

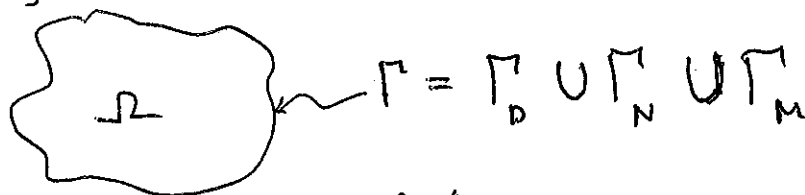
Mech 417 FEA Class #1 Summary

Integral formulations of most linear problems were given by Euler (1750). He proved that making special integrals stationary (min or max) was exactly the same as solving a partial differential equation, over the same domain, and other differential equations on portions of the boundary where the primary variable was not given.

Euler's theorem gives the process for finding the equivalent partial differential equation from the integral form.

A century or two later we learned to form integral formulations starting from a partial differential equation by using the method(s) of weighted residuals.

The FEM (finite element method) requires an integral formulation. It can yield ^{an} exact solution, but is usually a numerical approximation. Denote the arbitrary spatial domain as Ω and its boundary as Γ . The boundary has portions with essential boundary conditions, Γ_D , natural boundary conditions, Γ_N , and/or mixed boundary conditions, Γ_M .



The exact integral form, say I , is usually of the form

$$I = \int_{\Omega} f(\underline{x}) d\Omega + \int_{\Gamma_N} g(\underline{x}) d\Gamma + \int_{\Gamma_M} h(\underline{x}) d\Gamma$$

and its solution is obtained from

$$I = \begin{cases} \text{stationary,} & \text{if variational} \\ 0, & \text{if MWR} \end{cases}$$

Meshing: The FEM splits Ω into the union of finite sized, non-overlapping sub-regions (called finite elements). Thus, it also splits the boundary into non-overlapping segments:

$$\Omega = \bigcup_{e=1}^{n-e} \Omega^e \quad n-e \sim \text{number of elements}$$

$$\Gamma = \bigcup_{b=1}^{n-b} \Gamma^b \quad n-b \sim \text{number of segments}$$

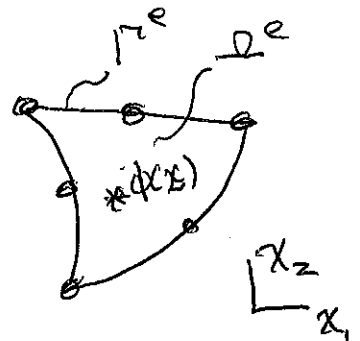
Assembly: This requires integration over the elements and segments of the mesh.

$$I = \sum_e \int_{\Omega^e} f(\underline{x}) d\Omega + \sum_{b_n} \int_{\Gamma_{N_b}^b} g(\underline{x}) d\Gamma + \sum_{b_m} \int_{\Gamma_{M_b}^b} h(\underline{x}) d\Gamma$$

Interpolation: The integrand $f(\underline{x})$ always includes spatial derivatives of the solution, and often includes the solution, say $\phi(\underline{x})$.

The finite element method assumes the spatial form of the solution within each finite element, Ω^e , and thus likewise assumes the form on the (lower spatial region) boundary of the element, Γ^e . Having assumed $\phi(\underline{x})$ for $\underline{x} \in \Omega^e$ then standard calculus will yield the gradient terms

$$\frac{\partial \phi(\underline{x})}{\partial x_j} \quad \text{for } k_j \leq n_s \text{ the physical space dimension}$$



and the boundary normal derivative $\frac{\partial \phi}{\partial n} = \vec{\nabla} \phi \cdot \vec{n}$.

