Topics on PDEs and Numerical Methods

Part 2: Finite Difference Method

Guangming Yao
Department of Mathematics
Clarkson University

Finite Difference Method (FDM)

Questions:

- How to solve PDEs using FDM?
  - Numerical differentiation
  - Domain discretization
  - PDE discretization
  - Linear or nonlinear system solver
- What are your choices?
  - Explicit
  - Implicit
  - Crank Nicolson

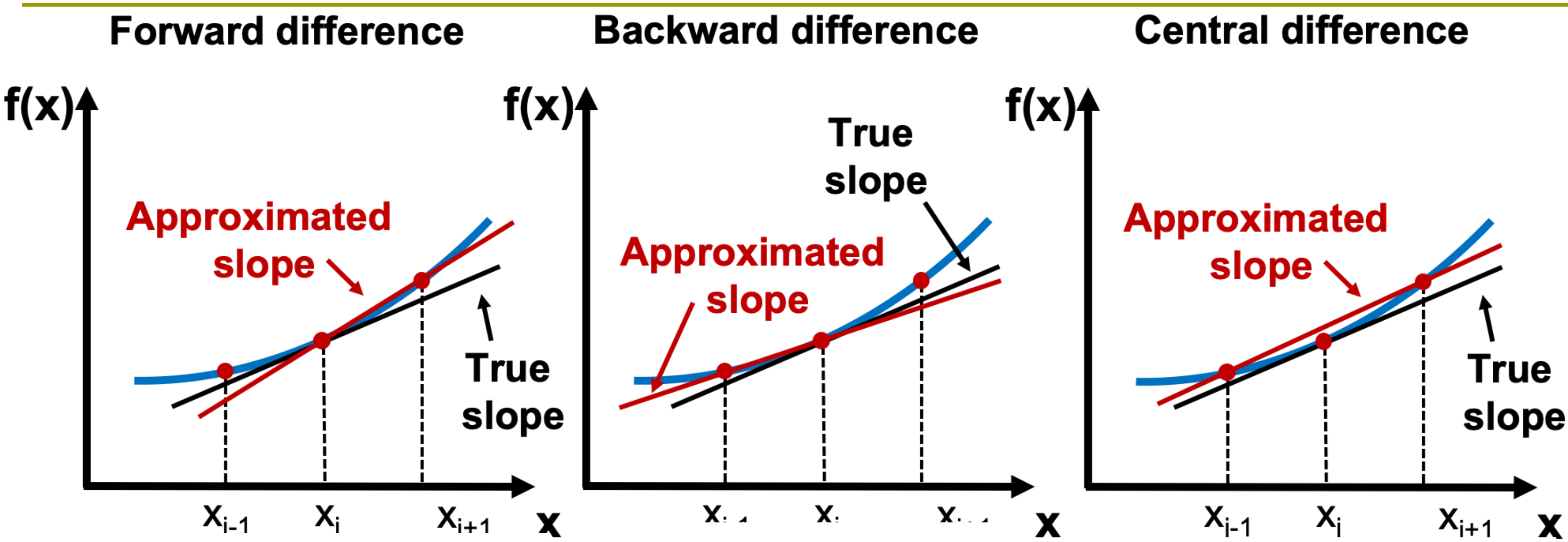
- Discretize PDE by using FDM
- Approximate solutions to PDE at discretized points in domain and on boundary.

Note: Methods other than FDM
FEM, Spectral Method, FVM, …

The Solution Methods for PDEs

- Analytic solutions are possible for simple and special (idealized) cases only.
- To make use of the nature of the equations, different methods are used to solve different classes of PDEs.
- The methods discussed here are based on the finite difference technique.
- With initial-value problem, solution is obtained by starting with intial values along boundary of problem domain, and marking toward in time step, generating successive rows in solution table
- Time-stepping procedure may be explicit or implicit depending on whether formula for solution values at next time step involves only past info
- Good accuracy may be obtained by taking sufficiently small step size in time and space
- Time and space step sizes cannot always be chosen independently of each other.

