Chapter 4

ELEMENT INTERPOLATION AND LOCAL COORDINATES

4.1 Introduction

Up to this point we have relied on the use of a linear interpolation relation that was expressed in *global coordinates* and given by inspection. In the previous chapter we saw numerous uses of these interpolation functions. By introducing more advanced interpolation functions, $\mathbf{H}$, we can obtain more accurate solutions. Here we will show how the common interpolation functions are derived. Then a number of expansions will be given without proof. Also, we will introduce the concept of non-dimensional *local* or element *coordinate* systems. These will help simplify the algebra and make it practical to automate some of the integration procedures.

4.2 Linear Interpolation

Assume that we desire to define a quantity, u, by interpolating in space, from certain given values, $\mathbf{u}$. The simplest interpolation would be linear and the simplest space is the line, e.g. x-axis. Thus to define $u(x)$ in terms of its values, $\mathbf{u}^e$, at selected points on an element we could choose a linear polynomial in x. That is:

$$u^e(x) \;=\; c_1^e + c_2^e x \;=\; \mathbf{P}(x)\,\mathbf{c}^e \tag{4.1}$$

where $\mathbf{P} = [\,1 \quad x\,]$ denotes the linear polynomial behavior in space and $\mathbf{c}^{e^T} = [\,c_1^e \quad c_2^e\,]$ are undetermined constants that relate to the given values, $\mathbf{u}^e$. Referring to Fig. 4.2.1, we note that the element has a physical length of L^e and we have defined the nodal values such that $u^e(x_1) = u_1^e$ and $u^e(x_2) = u_2^e$. To be useful, Eq. (4.1) will be required to be valid at all points on the element, including the nodes. Evaluating Eq. (4.1) at each node of the element gives the set of identities: $u^e(x_1^e) \;=\; u_1^e \;=\; c_1^e + c_2^e\, x_1^e$, or

$$\mathbf{u}^e \;=\; \mathbf{g}^e\,\mathbf{c}^e \tag{4.2}$$

where

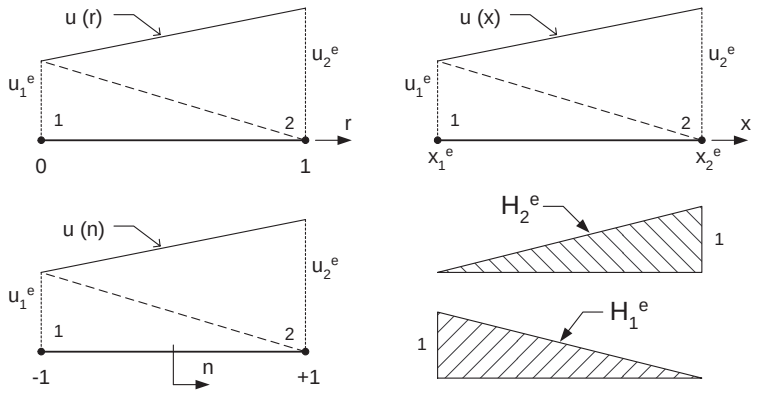

Figure 4.2.1 One-dimensional linear interpolation

$$\mathbf{g}^e = \begin{bmatrix} 1 & x_1^e \\ 1 & x_2^e \end{bmatrix}. \tag{4.3}$$

This shows that the physical constants, $\mathbf{u}^e$, are related to the polynomial constants, $\mathbf{c}^e$ by information on the geometry of the element, $\mathbf{g}^e$. Since $\mathbf{g}^e$ is a square matrix we can (usually) solve Eq. (4.2) to get the polynomial constants:

$$\mathbf{c}^e = \mathbf{g}^{e^{-1}} \mathbf{u}^e. \tag{4.4}$$

In this case the element geometry matrix can be easily inverted to give

$$\mathbf{g}^{e^{-1}} = \frac{1}{x_2^e - x_1^e} \begin{bmatrix} x_2^e & -x_1^e \\ -1 & 1 \end{bmatrix}. \tag{4.5}$$

