Lecture 14

Higher-order Finite Elements

14.1 Nodal Basis for Higher Order Elements

Until now, we have considered solutions which were allowed to vary at most linearly across an element. We now consider higher-order functions within the element. For simplicity, we will restrict our attention to one-dimensional problems. A p-th order polynomial has $p+1$ degrees of freedom, i.e. the coefficients of each term,

$$\tilde{T}(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_p x^p.$$

Thus, a general basis for a p-th order polynomial will require $p+1$ basis functions within an element,

$$\tilde{T}(x) = \sum_{i=1}^{p+1} a_i \phi_i(x) \quad \text{in an element.}$$

Building on the nodal basis approach described in Section 12.3 for linear elements, a common approach to choosing a basis for higher-order elements is to insert nodes within an element in addition to the nodes at the boundaries of the elements, specifically $p-1$ additional nodes internal to the element.

Let's consider building a nodal basis for quadratic elements. In this case, one additional node is added and this node is placed at the midpoint of the element. Using the one-dimensional reference element which extends from $-1 \leq \xi \leq 1$, this places nodes at $\xi = -1$, 0, and 1. The unknowns are assumed to be the values at these nodes,

$$a_1 = \tilde{T}(-1), \qquad a_2 = \tilde{T}(0), \qquad a_3 = \tilde{T}(1).$$

which leads to the following constraints on $\phi_i(\xi)$'s,

$$\phi_1(-1) = 1, \quad \phi_1(0) = 0, \quad \phi_1(1) = 0.$$

$$\phi_2(-1) = 0, \quad \phi_2(0) = 1, \quad \phi_2(1) = 0.$$

$$\phi_3(-1) = 0, \quad \phi_3(0) = 0, \quad \phi_3(1) = 1.$$

Applying these constraints and solving for the quadratic $\phi_i(\xi)$ gives,

$$\begin{aligned}\phi_1(\xi) &= -\frac{1}{2}\xi(1-\xi),\\ \phi_2(\xi) &= (1-\xi)(1+\xi),\\ \phi_3(\xi) &= \frac{1}{2}\xi(1+\xi).\end{aligned}$$

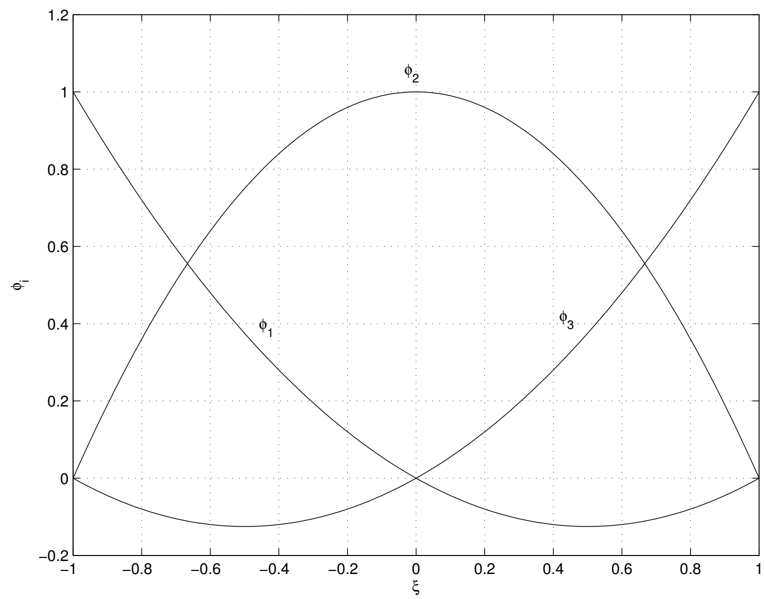


Figure 14.1: Nodal basis functions for a quadratic element with equally-spaced nodes.

In-class Discussion 14.1 (Construction of global $\phi(x)$ for quadratic elements)

