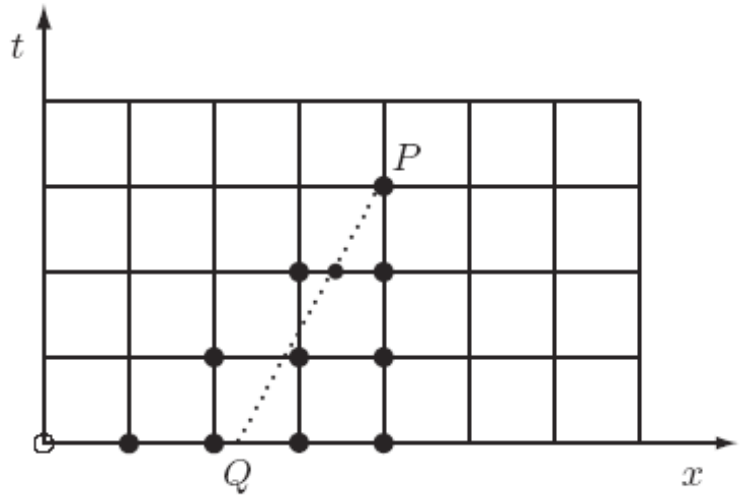
$$U_j^{n+1} = U_j^n - \frac{a\Delta t}{\Delta x}\left(U_i^n - U_{i-1}^n\right) = (1-\nu)U_i^n + \nu U_{i-1}^n \qquad (\#\#)$$

where $\nu = \frac{a\Delta t}{\Delta x}$.

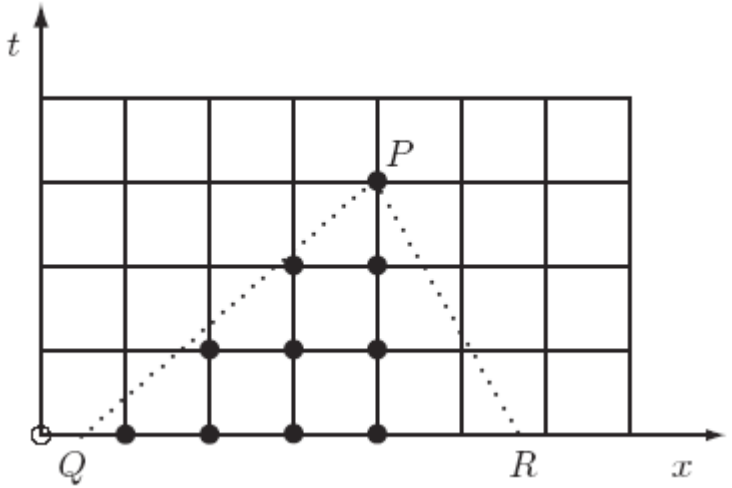
One can notice that the value U_j^{n+1} depends on two values calculated at the time step n, and these two values depend on other two values calculated at the time step $n-1$.

Thus, it is possible to obtain a domain of dependence for the data involved in the explicit numerical scheme. It has a triangular shape.

The dependence domain corresponding to the partial differential equation is exactly the characteristic passing through the point (x_i, t_n).



Dependenţa datelor în schema explicită

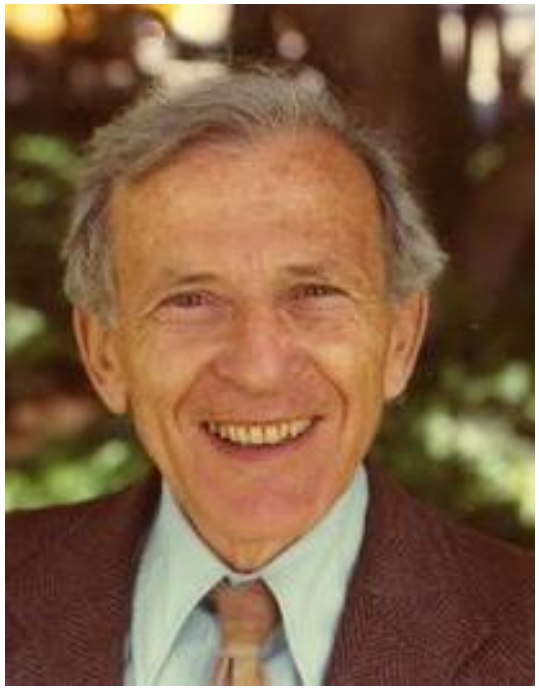


Violation of the CFL condition

The CFL condition mention that a numerical scheme is convergent if the data dependence domain of the characteristic belong to the dependence domain of the numerical scheme.