By putting these results into our original assumption, Eq. (4.1), it is possible to write $u^e(x)$ directly in terms of $\mathbf{u}^e$. That is,

$$u^e(x) = \mathbf{P}(x)\, \mathbf{g}^{e^{-1}} \mathbf{u}^e = \mathbf{H}^e(x)\, \mathbf{u}^e \tag{4.6}$$

or

$$u^e(x) = [\,1 \quad x\,] \frac{1}{L^e} \begin{bmatrix} x_2^e & -x_1^e \\ -1 & 1 \end{bmatrix} \begin{Bmatrix} u_1^e \\ u_2^e \end{Bmatrix} = \begin{bmatrix} \dfrac{x_2^e - x}{L^e} & \dfrac{x - x_1^e}{L^e} \end{bmatrix} \{u^e\} \tag{4.7}$$

where H^e is called the element interplation array. Clearly

$$\mathbf{H}^e(x) = \mathbf{P}(x)\, \mathbf{g}^{e^{-1}}. \tag{4.8}$$

From Eq. (4.6) we can see that the approximate value, $u^e(x)$ depends on the assumed behavior in space, $\mathbf{P}$, the element geometry, $\mathbf{g}^e$, and the element nodal parameters, $\mathbf{u}^e$. This is also true in two- and three-dimensional problems. Since this element interpolation has been defined in a global or physical space the geometry matrix, $\mathbf{g}^e$, and thus $\mathbf{H}^e$ will be different for every element. Of course, the algebraic form is common but the numerical terms differ from element to element. For a given type of element it is possible to make $\mathbf{H}$ unique if a local non-dimensional coordinate is utilized. This will

also help reduce the amount of calculus that must be done by hand. Local coordinates are usually selected to range from 0 to 1, or from −1 to +1. These two options are also illustrated in Fig. 4.2.1. For example, consider the *unit coordinates* shown in Fig. 4.2.1 where the linear polynomial is now $\mathbf{P} = [\,1 \quad r\,]$. Repeating the previous steps yields

$$u^e(r) = \mathbf{P}(r)\,\mathbf{g}^{-1}\,\mathbf{u}^e, \quad \mathbf{g} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \quad \mathbf{g}^{-1} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \tag{4.9}$$

so that

$$u^e(r) = \mathbf{H}(r)\,\mathbf{u}^e \tag{4.10}$$

where the unit coordinate interpolation function is

$$\mathbf{H}(r) = \lfloor\,(1-r) \quad r\,\rfloor = \mathbf{P}\,\mathbf{g}^{-1}. \tag{4.11}$$

Expanding back to scalar form this means

$$u^e(r) = H_1(r)\,u_1^e + H_2(r)\,u_2^e = (1-r)\,u_1^e + ru_2^e = u_1^e + r(u_2^e - u_1^e)$$

so that at $r = 0$, $u^e(0) = u_1^e$ and at $r = 1$, $u^e(1) = u_2^e$ as required.

A possible problem here is that while this simplifies **H** one may not know "where" a given r point is located in global or physical space. In other words, what is x when r is given? One simple way to solve this problem is to note that the nodal values of the global coordinates of the nodes, x^e, are given data. Therefore, we can use the concepts in Eq. (4.10) and define $x^e(r) = \mathbf{H}(r)\,\mathbf{x}^e$, or

$$x^e(r) = (1-r)\,x_1^e + r\,x_2^e = x_1^e + L^e\,r \tag{4.12}$$

for any r in a given element, e. If we make this popular choice for relating the local and global coordinates, we call this an *isoparametric* element. The name implies that a single (iso) set of parametric relations, $\mathbf{H}(r)$, is to be used to define the geometry, $x(r)$, as well as the primary unknowns, $u(r)$.

If we select the symmetric, or Gaussian, local coordinates such that $-1 \le n \le +1$, then a similar set of interpolation functions are obtained. Specifically, $u^e(n) = \mathbf{H}(n)\,\mathbf{u}^e$ with $H_1(n) = (1-n)/2$, $H_2(n) = (1+n)/2$, or simply

$$H_i(n) = (1 + n_i\,n)/2 \tag{4.13}$$

where n_i is the local coordinate of node i. This coordinate system is often called a *natural* coordinate system. Of course, the relation to the global system is

$$x^e(n) = \mathbf{H}(n)\,\mathbf{x}^e. \tag{4.14}$$

The relationship between the unit and natural coordinates is $r = (1+n)/2$. This will sometimes by useful in converting tabulated data in one system to the other. The above local coordinates can be used to define how an approximation changes in space. They also allow one to calculate derivatives. For example, from Eq. (4.10)