14.2 Implementation of higher-order FEM

The implementation details are essentially the same as in the linear element case. Shown below is a Matlab script for a quadratic FEM using a nodal basis.

```
% FEM solver for d2T/dx2 + q = 0 where q = 50 exp(x)
%
% BC's: T(-1) = 100 and T(1) = 100.
%
% Note: the finite element degrees of freedom are
%       stored in the vector, v.

% Set number of Gauss points (only used for forcing term in this example)
NGf = 3;
if (NGf == 3),
  xiGf = [-sqrt(3/5);          0; sqrt(3/5)];
  aGf  = [       5/9;        8/9;       5/9];
elseif (NGf == 2),
  xiGf = [-1/sqrt(3); +1/sqrt(3)];
  aGf  = [       1.0;        1.0];
else,
  NGf = 1;
  xiGf = [0.0];
  aGf  = [2.0];
end

% Number of elements
Ne = 10;
x = linspace(-1,1,Ne+1);

% Zero stiffness matrix
K = zeros(2*Ne+1, 2*Ne+1);
b = zeros(2*Ne+1, 1);

% Loop over all elements and calculate stiffness and residuals
for ii = 1:Ne,

  kn1 = 1 + 2*(ii-1);
  kn2 = 2 + 2*(ii-1);
  kn3 = 3 + 2*(ii-1);

  x1 = x(ii);
  x3 = x(ii+1);

  dx    = x3 - x1;
  dxidx = 2/dx;
  dxdxi = 1/dxidx;
```

```
% Add contribution to kn1 weighted residual
K(kn1, kn1) = K(kn1, kn1) - dxidx*( 7/6);
K(kn1, kn2) = K(kn1, kn2) - dxidx*(-4/3);
K(kn1, kn3) = K(kn1, kn3) - dxidx*( 1/6);

% Add contribution to kn2 weighted residual
K(kn2, kn1) = K(kn2, kn1) - dxidx*(-4/3);
K(kn2, kn2) = K(kn2, kn2) - dxidx*( 8/3);
K(kn2, kn3) = K(kn2, kn3) - dxidx*(-4/3);

% Add contribution to kn3 weighted residual
K(kn3, kn1) = K(kn3, kn1) - dxidx*( 1/6);
K(kn3, kn2) = K(kn3, kn2) - dxidx*(-4/3);
K(kn3, kn3) = K(kn3, kn3) - dxidx*( 7/6);

% Use Gaussian quadrature to evaluate forcing term integral
for nn = 1:NGf,

  % Get xi for Gauss point
  xiG = xiGf(nn);

  % Find N1, N2 and N3 (i.e. weighting/intepolants) at xiG
  N1 = -0.5*xiG*(1-xiG);
  N2 =  (1-xiG)*(1+xiG);
  N3 =  0.5*xiG*(1+xiG);

  % Find x for Gauss point
  xG = 0.5*(1-xiG)*x1 + 0.5*(1+xiG)*x3;

  % Find f for Gauss point
  fG = -50*exp(xG);

  % Evaluate integrand at Gauss point for weight functions at nodes
  gG1 = N1*fG*dxdxi;
  gG2 = N2*fG*dxdxi;
  gG3 = N3*fG*dxdxi;

  % Send to correct right-hand side term
  b(kn1) = b(kn1) + aGf(nn)*gG1;
  b(kn2) = b(kn2) + aGf(nn)*gG2;
  b(kn3) = b(kn3) + aGf(nn)*gG3;

end
```

```
end


% Set Dirichlet conditions at x=0
kn1 = 1;
K(kn1,:)    = zeros(size(1,2*Ne+1));
K(kn1, kn1) = 1.0;
b(kn1)      = 100.0;


% Set Dirichlet conditions at x=1
kn1 = 2*Ne+1;
K(kn1,:)    = zeros(size(1,2*Ne+1));
K(kn1, kn1) = 1.0;
b(kn1)      = 100.0;


% Solve for solution
v = K\b;


% Plot it and compare.  Note: since even the finite element
% solution varies more than linearly across an element, we need to
% subdivide each element, evaluate the basis functions, and plot
% the FEM solution to see the higher order variations.

Nplot = 20; % Number of points per element to plot
nnn = 0;
for ii = 1:Ne,

  kn1 = 1 + 2*(ii-1);
  kn2 = 2 + 2*(ii-1);
  kn3 = 3 + 2*(ii-1);

  x1 = x(ii);
  x3 = x(ii+1);

  v1 = v(kn1);
  v2 = v(kn2);
  v3 = v(kn3);

  for nn = 1:Nplot,

    % Get xi for plot point
```

```
    xiG = -1 + 2*(nn-1)/(Nplot-1);

    % Find N1, N2 and N3 (i.e. weighting/intepolants) at xiG
    N1 = -0.5*xiG*(1-xiG);
    N2 =  (1-xiG)*(1+xiG);
    N3 =  0.5*xiG*(1+xiG);

    % Find x and v for plot point
    xG = 0.5*(1-xiG)*x1 + 0.5*(1+xiG)*x3;
    vG = v1*N1 + v2*N2 + v3*N3;

    nnn = nnn + 1;
    xp(nnn) = xG;
    vp(nnn) = vG;
    up(nnn) = -50*exp(xG) + 50*xG*sinh(1) + 100 + 50*cosh(1);

  end
end

figure(1);
plot(xp,vp,'r');hold on;
plot(xp,up); hold off;
xlabel('x');
ylabel('u');

figure(2);
plot(xp,vp-up);
xlabel('x');
ylabel('Error');
```