Step 0: Numerical differentiation



$$f'(x_j) = \frac{f(x_{j+1}) - f(x_j)}{h} + O(h).$$

$$f'(x_j) \approx \frac{f(x_j) - f(x_{j-1})}{h},$$

$$f'(x_j) \approx \frac{f(x_{j+1}) - f(x_{j-1})}{2h}.$$

Central difference formula for 1st and 2nd derivatives

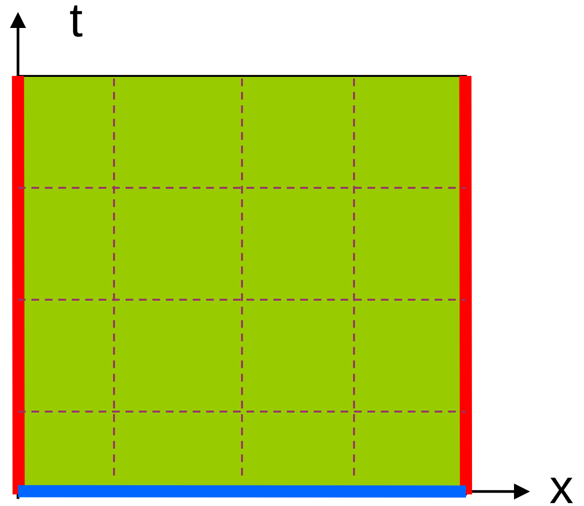
$$f'(x) \simeq \frac{f(x + \Delta x) - f(x - \Delta x)}{2\Delta x}$$

•

$$f''(x) \cong \frac{f(x + \Delta x) - 2f(x) + f(x - \Delta x)}{(\Delta x)^2}$$

FDM: Step 1. Discretize the domain

- Divide the interval x into sub-intervals, each of width h or Δx
- Divide the interval t into sub-intervals, each of width k or Δt
- A grid of points is used for the finite difference solution
- $T_{i,j}$ represents $T(xi, t_j)$
- Replace the derivatives by FDM



FDM Step 2. Discretize the PDE

Replace the derivatives by finite difference formulas:

- CentralDifferenceFormula for $\frac{\partial^2 T}{\partial x^2}$ $at\ (x_i, t_j)$:

$$\frac{\partial^2 T(x_i, t_j)}{\partial x^2} \approx \frac{T_{i-1,j} - 2T_{i,j} + T_{i+1,j}}{(\Delta x)^2} = \frac{T_{i-1,j} - 2T_{i,j} + T_{i+1,j}}{h^2}$$

- ForwardDifference Formula for $\frac{\partial T}{\partial t}$ $at\ (x_i, t_j)$:

$$\frac{\partial T(x_i, t_j)}{\partial t} \approx \frac{T_{i,j+1} - T_{i,j}}{\Delta t} = \frac{T_{i,j+1} - T_{i,j}}{k}$$

Example: heat equation
Explicit Method

Consider heat equation

$$u_t = c\,u_{xx}, \qquad 0 \le x \le 1, \qquad t \ge 0$$

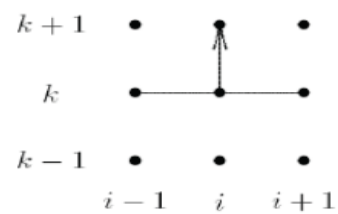
with initial and boundary conditions

$$u(0,x) = f(x), \qquad u(t,0) = \alpha, \qquad u(t,1) = \beta$$

- Step 1. Define spatial mesh points $x_i = i\Delta x, i = 0, 1, \cdots, n+1,$ where $\Delta x = \frac{1}{n+1}$ and temporal mesh points $t_k = k\Delta t$ for suitably chosen Δt
- Step 2. Let u_i^k denote approximate solution at (t_k, x_i) , forward difference in time and central difference in space leads discretized heat equation:

$$\frac{u_i^{k+1} - u_i^k}{\Delta t} = c\,\frac{u_{i+1}^k - 2u_i^k + u_{i-1}^k}{(\Delta x)^2},$$

$$u_i^{k+1} = u_i^k + \frac{c\Delta t}{(\Delta x)^2}(u_{i+1}^k - 2u_i^k + u_{i-1}^k), i = 1, 2, \cdots, n$$



Stencil: pattern of mesh points involved at each level

- BCs: $u_0^k = \alpha, u_{n+1}^k = \beta$ for all k
- ICs: $u_i^0 = f(x_i), i = 1, 2, \cdots, n$ gives starting values

- March numerical solution forward in time using this explicit scheme
- Local truncation error: $O((\Delta t) + (\Delta x)^2)$,

  1st order accurate in time, 2nd order accurate in space
- Let $\lambda = \frac{c\Delta t}{(\Delta x)^2}$, error can be magnified(unstable). Stable condition: $\lambda \le \frac{1}{2}$. This means that k is much smaller than h. This makes it slow.

Convergence and Stability of the Solution

- Convergence

  The solutions converge means that the solution obtained using the finite difference method approaches the true solution as the steps Δx and Δt approach zero.

- Stability:

  An algorithm is stable if the errors at each stage of the computation are not magnified as the computation progresses: approximated solution at any fixed time must remain bounded when Δt and Δx go to zero.

- Accuracy:

  Difference between u_i^k and $u(t_k, x_i)$ cannot be too big

- Efficiency:

  Minimize the computational time.

- Consistency:

  local truncation error go to zero when Δt and Δx go to zero.

