Finite Element Method I
Finite Element Method

In Finite Difference Methods:
  → the solution domain is divided into a grid of discrete points or nodes
  → the PDE is then written for each node and its derivatives replaced by finite-divided differences

In Finite Element Method:
  → the solution domain is divided into shaped regions, or “elements”
  → an approximate solution for the PDE is developed for each of these elements
  → the total solution is generated by assembling the individual solutions, ensuring continuity at boundaries
Finite Element Method

Suppose $\Omega$ be a bounded domain in 3D-space $\mathbb{R}^3$ or the 2D-plane $\mathbb{R}^2$, with boundary $\Gamma = \partial \Omega$, with given:

- heat capacity $c = c(x)$
- conductivity $a = a(x)$
- heat production $f = f(x, t)$

\begin{equation}
\{ \text{in } \Omega \} \quad (23.1)
\end{equation}

Our aim is to seek

- temperature $u = u(x, t)$
- heat flow $q = -a \nabla u$

\begin{equation}
\{ \text{in } \Omega \} \quad (23.2)
\end{equation}

determined by

- the initial condition $u = u_0$ for $t = 0$
- the boundary condition $-a \partial_n u = \gamma(u - g_D) + g_N$

\begin{equation}
(23.3)
\end{equation}
Finite Element Method

The differential equation
\[ c u_t = -\nabla \cdot q + f \]  
(23.4)
expresses conservation of energy, that is, heat energy produced minus heat energy lost through outflow is accumulated. If we take \( c = 1 \), equation (23.4) can be written
\[ u_t = \nabla \cdot (a \nabla u) + f \]  
(23.5)

Typical units for
\[ a = \text{Watts/(meter Kelvin)} \]
\[ f = \text{Watts/meter}^3 \]
Finite Element Method

For time discretization of the problem, we introduce:

For space discretization of the problem, using finite elements, we introduce a triangulation of $\Omega$: 
Then the approximate solution of $u(x)$ is a piecewise linear $U_n(x)$ which can be represented in terms of the basis functions $\phi_j(x)$ and (unknown) node values $U_{n,j} = U_n(x_j)$ with formula

$$U_n(x) = \sum_{j=1}^{m} U_{n,j} \phi_j(x)$$  \hspace{1cm} (23.6)

Basis functions $\phi_j(x)$ are particular continuous piecewise linear functions on the given triangulation, and characterized by

$$\phi_j(x_j) = \begin{cases} 
1 & \text{for } i = j \\
0 & \text{for } i \neq j
\end{cases} \hspace{1cm} (23.7)$$
Finite Element Method

In order to find suitable node values $U_{n,j}$ for which the ansatz $U_n(x)$ (approximately) solves the given differential equation (23.5), we write the equation in weak form or variational form, that is by multiplying the equation with a so-called test functions $v = v(x, t)$ and integrate in space and time over $\Omega$ and the time interval.

This gives

$$\int_{I_n} \int_{\Omega} v u_t \, dV \, dt = \int_{I_n} \int_{\Omega} v (\nabla \cdot a \nabla u) \, dV \, dt + \int_{I_n} \int_{\Omega} v f \, dV \, dt$$

(23.8)
Finite Element Method

Review:
Let \( u \) and \( v \) be scalar functions and \( \mathbf{w} = (w_1, w_2, \ldots, w_d) \) be a vector-valued function of \( x = (x_1, x_2, \ldots, x_d) \), where

\[
\nabla v = \text{grad } v = \left( \frac{\partial v}{\partial x_1}, \frac{\partial v}{\partial x_2}, \ldots, \frac{\partial v}{\partial x_d} \right)
\]

\[
\nabla \cdot \mathbf{w} = \text{div } \mathbf{w} = \left( \frac{\partial w_1}{\partial x_1} + \frac{\partial w_2}{\partial x_2} + \ldots + \frac{\partial w_d}{\partial x_d} \right)
\]

\[
\nabla^2 v = \nabla \cdot \nabla v = \left( \frac{\partial^2 v}{\partial x_1^2} + \frac{\partial^2 v}{\partial x_2^2} + \ldots + \frac{\partial^2 v}{\partial x_d^2} \right)
\]
Finite Element Method

