MECH 479/587

Computational Fluid Dynamics

Module 3 – Part B: 2D Finite Difference Approximation

Rajeev K. Jaiman
Term 1, 2022

Unsteady Multi-Dimensional Equation

- Consider 2D model equation

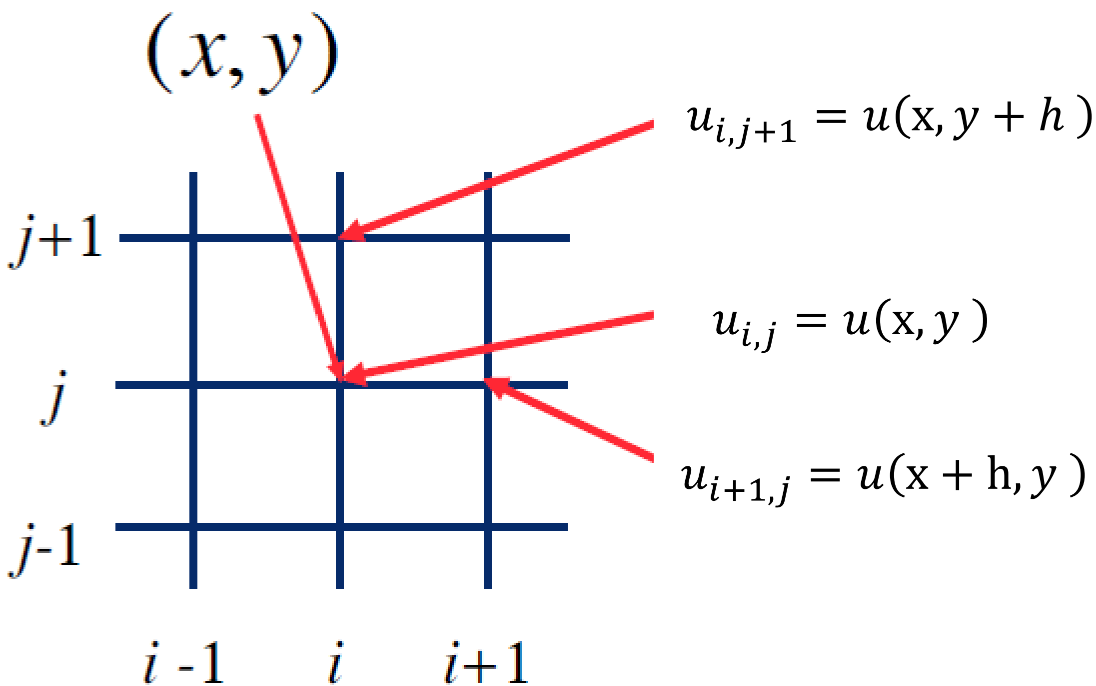
$$\frac{\partial u}{\partial t}+U\frac{\partial u}{\partial x}+V\frac{\partial u}{\partial y}=\nu\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)$$

$$\left(\frac{\partial u}{\partial t}\right)_{i,j}^{n}+U\left(\frac{\partial u}{\partial x}\right)_{i,j}^{n}+V\left(\frac{\partial u}{\partial y}\right)_{i,j}^{n}=\nu\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)_{i,j}^{n}$$

Based on the one-dimensional procedure discussed earlier, the extension to two-dimension and three-dimension is relatively straight forward.

Two-Dimensional Grid

- For 2D flow physics, discretize the variables on a two-dimensional grid



2D Discretization

$$\left(\frac{\partial u}{\partial t}\right)_{i,j}^{n} + U\left(\frac{\partial u}{\partial x}\right)_{i,j}^{n} + V\left(\frac{\partial u}{\partial y}\right)_{i,j}^{n} = \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)_{i,j}^{n}$$

$$\frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t} = -U\left(\frac{u_{i+1,j}^{n} - u_{i-1,j}^{n}}{2h}\right) - V\left(\frac{u_{i,j+1}^{n} - u_{i,j-1}^{n}}{2h}\right)$$

$$+\nu\left(\frac{u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n}}{h^2} + \frac{u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}}{h^2}\right)$$

$$\boxed{\begin{aligned} u_{i,j}^{n+1} = u_{i,j}^{n} &- \frac{U\Delta t}{2h}\left(u_{i+1,j}^{n} - u_{i-1,j}^{n}\right) - \frac{V\Delta t}{2h}\left(u_{i,j+1}^{n} - u_{i,j-1}^{n}\right) \\ &+ \frac{\nu\Delta t}{h^2}\left(u_{i+1,j}^{n} + u_{i-1,j}^{n} + u_{i,j+1}^{n} + u_{i,j-1}^{n} - 4u_{i,j}^{n}\right) \end{aligned}}$$