Heat equation, cont. Implicit FDM

- Implicit method offers much greater stability, which implies larger time steps than explicit method
- It requires more work per step b/c system of equations must be solved at each step
- For heat equation in 1D, the linear system is tridiagonal, thus the work and storage required are modest
- In higher dimensions, matrix of linear system does not have much simple form, but it is still very sparse, with nonzeros in regular pattern
- Lax Equivalence Theorem: For well-posed linear PDEs, consistency and stability are necessary and sufficient for convergence.

Example: heat equation
Implicit Method

Consider heat equation

$$u_t = c\,u_{xx}, \qquad 0 \le x \le 1, \qquad t \ge 0$$

with initial and boundary conditions

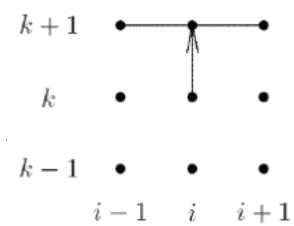
$$u(0,x) = f(x), \qquad u(t,0) = \alpha, \qquad u(t,1) = \beta$$

- Step 1. Define spatial mesh points $x_i = i\Delta x, i = 0, 1, \cdots, n+1,$ where $\Delta x = \frac{1}{n+1}$ and temporal mesh points $t_k = k\Delta t$ for suitably chosen Δt
- Step 2. Let u_i^k denote approximate solution at (t_k, x_i) , backward difference in time and central difference in space leads discretized heat equation:

$$\frac{u_i^{k+1} - u_i^k}{\Delta t} = c\,\frac{u_{i+1}^{k+1} - 2u_i^{k+1} + u_{i-1}^{k+1}}{(\Delta x)^2},$$

$$\lambda u_{i+1}^{k+1} - (1+2\lambda)u_i^{k+1} + \lambda u_{i-1}^{k+1} = -u_i^k, \qquad i = 1, 2, \cdots,$$

where $\lambda = \frac{c\Delta t}{(\Delta x)^2}$.



Stencil: pattern of mesh points involved at each level

- BCs: $u_0^k = \alpha, u_{n+1}^k = \beta$ for all k
- ICs: $u_i^0 = f(x_i), i = 1, 2, \cdots, n$ gives starting values

- Solve the system of equation at each time step makes it implicit scheme
- Local truncation error: $O(\Delta t) + (\Delta x)^2$,

  1st order accurate in time, 2nd order accurate in space
- Unconditionally stable.

Image blurring and deblurring using heat equation

Original image

Blurred

Deblurred

Example: wave equation

- Consider wave equation

$$u_{tt} = c\, u_{xx}, \qquad 0 \le x \le 1, \qquad t \ge 0$$

with initial and boundary conditions

$$u(0,x) = f(x), \qquad u_t(0,x) = g(x)$$

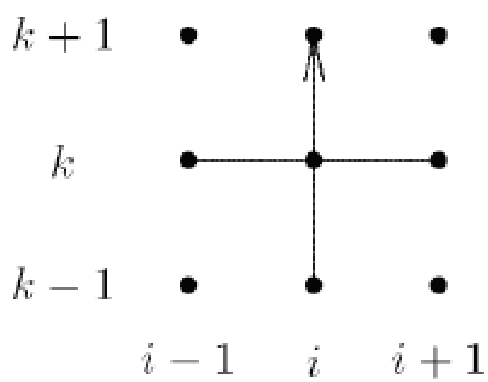
$$u(t,0) = \alpha, \qquad u(t,1) = \beta$$

Example: wave equation

- With mesh points defined as before, using centered difference formulas for both u_{tt} and u_{xx} gives finite difference scheme

$$\frac{u_i^{k+1} - 2u_i^k + u_i^{k-1}}{(\Delta t)^2} = c\,\frac{u_{i+1}^k - 2u_i^k + u_{i-1}^k}{(\Delta x)^2}, \qquad \text{or}$$

$$u_i^{k+1} = 2u_i^k - u_i^{k-1} + c\left(\frac{\Delta t}{\Delta x}\right)^2 \left(u_{i+1}^k - 2u_i^k + u_{i-1}^k\right),\ i = 1, \ldots, n$$



Wave equation

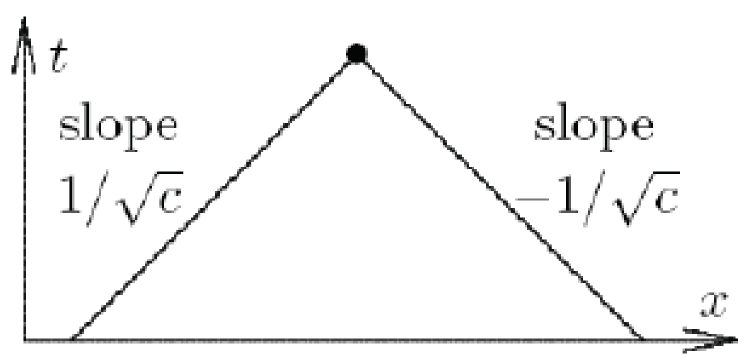
- Using data at two levels in time requires additional storage
- We also need u_i^0 and u_i^1 to get started, which can be obtained from initial conditions

$$u_i^0 = f(x_i), \qquad u_i^1 = f(x_i) + (\Delta t)g(x_i)$$

where latter uses forward difference approximation to initial condition $u_t(0, x) = g(x)$