From the **Divergence Theorem**:

\[ \int_{\Omega} \nabla \cdot \mathbf{w} \, dV = \int_{\Gamma} \mathbf{w} \cdot \mathbf{n} \, dS \]  \hspace{1cm} (23.9)

where \( \mathbf{n} = (n_1, n_2, \ldots, n_d) \) is the outward unit normal to boundary \( \Gamma \). From the product rule between the scalar function \( v \) and the vector-valued function \( \mathbf{w} \):

\[ \nabla \cdot (\mathbf{w}v) = \mathbf{w} \cdot \nabla v + v(\nabla \cdot \mathbf{w}) \]  \hspace{1cm} (23.10)

Applying (23.10) to (23.9) we obtain the **Green’s formula**:

\[ \int_{\Omega} \mathbf{w} \cdot \nabla v \, dV = \int_{\Gamma} v(\mathbf{w} \cdot \mathbf{n}) \, dS - \int_{\Omega} v(\nabla \cdot \mathbf{w}) \, dV \]  \hspace{1cm} (23.11)
Finite Element Method

If \( \mathbf{w} = \nabla u \) the formula (23.11) becomes

\[
\int_{\Omega} \nabla u \cdot \nabla v \, dV = \int_{\Gamma} v(\nabla u \cdot \mathbf{n}) \, dS - \int_{\Omega} v(\nabla \cdot \nabla u) \, dV
\]

or

\[
\int_{\Omega} v(\nabla \cdot \nabla u) \, dV = \int_{\Gamma} v(\nabla u \cdot \mathbf{n}) \, dS - \int_{\Omega} \nabla v \cdot \nabla u \, dV
\]  
(23.12)

Relating to the second term on the right hand side of (23.8):

\[
\int_{\Omega} v(\nabla \cdot a \nabla u) \, dV = \int_{\Gamma} v(a \nabla u \cdot \mathbf{n}) \, dS - \int_{\Omega} \nabla v \cdot a \nabla u \, dV
\]  
(23.13)
Substituting (23.13) to (23.8) and due to the boundary conditions we obtain

$$\int_{I_n} \int_{\Omega} v\ u_t = \int_{I_n} \int_{\Omega} \nabla v \cdot a \nabla u - \int_{I_n} \int_{\Gamma} v\gamma u$$

$$+ \int_{I_n} \int_{\Omega} v\ f + \int_{I_n} \int_{\Gamma} v\ g \quad (23.14)$$

where $g = \gamma g_D - g_N$

For $v = v(x)$ and $\gamma = \gamma(x)$ ($t$-independent), we get

$$\int_{\Omega} v \int_{I_n} u_t = \int_{\Omega} \nabla v \cdot a \nabla \int_{I_n} u - \int_{\Gamma} v\gamma \int_{I_n} u$$

$$+ \int_{\Omega} v \int_{I_n} f + \int_{\Gamma} v \int_{I_n} g \quad (23.15)$$
Finite Element Method

where

\[ \int_{I_n} u \approx k_n u_n \quad \text{and} \quad \int_{I_n} u_t \approx u_n - u_{n-1} \]

\[ u_n = u(x, t_n) \]

Thus, we seek

\[ \int_{\Omega} v U_n + k_n \int_{\Omega} \nabla v \cdot a \nabla U_n + k_n \int_{\Gamma} v \gamma U_n = \]

\[ \int_{\Omega} v U_{n-1} + \int_{\Omega} v \int_{I_n} f + \int_{\Gamma} v \int_{I_n} g \quad \text{(23.16)} \]

where \( v = \phi_i, \quad i = 1, 2, \ldots, m \) which gives \( m \) equations for the \( m \) node values \( U_{n,j} \) in the ansatz \( U_n = U_n(x) \).
Finite Element Method

Start by computing

\[ \int_\Omega \phi_i U_n = \int_\Omega \phi_i(U_n,1\phi_1 + U_n,2\phi_2 + \ldots + U_n,m\phi_m) \]

\[ = \int_\Omega (U_n,1\phi_i\phi_1 + U_n,2\phi_i\phi_2 + \ldots + U_n,m\phi_i\phi_m) \]

\[ = U_n,1 \int_\Omega \phi_i\phi_1 + U_n,2 \int_\Omega \phi_i\phi_2 + \ldots + U_n,m \int_\Omega \phi_i\phi_m \]

\[ = M_{i1}U_n,1 + M_{i2}U_n,2 + \ldots + M_{im}U_n,m \]