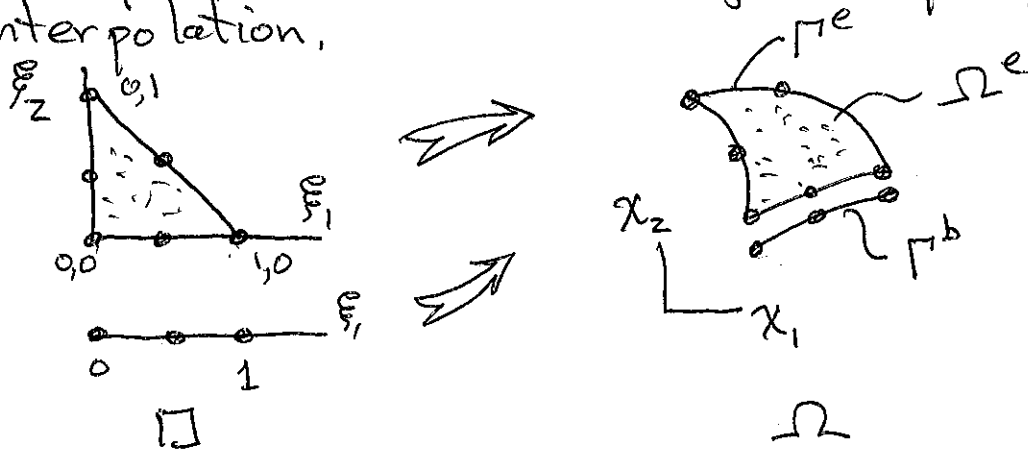
Therefore, all the terms in $f^e(\underline{x})$ (and $g^b(\underline{x})$ and $h^b(\underline{x})$) will be known and available to integrate.

Numerical Integration: Since the regions Ω and Ω^e can be arbitrary curved shapes the spatial integrals can be difficult to evaluate. We also want to automate this process, so we use numerical integration (quadrature).

$$I^e = \int_{\Omega^e} f(\underline{x}) d\Omega \approx \sum_{g=1}^{n_g} f(\underline{x}_g) W_g$$

\underline{x}_g ~ tabulated integration point location
 W_g ~ " " " weight.

Parametric Interpolation: The tabulate quadrature data are given in non-dimensional parametric spaces, say Ω . Also, to have compatible ϕ on a curved boundary we must use parametric interpolation instead of physical space, \underline{x} , interpolation.



Algebraic Solution: The mesh introduces node points on the elements that connect the elements to each other. The FEM solves for the solution at each of these node points. Let the unknowns be a vector $\underline{\Phi}$.

$$\underline{\Phi}^T = [\phi_1 \phi_2 \phi_3 \dots \phi_{n,m}] \quad n,m \sim \# \text{ nodes in the mesh}$$

Each element has some (a sub-set) of those unknowns, say $\underline{\Phi}^e$ with $\underline{\Phi}^e \subseteq \underline{\Phi}$.

$$\underline{\Phi}^{eT} = [\phi_1^e \phi_2^e \dots \phi_{n,n}^e] \quad n,n \sim \# \text{ nodes on the element,}$$

In each element the spatial approximate solution is interpolated from the element node values;

$$\phi(\underline{x}) = \phi(\underline{\xi}) = \underline{H}(\underline{\xi}) \underline{\Phi}^e = H_1(\underline{\xi}) \phi_1^e + \dots + H_{n,n}(\underline{\xi}) \phi_{n,n}^e$$