Wave equation

- Consider explicit finite difference scheme for wave equation given previously
- Characteristics of wave equation are straight lines in (t, x) plane along which either $x + \sqrt{c}\,t$ or $x - \sqrt{c}\,t$ is constant
- Domain of dependence for wave equation for given point is triangle with apex at given point and with sides of slope $1/\sqrt{c}$ and $-1/\sqrt{c}$

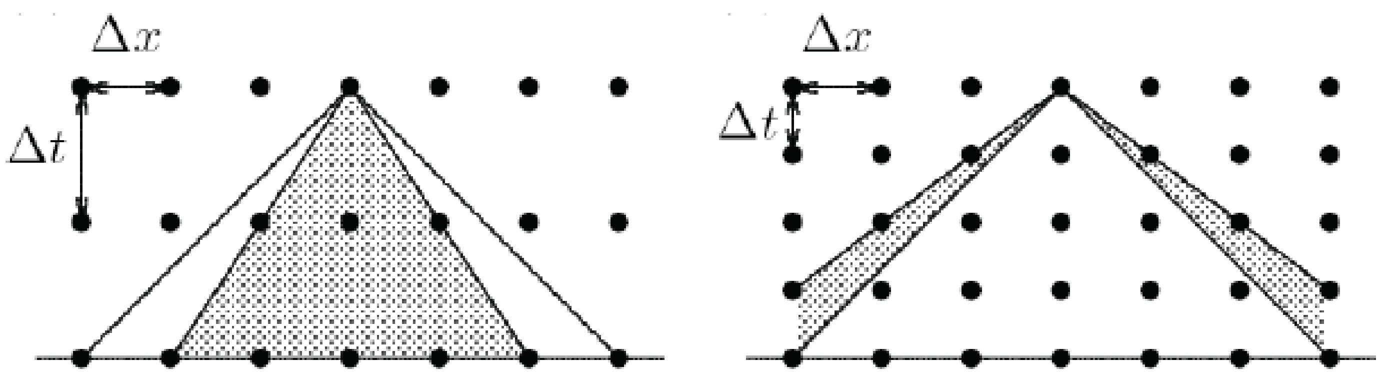


Wave equation

- CFL condition implies step sizes must satisfy

$$\Delta t \leq \frac{\Delta x}{\sqrt{c}}$$

for this particular finite difference scheme



unstable

stable

Time-independent problems: Laplace equations

- We next consider time-independent, elliptic PDEs in two space dimensions, such as *Helmholtz equation*

$$u_{xx} + u_{yy} + \lambda u = f(x, y)$$

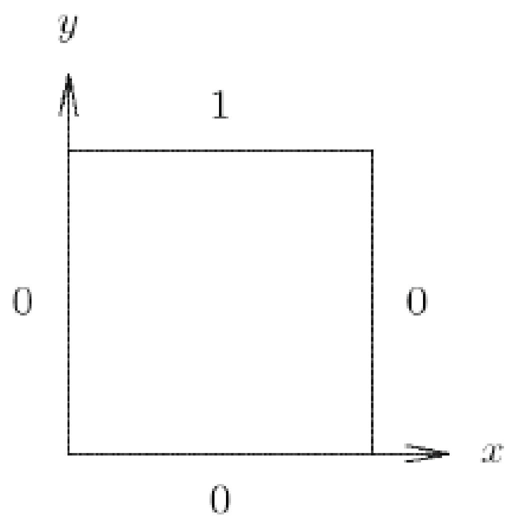
- Important special cases
  - *Poisson equation*: $\lambda = 0$
  - *Laplace equation*: $\lambda = 0$ and $f = 0$
- For simplicity, we will consider this equation on unit square
- Numerous possibilities for boundary conditions specified along each side of square
  - *Dirichlet*: u is specified
  - *Neumann*: u_x or u_y is specified
  - *Mixed*: combinations of these are specified

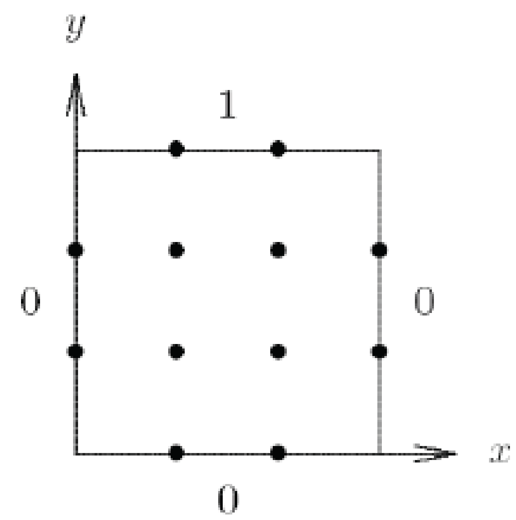
Example: Laplace equation

- Consider Laplace equation

$$u_{xx} + u_{yy} = 0$$

on unit square with boundary conditions shown below left





- Define discrete mesh in domain, including boundaries, as shown above right

Laplace equation, cont.