In-class Discussion 14.2 (Behavior of Quadratic FEM) *The results in Figures 14.2 and 14.3 will be discussed in class.*

14.3 Hierarchical Basis for Quadratic Elements

In this section, we consider a different basis for quadratic polynomials. Since we want the solution to be continuous from element-to-element, we will still specify that a_1 and a_3 are the values at the end of the elements (i.e. at the nodes),

$$a_1 = \tilde{T}(-1), \qquad a_3 = \tilde{T}(1).$$

However, we will no longer associate a_2 with the midpoint value of the temperature. Instead, let the additional constraint be that a_2 is the value of the second derivative in the middle of the reference element, specifically,

$$a_2 = \tilde{T}_{\xi\xi}(0).$$

These three constraints lead to the following conditions on the $\phi_i(\xi)$:

$$\phi_1(-1) = 1, \quad \phi_{1_{\xi\xi}}(0) = 0, \quad \phi_1(1) = 0.$$

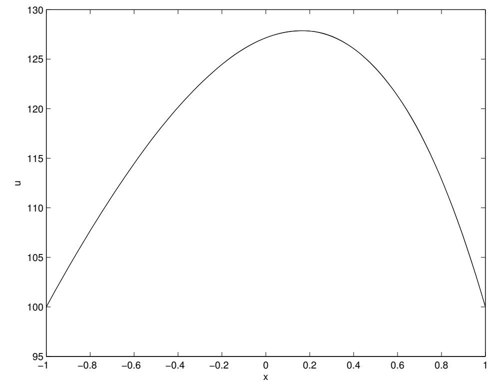
$$\phi_2(-1) = 0, \quad \phi_{2_{\xi\xi}}(0) = 1, \quad \phi_2(1) = 0.$$

$$\phi_3(-1) = 0, \quad \phi_{3_{\xi\xi}}(0) = 0, \quad \phi_3(1) = 1.$$

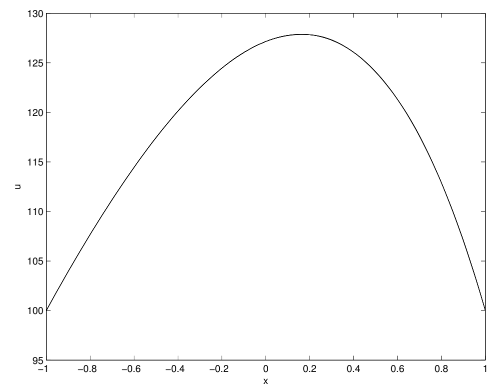
Applying these constraints and solving for the quadratic $\phi_i(\xi)$ gives,

$$\begin{aligned}
\phi_1(\xi) &= \frac{1}{2}(1-\xi), \\
\phi_2(\xi) &= \frac{1}{2}\left(\xi^2 - 1\right), \\
\phi_3(\xi) &= \frac{1}{2}(1+\xi).
\end{aligned}$$

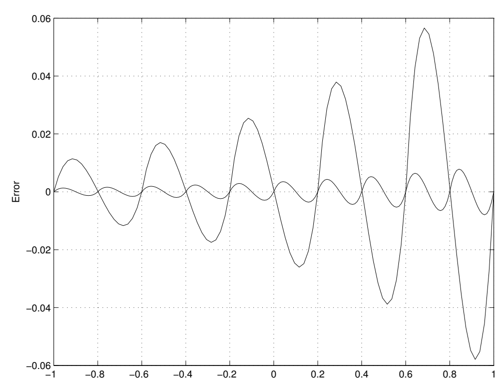
This basis is known as a hierarchical basis because the quadratic basis functions are the usual linear basis functions (ϕ_1 and ϕ_3) with an additional function (ϕ_2) that brings the quadratic contribution into the approximate solution. In other words, the basis is hierarchical because the basis for a linear-varying solution is a subset of the basis for the quadratic solution. The plots of ϕ_1, ϕ_2, and ϕ_3 are shown in Figure 14.4.



