ME40064: System Modelling & Simulation ME50344: Engineering Systems Simulation Lecture 10

Dr Andrew Cookson University of Bath, 2019-20

LECTURE 10 FEM: Basis Functions Revisited

- Deeper understanding of nodes and elements
- Ability to use spatially varying fields represented by basis functions
- Ability to implement numerical integration of FEM expressions

BASIS FUNCTIONS Why Are We Back Here Again?

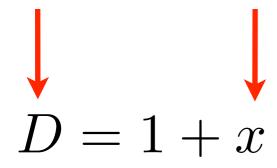
- Want to deal with spatially varying material parameters of different kinds
- Links with numerical integration procedures
- Linear basis may not be accurate enough
- Want to generalise our code as in realworld FEM codes

EVALUATING A FIELD A Coding Perspective

```
%MaterialFieldEvalScript.m
%Script to help write solution for Tutorial 5, Part B.Q2

msh=OneDimLinearMeshGen(0,1,10); %Setup a 1D mesh with 10 elements
CreateGQScheme(1); %Create a Gaussian Quadrature scheme of order 1

%Create a linearly varying diffusion coefficient field and store it as a
% vector, where each vector element corresponds to a global mesh node
msh.DCvec = 1 + msh.nvec;
```



xipt = gq.xipts(1); %Select a Gauss point to test the function at eID = 5; %Set an element ID to test the function for

%THIS IS THE FUNCTION YOU NEED TO WRITE - USE THESE INPUTS EvalField(msh,msh.DCvec,eID,xipt)

WHAT DOES EVALFIELD() NEED? A Coding Perspective

$$D(\xi) = d_0.\psi_0(\xi) + d_1.\psi_1(\xi) = d_n\psi_n(\xi)$$

Therefore need function to evaluate basis functions at a given $\boldsymbol{\xi}$

function [psi] = EvalBasis(lnid,xipt)
%UNTITLED7 Summary of this function goes here
% Detailed explanation goes here

$$\psi_0 = \frac{1-\xi}{2}, \quad \psi_1 = \frac{1+\xi}{2} \qquad \text{xipt}$$
 end

Local node ID: Inid

WHAT DOES EVALFIELD() NEED? A Coding Perspective

An example is given by the function for evaluating gradients of the basis functions:

```
function [ dpsidxi ] = EvalBasisGrad(lnid,xipt)
%EvalBasisGrad Returns gradient of basis functions
%    Returns the gradients of the linear Lagrange basis functions for a
%    specificed local node id (lnid = 0 or 1) and xipt (gradient is a constant
%    value in this case, but for higher order basis functions, would vary
%    with xi - and hence the argument is included for generality/future
proofing)

%Use the node id to generate the sign of the basis gradient - ie.
% either + or -. when lnid=0, sign is -ve, when lnid=1, sign is +ve.
sign = (-1)^(lnid+1);
dpsidxi = 0.5 * sign;

Could use if statement, but
```

Could use if statement, but this is mathematically elegant

WHAT DOES EVALFIELD() NEED? Some Pseudo-Code

```
function [ val ] = EvalField(msh,field,eID,xipt)
%EvalField Evaluates a nodally stored scalar field at a given xi point in
%an element
% Detailed explanation goes here
```

- 1. Evaluate the basis function values at the specified Gauss point: xipt
- 2. Find the local node values of the field in the element specified by: eID. Do this by finding the corresponding global node IDs and extract these values from the msh.DCvec vector variable
- 3. Perform the multiplication & sum to find the interpolated value

end

PUTTING IT TOGETHER Recap Of Integration Pseudo-Code

- 1. Initialise quadrature scheme number of points, generate weights and Gauss points
- 2. Initialise integral value to zero
- 3. Loop over number of Gauss points:
 - 1. At each Gauss point, xipt, call functions to evaluate:
 - 1. Basis functions & gradients (as appropriate)
 - 2. Material parameters using EvalField()
 - 2. Multiply together & multiply by matching Gauss weight
 - 3. Add to integral value