- Interior grid points where we will compute approximate solution are given by

$$(x_i, y_j) = (ih, jh), \qquad i, j = 1, \ldots, n$$

  where in example $n = 2$ and $h = 1/(n+1) = 1/3$

- Next we replace derivatives by centered difference approximation at each interior mesh point to obtain finite difference equation

$$\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} = 0$$

  where $u_{i,j}$ is approximation to true solution $u(x_i, y_j)$ for $i, j = 1, \ldots, n$, and represents one of given boundary values if i or j is 0 or $n+1$

Laplace equation, cont.

- Simplifying and writing out resulting four equations explicitly gives

$$4u_{1,1} - u_{0,1} - u_{2,1} - u_{1,0} - u_{1,2} = 0$$
$$4u_{2,1} - u_{1,1} - u_{3,1} - u_{2,0} - u_{2,2} = 0$$
$$4u_{1,2} - u_{0,2} - u_{2,2} - u_{1,1} - u_{1,3} = 0$$
$$4u_{2,2} - u_{1,2} - u_{3,2} - u_{2,1} - u_{2,3} = 0$$

Laplace equation, cont.

- Writing previous equations in matrix form gives

$$\boldsymbol{A}\boldsymbol{x} = \begin{bmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix} \begin{bmatrix} u_{1,1} \\ u_{2,1} \\ u_{1,2} \\ u_{2,2} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} = \boldsymbol{b}$$

- System of equations can be solved for unknowns $u_{i,j}$ either by direct method based on factorization or by iterative method, yielding solution

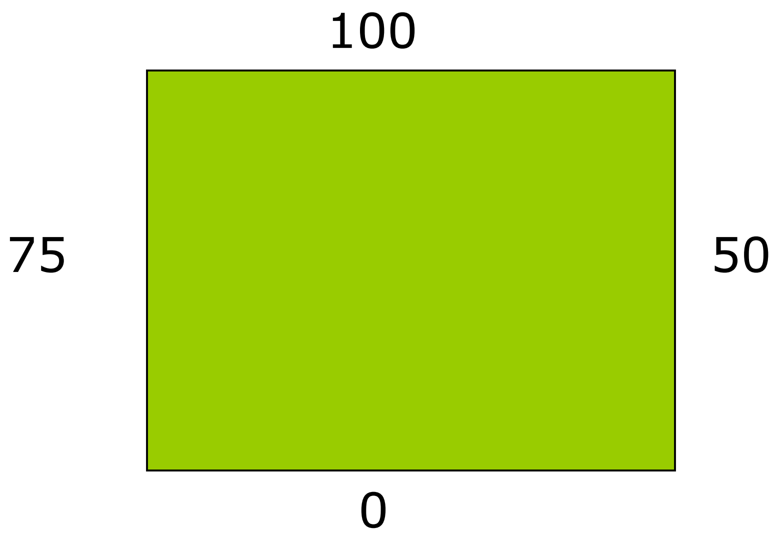
$$\boldsymbol{x} = \begin{bmatrix} u_{1,1} \\ u_{2,1} \\ u_{1,2} \\ u_{2,2} \end{bmatrix} = \begin{bmatrix} 0.125 \\ 0.125 \\ 0.375 \\ 0.375 \end{bmatrix}$$

Laplace equation, cont.

- In practical problem, mesh size h would be much smaller, and resulting linear system would be much larger
- Matrix would be very sparse, however, since each equation would still involve only five variables, thereby saving substantially on work and storage