Richard Courant (1888 – 1972)

Kurt Otto Friedrichs (1901 – 1982)



Hans Lewy (1904 – 1988)

For the scheme (##) we don't have convergence for $a<0$ (the characteristics are similar with PR). For $a>0$ in order to achieve the convergence it is necessary that:

$$a\Delta t/\Delta x \le 1.$$

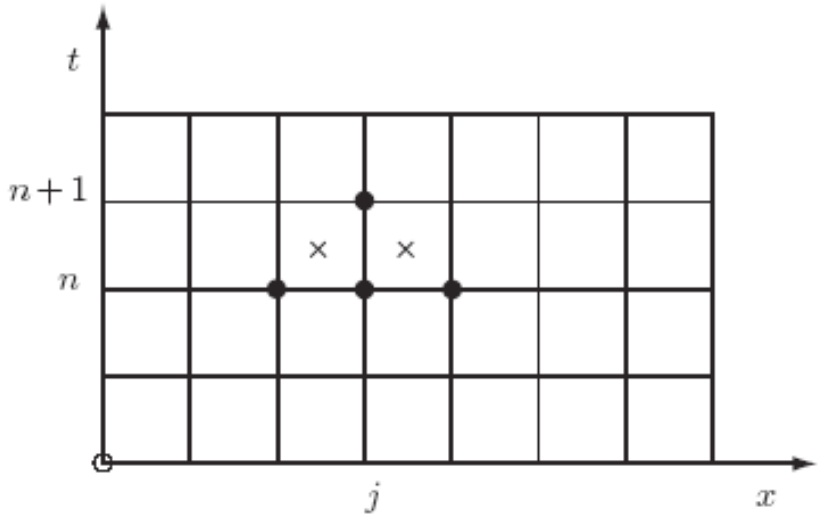
If a central difference discretization for the space derivative is used the following difference scheme is obtained:

$$\frac{U_i^{n+1}-U_i^n}{\Delta t}+a\frac{U_{i+1}^n-U_{i-1}^n}{2\Delta x}=0$$

For suitable steps Δt and Δx, this scheme satisfy the CFL condition

$$c=|a|\frac{\Delta t}{\Delta x}\le 1 \qquad \text{(Courant number)}$$

For both negative and positive values of a. Data dependence is given bellow.



Data dependence for the central scheme.

In order to study the scheme stability we use again the normal modes from the Fourier analysis

$$U_i^n = \lambda^n e^{I\,k(i\Delta x)}$$

And substituting it into the central difference scheme the following amplification factor is obtained:

$$\lambda \equiv \lambda(k) = 1 - a\frac{\Delta t}{\Delta x}\sin(k\Delta x)$$

which can be $|\lambda > 1|$ and then the scheme is unstable. The scheme respect the CFL condition, but it is not stable. Thus, the CFL condition is only necessary to obtain the convergence, the condition is not sufficient.

A simple scheme that solve this problem can be (Morton şi Meyers 2005):

$$U_j^{n+1} = \begin{cases} U_j^n - a\dfrac{\Delta t}{\Delta x}\Delta_{+x}U_j^n \text{ if } a < 0, \\ U_j^n - a\dfrac{\Delta t}{\Delta x}\Delta_{-x}U_j^n \text{ if } a > 0. \end{cases} \qquad (\#\#\#)$$

We notice that the scheme can be used for $a = a(x,t)$, also. Schema (###) satisfy the CFL condition and in addition it is stable. Indeed for $a > 0$ (if $a < 0$ one replaces a with $|a|$) we have:

$$\lambda \equiv \lambda(k) = 1 - (a\Delta t/\Delta x)(1 - \mathrm{e}^{-\mathrm{i}k\Delta x}) \equiv 1 - \nu(1 - \mathrm{e}^{-\mathrm{i}k\Delta x}).$$

and

$$\begin{aligned} |\lambda(k)|^2 &= [(1-\nu) + \nu\cos k\Delta x]^2 + [\nu \sin k\Delta x]^2 \\ &= (1-\nu)^2 + \nu^2 + 2\nu(1-\nu)\cos k\Delta x \\ &= 1 - 2\nu(1-\nu)(1-\cos k\Delta x) \end{aligned}$$

or

$$|\lambda|^2 = 1 - 4\nu(1-\nu)\sin^2 \tfrac{1}{2}k\Delta x.$$

Obviously, $|\lambda| \le 1$ for $0 \le \nu \le 1$.

Other explicit schemes

Using different kind of discretization many finite difference schemes can be obtained. For example, by using the average values of U_i^n in the explicit scheme (##) the **Lax's method** is obtained:

$$u_i^{n+1} = \frac{1}{2}(u_{i+1}^n + u_{i-1}^n) - \frac{a\Delta t}{2\Delta x}(u_{i+1}^n - u_{i-1}^n) \qquad \text{(L)}$$

If for the both derivatives (in time and in space) the central discretization is used then the so called **leapfrog method (**săritura broaştei**)** is obtained. It is of order $O(\Delta t^2, \Delta x^2)$:

$$\frac{u_i^{n+1} - u_i^{n-1}}{2\Delta t} = -a\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x} \qquad \text{(LF)}$$

This two methods are stable for a Courant number $c \le 1$.