$$du^e/dr = d\,\mathbf{H}(r)/dr\;\mathbf{u}^e \tag{4.15}$$

and similarly for other quantities of interest. Another quantity that we will find very important is dx/dr. In a typical linear element, Eq. (4.12) gives

$$dx^e(r)/dr = dH_1/dr\;x_1^e + dH_2/dr\;x_2^e = -x_1^e + x_2^e$$

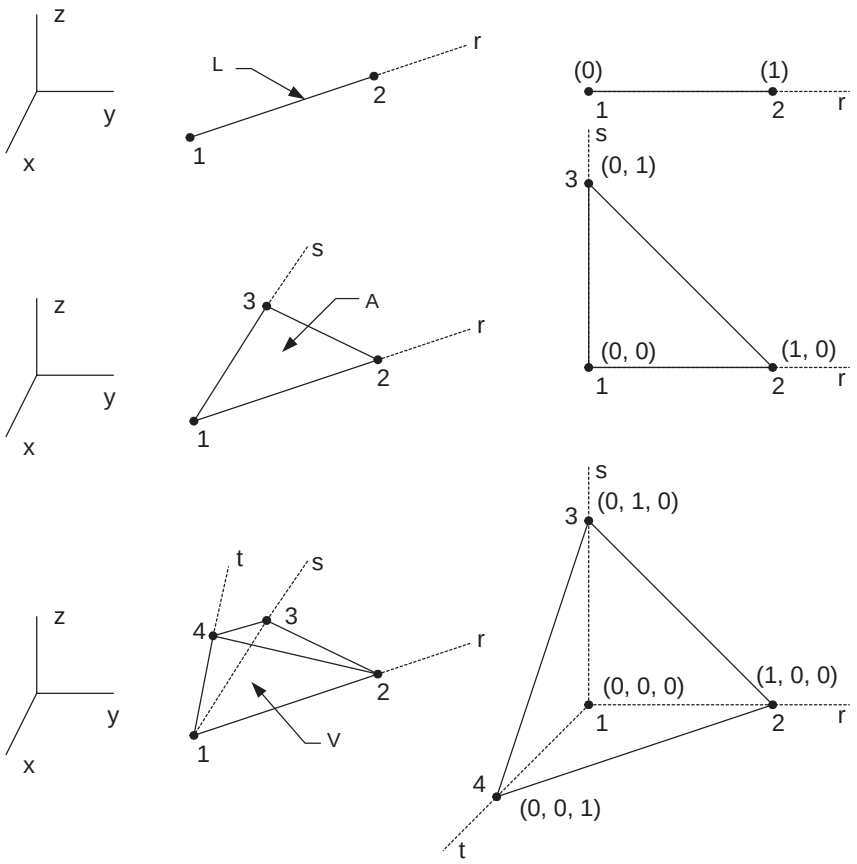

Figure 4.2.2 The simplex element family in unit coordinates

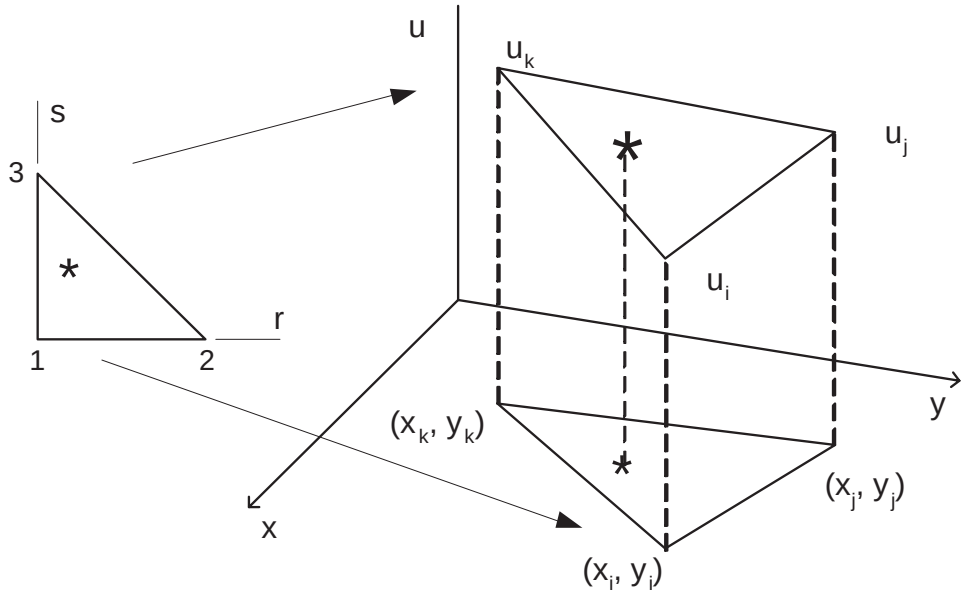

Figure 4.2.3 Isoparametric interpolation on a simplex triangle

or simply $dx^e/dr = L^e$. By way of comparison, if the natural coordinate is utilized