Assembly Figures: $\delta^e \equiv \beta^e \delta$

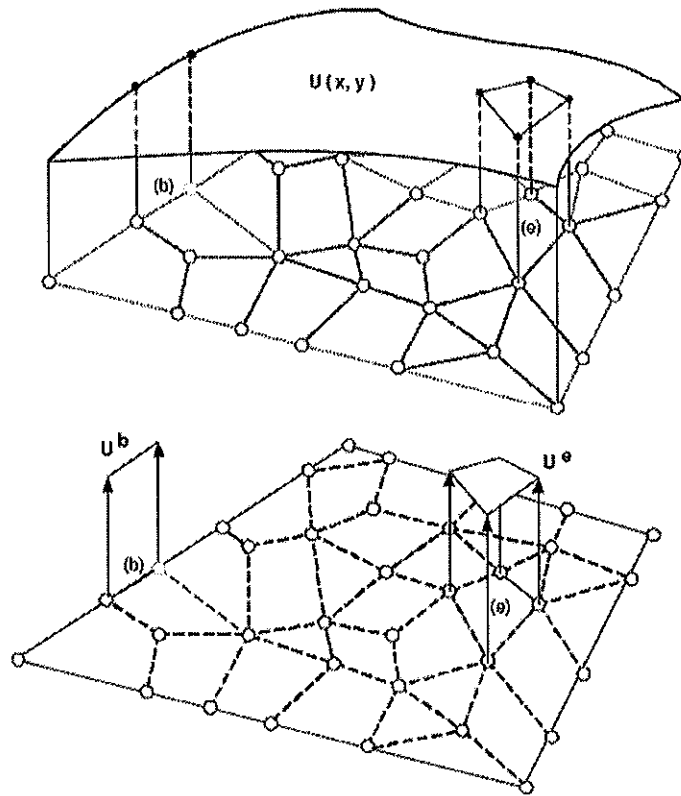


Figure 1.1 Piecewise approximation of a scalar function

Elements	(1)	(a)	...	(b)	(e)		
Mesh							
Nodes	1	3	6	7	6	4	2
Positions	x_1	x_3	x_6	x_7	x_6	x_4	x_2
Unknowns	D_1	D_3	D_6	D_7	D_6	D_4	D_2

$\beta^{(b)} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$
 $S D = C$

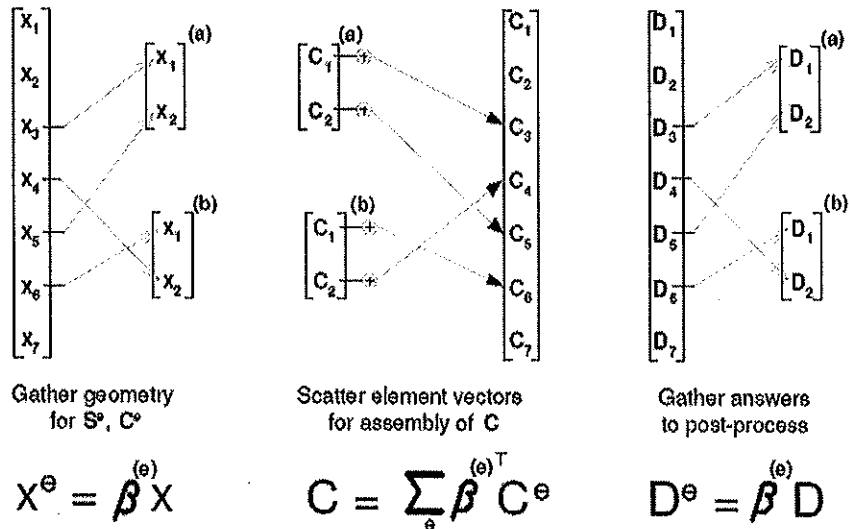
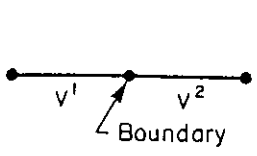
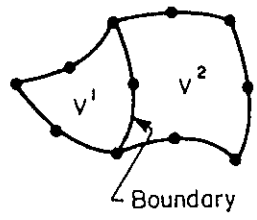


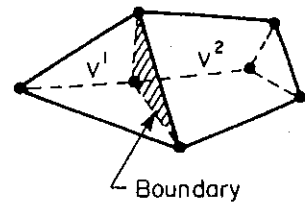
Figure 1.3 Gather and scatter concepts for finite elements