(a) $N = 5$ elements

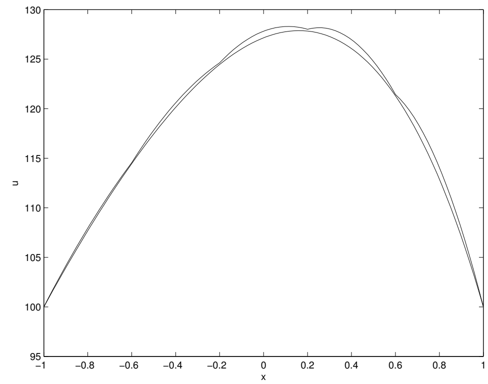


(b) $N = 10$ elements

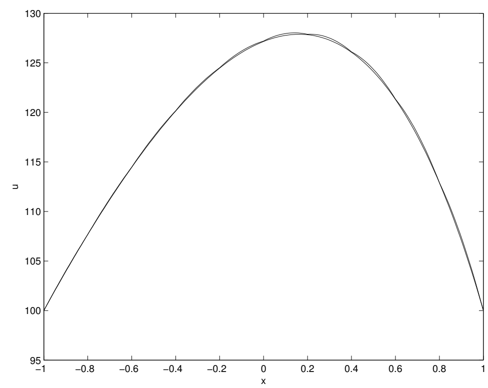


(c) Error

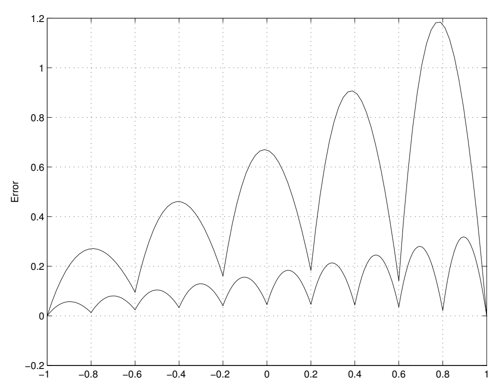
Figure 14.2: Comparison of quadratic finite element solution using 3 point Gaussian quadrature on forcing function.



(a) $N = 5$ elements



(b) $N = 10$ elements



(c) Error

Figure 14.3: Comparison of quadratic finite element solution using 1 point Gaussian quadrature on forcing function.

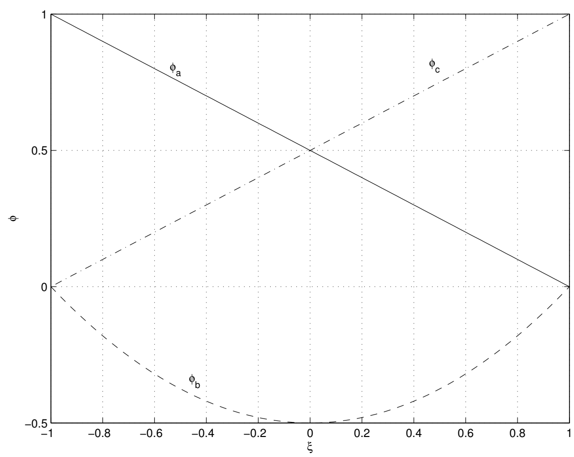
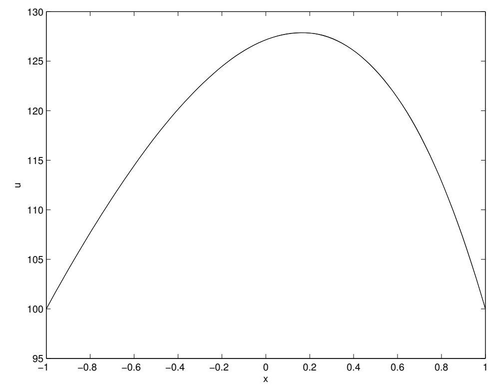
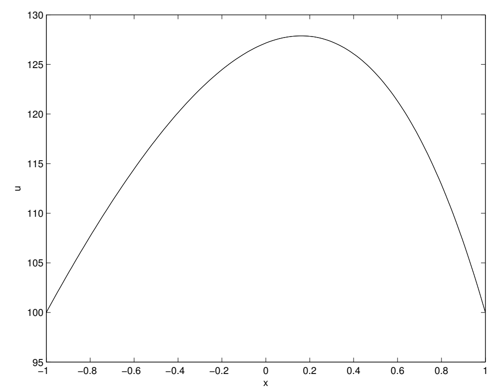


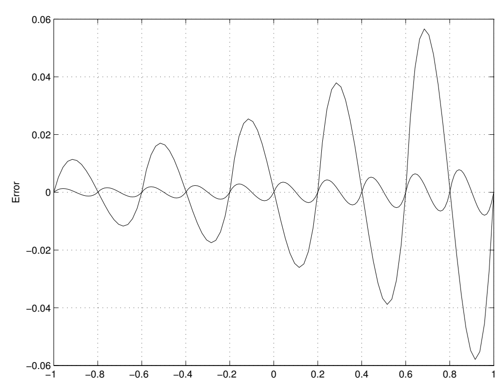
Figure 14.4: Plots of quadratic hierarchical basis functions.



(a) $N = 5$ elements



(b) $N = 10$ elements



(c) Error

Figure 14.5: Comparison of quadratic finite element solution with hierarchical basis using 3 point Gaussian quadrature on forcing function.