Accuracy: $O(\Delta t, h^2)$

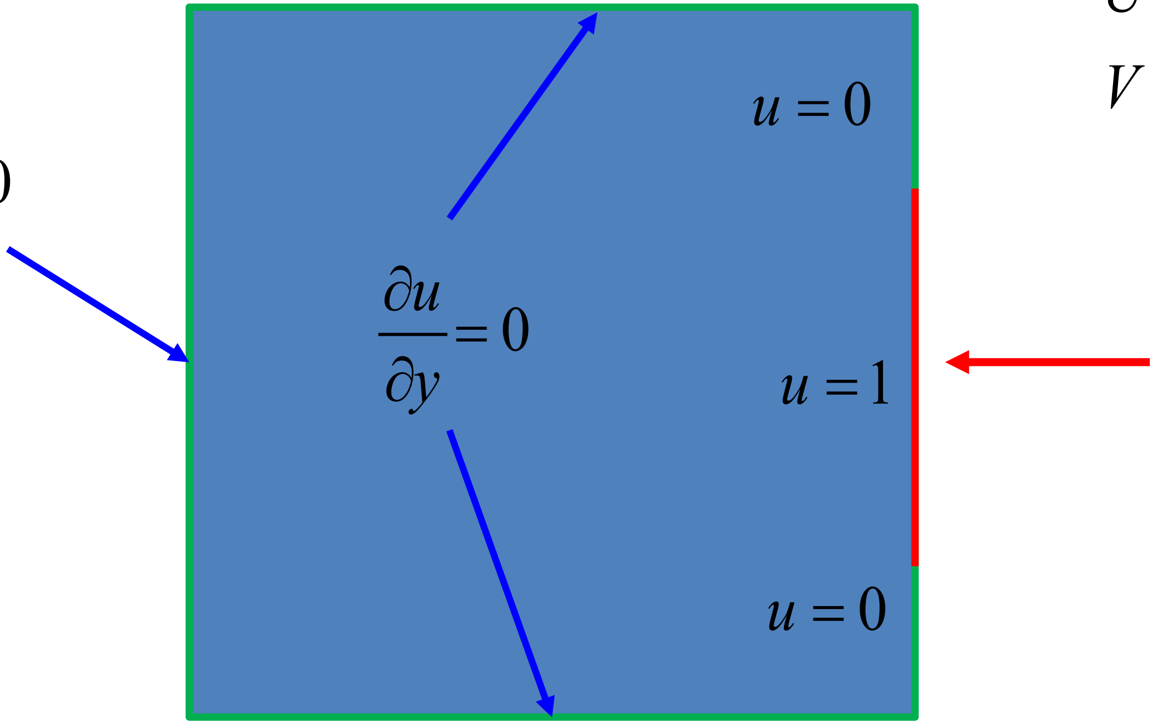
Example

$$\frac{\partial u}{\partial t} + U\frac{\partial u}{\partial x} + V\frac{\partial u}{\partial y} = \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$$

$$U = -1$$

$$V = 0$$

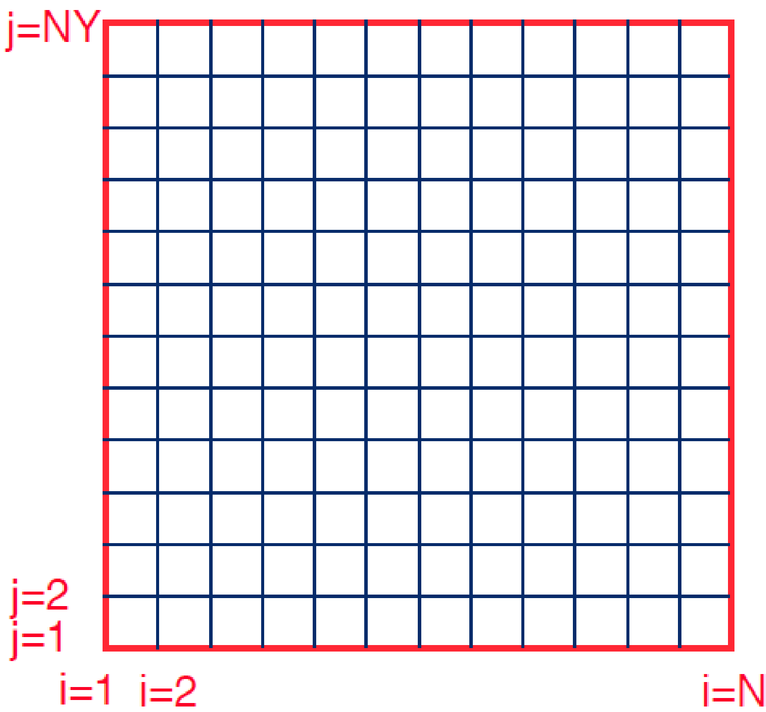
$$\frac{\partial u}{\partial x} \doteq 0$$