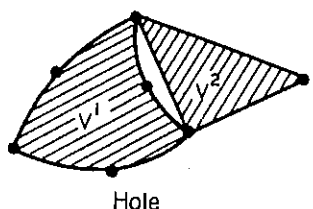
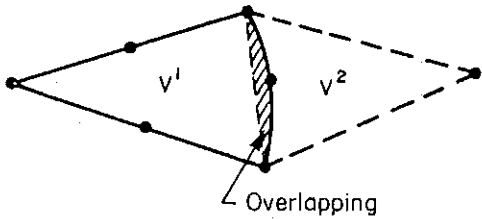
1 dimension



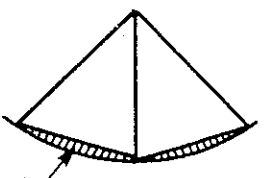
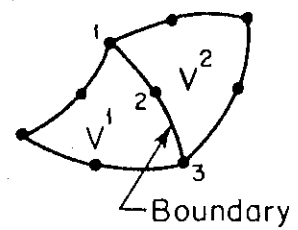
2 dimensions



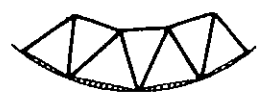
3 dimensions



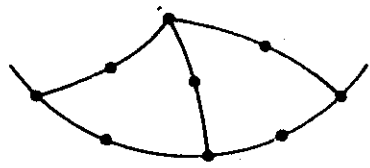
Hole



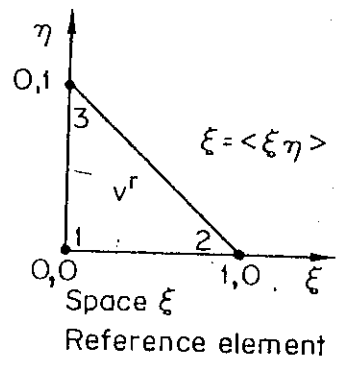
Discretization error



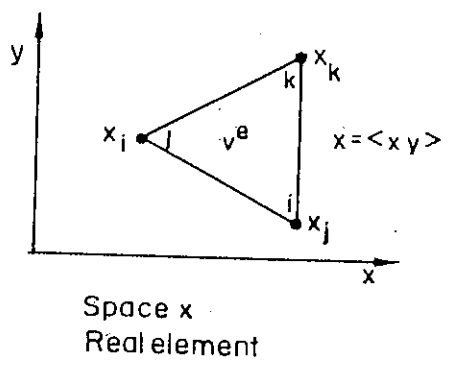
Increasing the number of elements



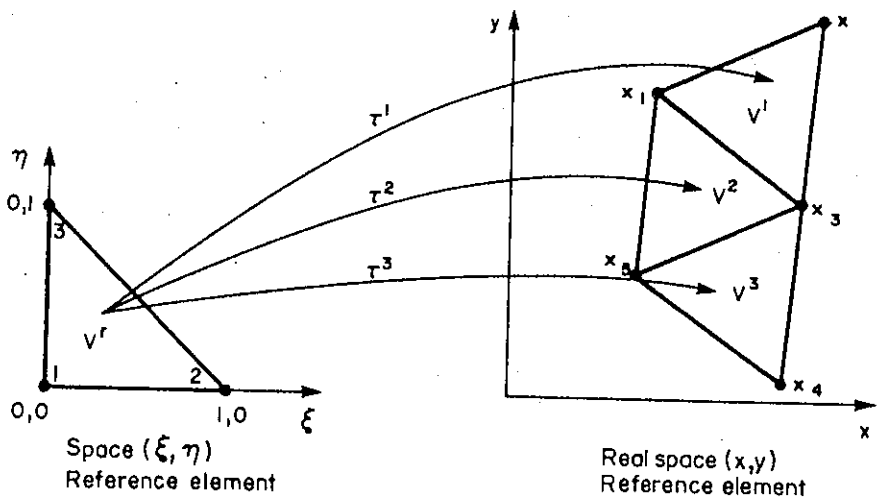
Using curved boundaries



Reference element



Real element



Reference element

Real element