Peter Lax (1926 -) and **Burton Wendroff** (1930 -)

We have to mention that in order to integrate the scheme (LF) two sets of starting data are necessary, and this diminishes the efficieny of the method even if the method is of order two.

One of the most efficient explicit scheme is the **Lax-Wendorff** method and can be obtained by using Taylor expansions. Consider the expansion:

$$u(x,\ t+\Delta t) = u(x,t) + \frac{\partial u}{\partial t}\Delta t + \frac{\partial^2 u}{\partial t^2}\frac{(\Delta t)^2}{2!} + O(\Delta t)^3$$

or

$$u_i^{n+1} = u_i^n + \frac{\partial u}{\partial t}\Delta t + \frac{\partial^2 u}{\partial t^2}\frac{(\Delta t)^2}{2!} + O(\Delta t)^3$$

Eq (#) can be written as

$$\frac{\partial u}{\partial t} = -a\frac{\partial u}{\partial x}$$

Deriving with respect to t we have:

$$\frac{\partial^2 u}{\partial t^2} = -a\frac{\partial}{\partial t}\left(\frac{\partial u}{\partial x}\right) = -a\frac{\partial}{\partial x}\left(\frac{\partial u}{\partial t}\right) = a^2\frac{\partial^2 u}{\partial x^2}$$

Using these values in the Taylor expansion one get:

$$u_i^{n+1} = u_i^n + \left(-a\frac{\partial u}{\partial x}\right)\Delta t + \frac{(\Delta t)^2}{2}\left(a^2\frac{\partial^2 u}{\partial x^2}\right)$$

Using central finite difference for the space derivatives the Lax-Wendorff scheme is obtained:

$$u_i^{n+1} = u_i^n - a\Delta t\left[\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x}\right] + \frac{1}{2}a^2(\Delta t)^2\left[\frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{(\Delta x)^2}\right] \qquad \text{(LW)}$$

Scheme (LW) is of second order and is stable for $c \leq 1$.

Implicit methods

By using in eq. (#) forward finite difference in time and central finite difference in space an implicit scheme of order $O\left(\Delta t, \Delta x^2\right)$ is obtained. This leads to a tridiagonal system of algebraic equations:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -\frac{a}{2\Delta x}\left[u_{i+1}^{n+1} - u_{i-1}^{n+1}\right]$$

or

$$\frac{1}{2}c\,u_{i-1}^{n+1} - u_i^{n+1} - \frac{1}{2}c\,u_{i+1}^{n+1} = -u_i^n \qquad \text{(FI)}$$

By using backward discretization for both temporal and space variable a method of order $O(\Delta t, \Delta x)$ is obtained, but the system is much easier to solve:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -a\frac{u_i^{n+1} - u_{i-1}^{n+1}}{\Delta x}$$

$$c\,u_{i-1}^{n+1} - (1+c)u_i^{n+1} = -u_i^n \qquad \text{(BI)}$$

One of the most used implicit method is the Crank-Nicolson method

$$\frac{u_i^{n+1}-u_i^n}{\Delta t}=-a\frac{1}{2}\left[\frac{u_{i+1}^{n+1}-u_{i-1}^{n+1}}{2\Delta x}+\frac{u_{i+1}^{n}-u_{i-1}^{n}}{2\Delta x}\right]$$

having the order $O\left(\Delta t^2,\Delta x^2\right)$ and leads to the following system:

$$\frac{1}{4}c\,u_{i-1}^{n+1}-u_i^{n+1}-\frac{1}{4}c\,u_{i+1}^{n+1}=-u_i^n+\frac{1}{4}c\left(u_{i+1}^n-u_{i-1}^n\right) \qquad \text{(CN)}$$

Example

Consider the equation:

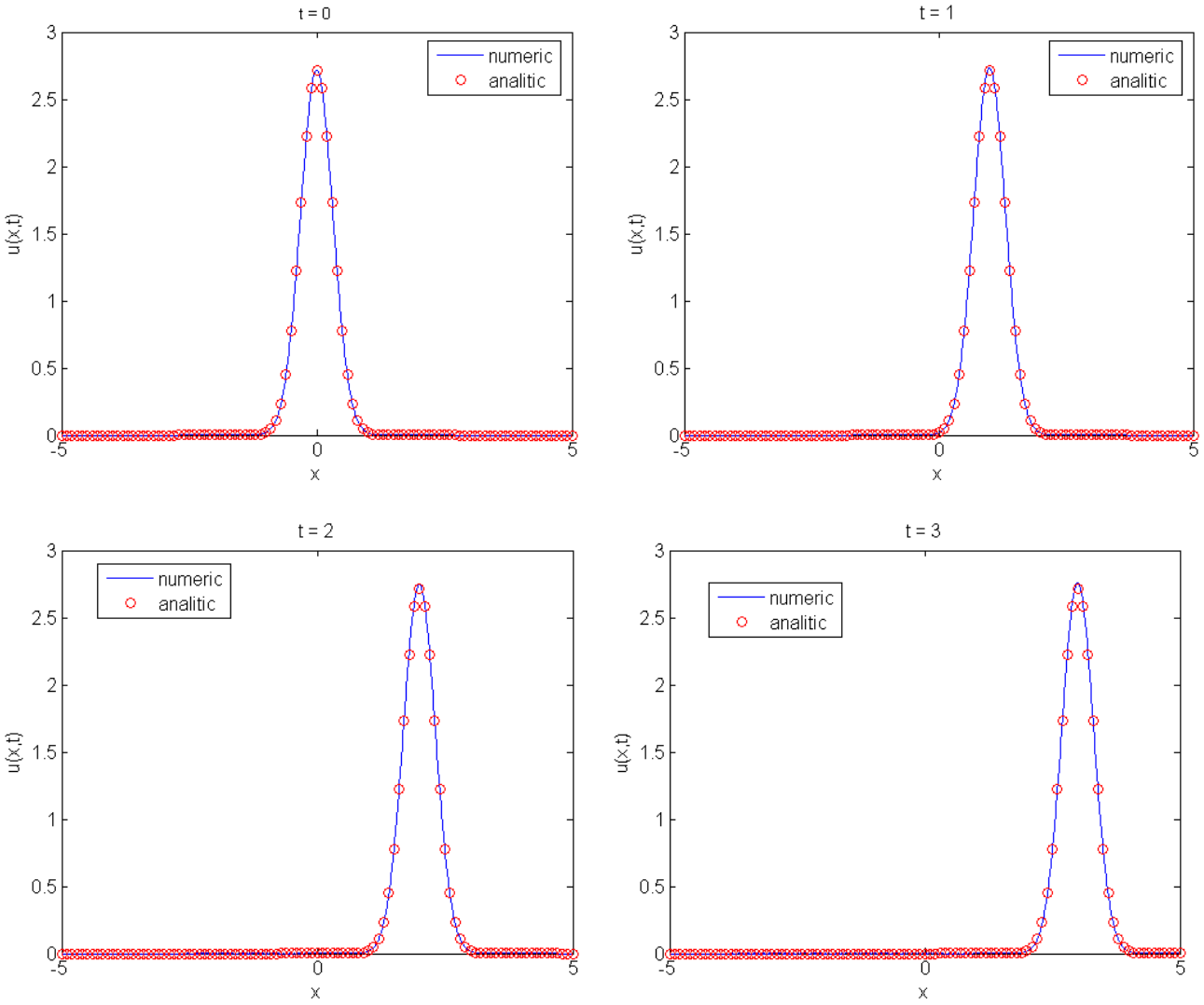
$$\frac{\partial u}{\partial t}+\frac{\partial u}{\partial x}=0,\quad x\in[-5,5],\ t\geq 0$$

$$u(x,0)=\exp\left(1-5x^2\right)$$

with the analytical solution $u(x,t)=\exp\left(1-5(x-t)^2\right)$.

We solve the problem numerically using the Lax-Wendorff scheme.

Variation of the solution:



The Matlab program:

```
a=-5;b=5;dt=0.001;dx=0.01;
x=a:dx:b;xa=a:10*dx:b;
N=length(x);
uo=exp(1-5*x.^2);% initial condition
tf=3;t=0;
uex=exp(1-5*(xa-t*ones(1,length(xa))).^2);% exact solution
nr_it=0;
plot(x,uo,'b',xa,uex,'or')
pause
k=1;
M(k)=getframe;
while t<tf
 t=t+dt;
 nr_it=nr_it+1;
 un(1)=uo(1)-dt/dx*(uo(2)-uo(1));
 for i=2:N-1
   un(i)=uo(i)-dt/dx/2*(uo(i+1)-uo(i-1))+0.5*dt*dt/dx/dx*(uo(i+1)
         -2*uo(i)+uo(i-1));
 end
 un(N)=uo(N)-dt/dx*(uo(N)-uo(N-1));
```

```
 if mod(nr_it,10)==0
   k=k+1;
   uex=exp(1-5*(xa-t*ones(1,length(xa))).^2);
   plot(x,un,'b',xa,uex,'or');
   M(k)=getframe;
 end
uo=un;
end
movie(M)
```