Boundary Conditions

$u_{i,j} = u(x, y)$
stored at each grid point

- When the solution u is given, we simply specify (*Dirichlet condition*)
- Where the normal derivative (*Neumann condition*) is specified, we approximate the value at the boundary by one-sided differences

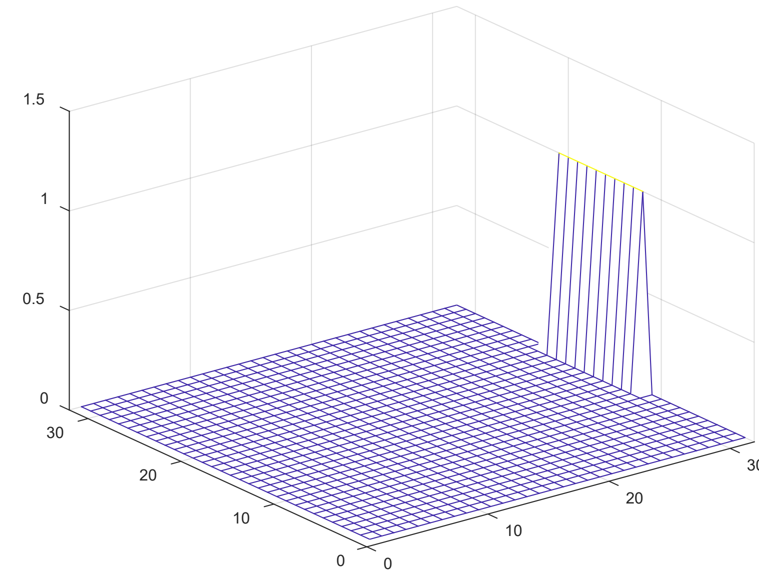


At $i = 1$ boundary, for example, $\dfrac{\partial u}{\partial y} = 0$

and by using $\dfrac{\partial u}{\partial y} \approx \dfrac{u_{i,2}^{n} - u_{i,1}^{n}}{h} = 0$

we obtain: $u_{i,2}^{n} = u_{i,1}^{n}$

```matlab
% MECH 479 - CFD
% EX3: Two-dimensional unsteady diffusion by the FTCS scheme
%------------------------------------------------------------
n=32;
m=32;
nstep=120;
D=0.025;
length=2.0;
h=length/(n-1);
dt=1.0*0.125*h*h/D;
u=zeros(n,m);
uo=zeros(n,m);
time=0.0;
U=-0.0; V=-1.0; u(12:21,n)=1.0;
for l=1:nstep,l,time
hold off;mesh(u); axis([0 n 0 m 0 1.5]);pause;
uo=u;
for i=2:n-1, for j=2:m-1
u(i,j)=uo(i,j)-(0.5*dt*U/h)*(uo(i+1,j)-uo(i-1,j))-...
(0.5*dt*V/h)*(uo(i,j+1)-uo(i,j-1))+...
(D*dt/h^2)*(uo(i+1,j)+ uo(i,j+1)+uo(i-1,j)+uo(i,j -1)-4*uo(i,j));
end,end
for i=1:n,
    u(i,1)=u(i,2);
end
for j=1:m,
    u(1,j)=u(2,j);
    u(m,j)=u(m-1,j);
end
time=time+dt;
end
```



Unsteady evolution of the solution

Multidimensional Steady Boundary Value Problems

- Consider Steady State Poisson Equation

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

The equation has a solution if u or $\partial u / \partial n$ is given on the boundary.

Using finite difference approximation for uniform grid:

$$\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} = S_{i,j}$$

Solve for $u_{i,j}$

$$\boxed{u_{i,j} = \frac{1}{4}\left(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - h^2 S_{i,j}\right)}$$

Iterative Solution Procedure

Solve for $u_{i,j}$ and use the right hand side to compute a new value

Denote the previous values by α and the new values with $\alpha+1$

$$u_{i,j}^{\alpha+1} = \frac{1}{4}\left(u_{i+1,j}^{\alpha} + u_{i-1,j}^{\alpha} + u_{i,j+1}^{\alpha} + u_{i,j-1}^{\alpha} - h^2 S_{i,j}\right)$$

The iteration process (Jacobi-type) can be employed.