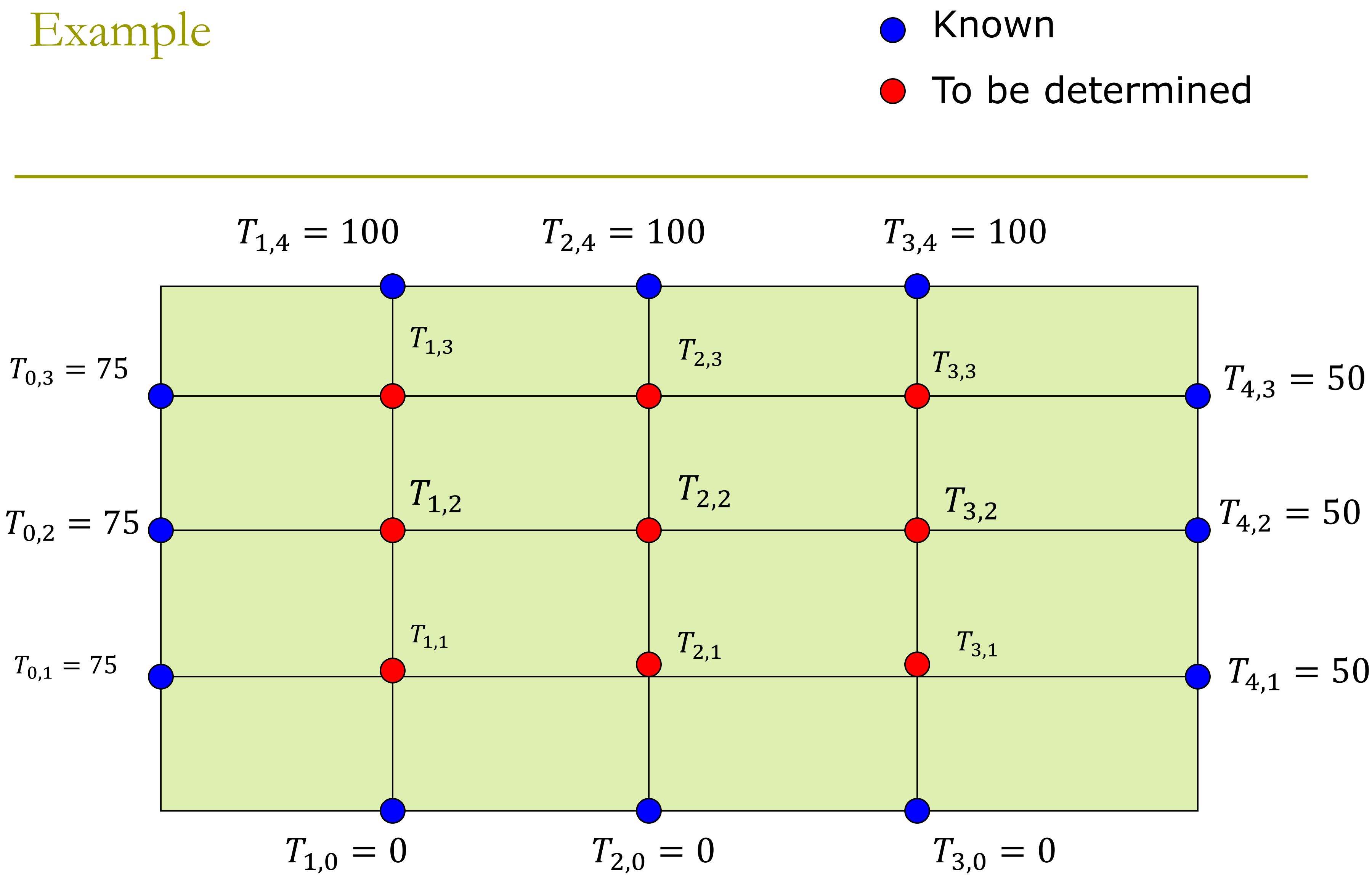
Example

It is required to determine the steady state temperature at all points of a heated sheet of metal. The edges of the sheet are kept at a constant temperature: 100, 50, 0, and 75 degrees.