$$dx^e(n)/dn = L^e/2. \tag{4.16}$$

This illustrates that the choice of the local coordinates has more effect on the derivatives than it does on the interpolation itself. The use of unit coordinates is more popular with *simplex elements*. These are elements where the number of nodes is one higher than the dimension of the space. The generalization of unit coordinates for common simplex elements is illustrated in Fig. 4.2.2. For simplex elements the natural coordinates becomes *area coordinates* and *volume coordinates*, which the author finds rather unnatural. Fig. 4.2.3 shows how the same parametric interpolations can be used for more than one purpose in an analysis. There we see that the spatial positions of points on the element are interpolated from a linear parametric triangle, and the function value is interpolated in the same way. Both unit and natural coordinates are effective for use on squares or cubes in the local space. In global space those shapes become quadrilaterals or hexahedra. The natural coordinates are more popular for those shapes.

4.3 Quadratic Interpolation

The next logical spatial form to pick is that of a quadratic polynomial. Select three nodes on the line element, two at the ends and the third inside the element. In local space the third node is at the element center. Thus, the local unit coordinates are $r_1 = 0$, $r_3 = \frac{1}{2}$, and $r_2 = 1$. It is usually desirable to have x_3 also at the center of the element in global space. If we repeat the previous procedure using $u(r) = c_1 + c_2 r + c_3 r^2$, then the element interpolation functions are found to be

$$\begin{aligned} H_1(r) &= 1 - 3r + 2r^2 \\ H_2(r) &= -r + 2r^2 \\ H_3(r) &= 4r - 4r^2 \\ \hline \Sigma H_i(r) &= 1 \end{aligned}. \tag{4.17}$$

These quadratic functions are completely different from the linear functions. Note that these functions have a sum that is unity at any point, r, in the element. These three functions illustrate another common feature of C^0 interpolation functions. They are unity at one node and zero at all others: $H_i(r_j) = \delta_{ij}$. These functions expressed in natural coordinates are

$$H_1(n) = n(n-1)/2, \quad H_2(n) = n(n+1)/2, \quad H_3(n) = 1 - n^2. \tag{4.18}$$

4.4 Lagrange Interpolation

Clearly this one dimensional procedure can be readily extended by adding more nodes to the interior of the element. Usually the additional nodes are equally spaced along the element. However, they can be placed in arbitrary locations. The interpolation function for such an element is known as the Lagrange interpolation polynomial. The one-dimensional m-th order *Lagrange interpolation* polynomial is the ratio of two

$-1 < n < 1$	$0 < r < 1$
a) Linear $1 - - - - - - - 2$	
$H_1 = (1-n)/2$	$H_1 = (1-r)$
$H_2 = (1+n)/2$	$H_2 = r$
b) Quadratic $1 - - - 3 - - - 2$	
$H_1 = n(n-1)/2$	$H_1 = (r-1)(2r-1)$
$H_2 = n(n+1)/2$	$H_2 = r(2r-1)$
$H_3 = (1+n)(1-n)$	$H_3 = 4r(1-r)$
c) Cubic $1 - - 3 - - 4 - - 2$	
$H_1 = (1-n)(3n+1)(3n-1)/16$	$H_1 = (1-r)(2-3r)(1-3r)/2$
$H_2 = (1+n)(3n+1)(3n-1)/16$	$H_2 = r(2-3r)(1-3r)/2$
$H_3 = 9(1+n)(n-1)(3n-1)/16$	$H_3 = 9r(1-r)(2-3r)/2$
$H_4 = 9(1+n)(1-n)(3n+1)/16$	$H_4 = 9r(1-r)(3r-1)/2$