To measure the error, define residual:

$$R_{i,j} = \frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}}{h^2} - S_{i,j}$$

At steady state, the residual should approach to zero.

Jacobi Iteration

- The iteration must be carried out until the solution is sufficiently accurate. To measure the error, define the residual:

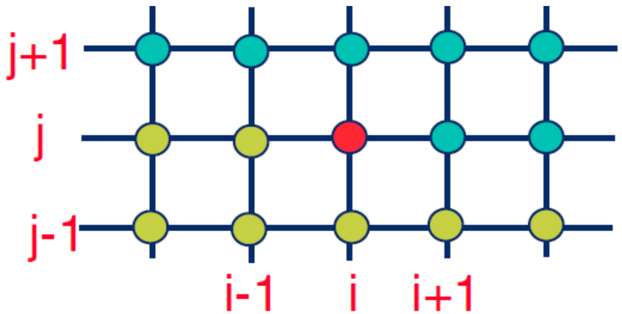
$$R_{i,j} = \frac{u_{i+1,j} + u_{i-1,j} + u_{i,j-1} + u_{i,j+1} - 4u_{i,j}}{h^2} - S_{i,j}$$

- At steady-state the residual should be zero.
  - The pointwise residual or average absolute residual can be used, depending on the problem.

Gauss Seidel Iteration

- Jacobi iteration is generally robust but many iterations are required to reach an accurate solution
  - ► Need a way to accelerate the convergence

- Using Gauss-Seidel, the Jacobi iteration can be improved somewhat by using new values as soon as they become available.



$$u_{i,j}^{\alpha+1} = \frac{1}{4}\left(u_{i+1,j}^{\alpha} + u_{i-1,j}^{\alpha+1} + u_{i,j+1}^{\alpha} + u_{i,j-1}^{\alpha+1} - h^2 S_{i,j}\right)$$

- Gauss-Seidel iteration can be further improved by Successive Over Relaxation (SOR)

Successive Over Relaxation

- Gauss-Seidel iteration can be further improved by SOR treatment

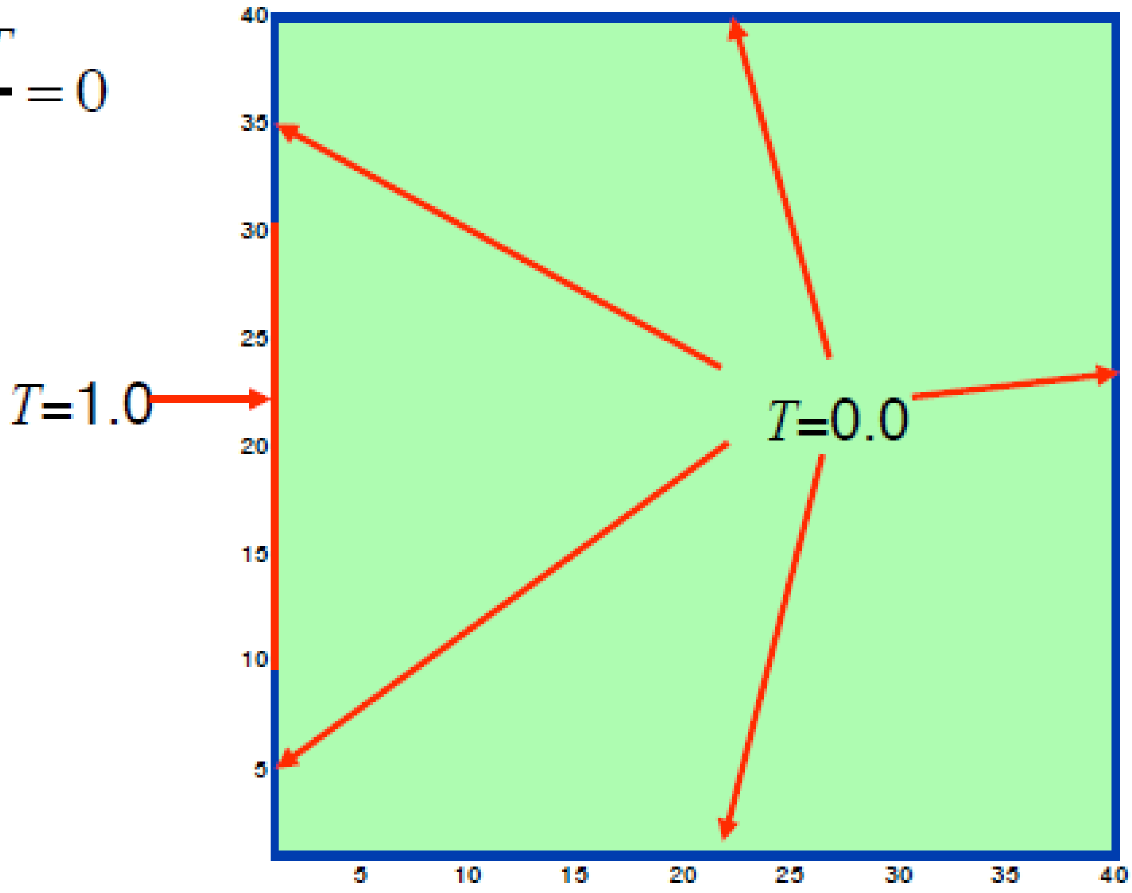
$$u_{i,j}^{\alpha+1} = \frac{\beta}{4}\left(u_{i+1,j}^{\alpha} + u_{i-1,j}^{\alpha+1} + u_{i,j+1}^{\alpha} + u_{i,j-1}^{\alpha+1} - h^2 S_{i,j}\right) + \left(1-\beta\right)u_{i,j}^{\alpha}$$