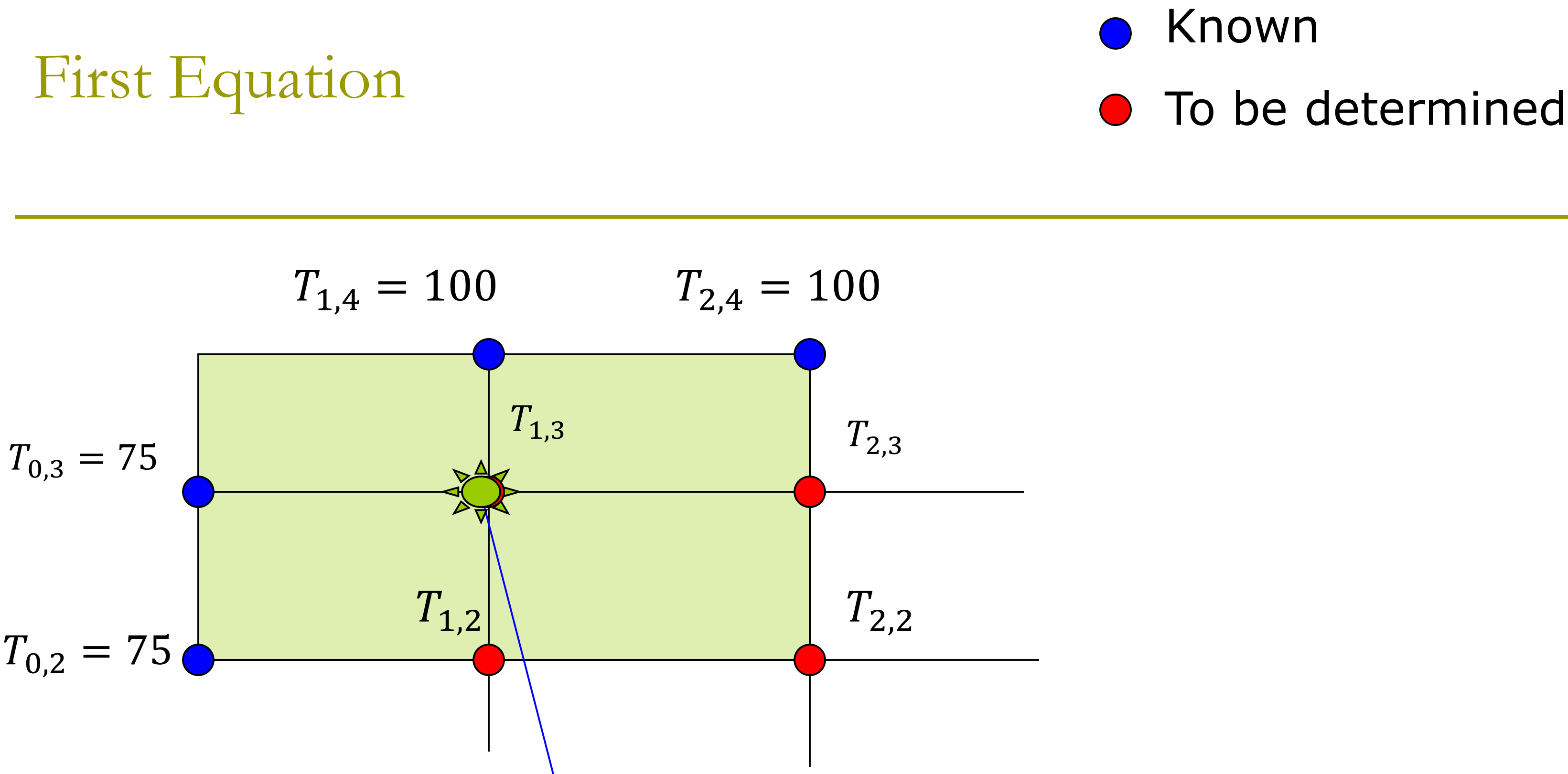
The sheet is divided to 5X5 grids.

Example



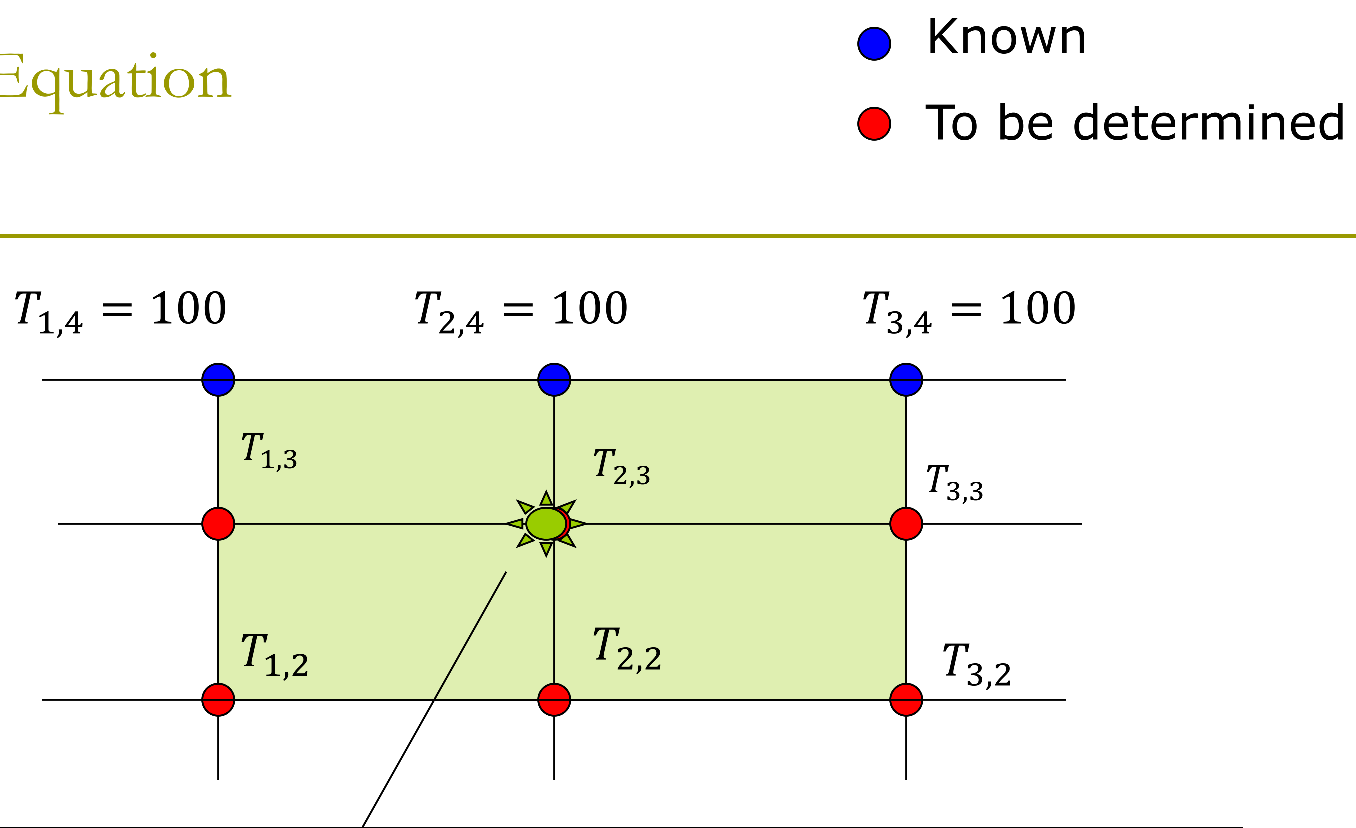
Known
To be determined
$T_{1,4} = 100$
$T_{2,4} = 100$
$T_{3,4} = 100$
$T_{1,3}$
$T_{2,3}$
$T_{3,3}$
$T_{0,3} = 75$
$T_{4,3} = 50$
$T_{1,2}$
$T_{2,2}$
$T_{3,2}$
$T_{0,2} = 75$
$T_{4,2} = 50$
$T_{1,1}$
$T_{2,1}$
$T_{3,1}$
$T_{0,1} = 75$
$T_{4,1} = 50$
$T_{1,0} = 0$
$T_{2,0} = 0$
$T_{3,0} = 0$