where \( M \) is the mass matrix with elements \( M_{ij} = \int_\Omega \phi_i\phi_j \) and \( U_n \) is the column vector with node values \( U_{n,j} \).
Similarly,

\[ \int_{\Omega} \nabla \phi_i \cdot a \nabla U_n \]

\[ = \int_{\Omega} \nabla \phi_i \cdot a \nabla (U_{n,1} \phi_1 + U_{n,2} \phi_2 + \ldots + U_{n,m} \phi_m) \]

\[ = \int_{\Omega} U_{n,1} \nabla \phi_i \cdot a \nabla \phi_1 + \ldots + U_{n,m} \nabla \phi_i \cdot a \nabla \phi_m \]

\[ = U_{n,1} \int_{\Omega} a \nabla \phi_i \cdot \nabla \phi_1 + \ldots + U_{n,m} \int_{\Omega} a \nabla \phi_i \cdot \nabla \phi_m \]

\[ = A_{i1} U_{n,1} + A_{i2} U_{n,2} + \ldots + A_{im} U_{n,m} \]

where \( A \) is the diffusion matrix with \( A_{ij} = \int_{\Omega} a \nabla \phi_i \cdot \nabla \phi_j \)
Finite Element Method

And,

\[ \int_{\Gamma} \gamma \phi_i U_n = \int_{\Gamma} \gamma \phi_i (U_{n,1} \phi_1 + U_{n,2} \phi_2 + \cdots + U_{n,m} \phi_m) \]

\[ = U_{n,1} \int_{\Gamma} \gamma \phi_i \phi_1 + \cdots + U_{n,m} \int_{\Gamma} \gamma \phi_i \phi_m \]

\[ = K_{i1} U_{n,1} + K_{i2} U_{n,2} + \cdots + K_{im} U_{n,m} \]

The load vectors:

\[ \int_{\Omega} \phi_i \int_{I_n} f = F_{n,i} \]

\[ \int_{\Gamma} \phi_i \int_{I_n} g = G_{n,i} \]
Finite Element Method

After perform approximation for all terms, we get a system of linearized approximations

\[ MU_n + k_n AU_n + k_n KU_n = MU_{n-1} + F_n + G_n \]

or

\[ (M + k_n A + k_n K)U_n = b_n \]  \hspace{1cm} (23.17)

From which we can solve

\[ U_n = D \backslash b_n \]

where

\[ D = M + k_n A + k_n K \]
MATLAB’s pdetool Toolbox

MATLAB has a GUI toolbox which is called pdetool that is used to solve linear PDEs based on finite element method. Basically the PDE toolbox can be used to solve the following problems:

1) Elliptic PDEs
2) Parabolic PDEs
3) Hyperbolic PDEs
4) Eigenmode PDE
Example

Using MATLAB’s \texttt{pdeTool}, solve the following 2D problem for temperature diffusion over a plate:

\[
\frac{\partial u}{\partial t} = 10^{-4} \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)
\]

for \( 0 \leq x \leq 4, \quad 0 \leq y \leq 4, \quad 0 \leq t \leq 5000 \)

with the initial condition

\[
u(x, y, 0) = 0 \quad \text{for} \quad t = 0
\]

and boundary conditions

\[
u(x, y, t) = e^y \cos(x) - e^x \cos(y)
\]

for \( x = 0, \quad x = 4, \quad y = 0, \quad \text{and} \quad y = 4.\)
Example

1) On MATLAB’s Command Window, type “pdetool” to have the PDE toolbox window on the screen.

2) Adjust the ranges of the $x$-axis and the $y$-axis to [0 5] and [0 5] by clicking Axes Limits in the Options drop-down menu.

3) Click the button and create a rectangle domain. Then double-click in the rectangle and set the Left=0, Bottom=0, Width=4, and Height=4.

4) Click the button to get into the boundary mode. Input the boundary conditions accordingly.

5) Click the PDE button to get into the PDE mode, and specify the type of PDE and its parameters.
Example

6. Click on the triangle \( \Delta \) button to get into the mesh mode for creating the triangular mesh for the specified domain. To improve the accuracy of the solution you can successively refine the mesh by clicking the button on the right of the mesh button.

7. Solve the PDE by clicking on the \( = \) button.
References

2) Finite Element Methods Lecture Notes, K. Eriksson