where $1 < \beta < 2$. In general, $\beta = 1.5$ is a good starting value.

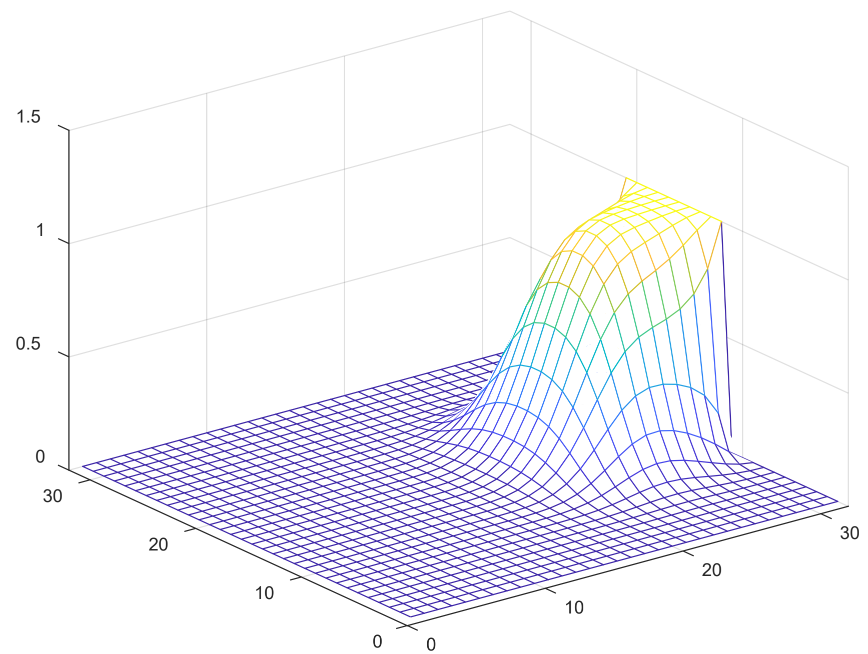
- The SOR iteration is very simple to program, just as the Gauss-Seidler iteration.

Example

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0$$



```
% MECH 479 - CFD
% EX4: Two-dimensional steady heat problem by SOR
%------------------------------------------------------------
-
n=40;
m=40;
iterations=5000;
length=2.0;
h=length/(n-1);
T=zeros(n,m);
bb=1.7;
T(10:n-10,1)=1.0;
for l=1:iterations,
for i=2:n-1, for j=2:m-1
T(i,j)=bb*0.25*(T(i+1,j)+...
T(i,j+1)+T(i-1,j)+T(i,j-1))+(1.0-bb)*T(i,j);
end,end
% find residual
res=0;
for i=2:n-1,
    for j=2:m-1
        res=res+abs(T(i+1,j)+...
        T(i,j+1)+T(i-1,j)+T(i,j-1)-4*T(i,j))/h^2;
    end
end
l,res/((m-2)*(n-2)) % Print iteration and residual
if (res/((m-2)*(n-2)) < 0.001),
    break
end
end;
contour(T);
```



Summary

- ❑ FDEs for multi-dimensional advection-diffusion are similar to 1D problem

- ❑ Iterative methods for boundary value problems. Elementary approaches to steady state problems
  - ► Jacobi iteration
  - ► Gauss-Seidel iteration
  - ► Successive Over-Relaxation