First Equation



$$T_{0,3} + T_{1,4} + T_{1,2} + T_{2,3} - 4T_{1,3} = 0$$
$$75 + 100 + T_{1,2} + T_{2,3} - 4T_{1,3} = 0$$

Another Equation



$$T_{1,3} + T_{2,4} + T_{3,3} + T_{2,2} - 4T_{2,3} = 0$$
$$T_{1,3} + 100 + T_{3,3} + T_{2,2} - 4T_{2,3} = 0$$

Solution: the Rest of the Equations

$$\begin{pmatrix} 4 & -1 & 0 & -1 & & & & & \\ -1 & 4 & -1 & 0 & -1 & & & & \\ 0 & -1 & 4 & 0 & 0 & -1 & & & \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & & \\ & -1 & 0 & -1 & 4 & -1 & 0 & -1 & \\ & & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ & & & -1 & 0 & 0 & 4 & -1 & 0 \\ & & & & -1 & 0 & -1 & 4 & -1 \\ & & & & & -1 & 0 & -1 & 4 \end{pmatrix} \begin{pmatrix} T_{1,1} \\ T_{2,1} \\ T_{3,1} \\ T_{1,2} \\ T_{2,2} \\ T_{3,2} \\ T_{1,3} \\ T_{2,3} \\ T_{3,3} \end{pmatrix} = \begin{pmatrix} 75 \\ 0 \\ 50 \\ 75 \\ 0 \\ 50 \\ 175 \\ 100 \\ 150 \end{pmatrix}$$

Finite difference methods

- Finite difference methods for such problems proceed as before
  - Define discrete mesh of points within domain of equation
  - Replace derivatives in PDE by finite difference approximations
  - Seek numerical solution at mesh points
- Unlike time-dependent problems, solution is not produced by marching forward step by step in time
- Approximate solution is determined at all mesh points simultaneously by solving single system of algebraic equations

Stability

- Unlike Method of Lines, where time step is chosen automatically by ODE solver, user must choose time step Δt in fully discrete method, taking into account both accuracy and stability requirements
- For example, fully discrete scheme for heat equation is simply Euler's method applied to semidiscrete system of ODEs for heat equation given previously
- We saw that Jacobian matrix of semidiscrete system has eigenvalues between $-4c/(\Delta x)^2$ and 0, so stability region for Euler's method requires time step to satisfy

$$\Delta t \le \frac{(\Delta x)^2}{2\,c}$$

Severe restriction on time step can make explicit methods relatively inefficient

Remarks

The Explicit Method:

- One needs to select small k to ensure **stability.**
- Computation per point is very simple but many points are needed.

The Implicit Method:

- Requires the solution of a **Tridiagonal** system.
- Stable (Larger Δt can be used).