Figure 4.4.1 Typical Lagrange interpolations

products. For an element with $(m+1)$ nodes, r_i, $i = 1, 2, \ldots, (m+1)$, the interpolation function for the k-th node is

$$H_k^m(n) = \prod_{\substack{i=1 \\ i \neq k}}^{m+1} (n - n_i) \Big/ \prod_{\substack{i=1 \\ i \neq k}}^{m+1} (n_k - n_i). \tag{4.19}$$

This is a complete m-th order polynomial in one dimension. It has the property that $H_k(n_i) = \delta_{ik}$. For local coordinates given on the domain [−1, 1], a typical term for three equally spaced nodes is

$$H_3(n) = \frac{(n-(-1))\,(n-1)}{(0-(-1))\,(0-1)} = (1-n^2).$$

Similarly, $H_1(n) = n(n-1)/2$ and $H_2 = n(n+1)/2$ and their sum is unity. Figure 4.4.1 shows typical node locations and interpolation functions for members of this family of complete polynomial functions on simplex elements. Figure 4.4.2 shows the typical coding of a quadratic line element (subroutines SHAPE_3_L and DERIV_3_L).

4.5 Hermitian Interpolation


```
SUBROUTINE SHAPE_3_L (X, H)                                          ! 1
! *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-*               ! 2
! CALCULATE SHAPE FUNCTIONS OF A 3 NODE LINE ELEMENT                 ! 3
!             IN NATURAL COORDINATES                                 ! 4
! *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-*               ! 5
Use Precision_Module                                                 ! 6
 IMPLICIT NONE                                                       ! 7
 REAL(DP), INTENT(IN)  :: X                                          ! 8
 REAL(DP), INTENT(OUT) :: H (3)                                      ! 9
                                                                     !10
! H = ELEMENT SHAPE FUNCTIONS                                       !11
! X = LOCAL COORDINATE OF POINT,    -1 TO +1                        !12
! LOCAL NODE COORD. ARE -1,0,+1    1-----2-----3                    !13
                                                                     !14
  H (1) = 0.5d0*(X*X - X)                                            !15
  H (2) = 1.d0 - X*X                                                 !16
  H (3) = 0.5d0*(X*X + X)                                            !17
END SUBROUTINE SHAPE_3_L                                             !18
                                                                     !19
SUBROUTINE DERIV_3_L (X, DH)                                         !20
! *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-*               !21
! FIND LOCAL DERIVATIVES FOR A 3 NODE LINE ELEMENT                   !22
!             IN NATURAL COORDINATES                                 !23
! *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-*               !24
Use Precision_Module                                                 !25
 IMPLICIT NONE                                                       !26
 REAL(DP), INTENT(IN)  :: X                                          !27
 REAL(DP), INTENT(OUT) :: DH (3)                                     !28
                                                                     !29
! DH = LOCAL DERIVATIVES OF SHAPE FUNCTIONS (SHAPE_3_L)             !30
! X  = LOCAL COORDINATE OF POINT,    -1 TO +1                       !31
! LOCAL NODE COORD. ARE -1,0,+1    1----2----3                      !32
                                                                     !33
  DH (1) = X - 0.5d0                                                 !34
  DH (2) = - 2.d0 * X                                                !35
  DH (3) = X + 0.5d0                                                 !36
END SUBROUTINE DERIV_3_L                                             !37
```

Figure 4.4.2 Coding a Lagrangian quadratic line element

All of the interpolation functions considered so far have C^0 continuity between elements. That is, the function being approximated is continuous between elements but its derivative is discontinuous. However, know that some applications, such as a beam analysis, also require that their derivative be continuous. These C^1 functions are most easily generated by using derivatives, or slopes, as nodal degrees of freedom.

The simplest element in this family is the two node line element where both y and dy/dx are taken as nodal degrees of freedom. Note that a global derivative has been selected as a degree of freedom. Since there are two nodes with two dof each, the interpolation function has four constants, thus, it is a cubic polynomial. The form of this *Hermite polynomial* is well known. The element is shown in Fig. 4.5.1 along with the interpolation functions and their global derivatives. The latter quantities are obtained from the relation between local and global coordinates, e.g., Eq. (4.12). On rare occasions one may also need to have the second derivatives continuous between

$$x = Lr \qquad (\,)' = d(\,)/dx$$

a) C^1 : $U = H_1 U_1 + H_2 U_1' + H_3 U_2 + H_4 U_2'$

$$H_1(r) = (2r^3 - 3r^2 + 1)$$
$$H_2(r) = (r^3 - 2r^2 + r)\, L$$
$$H_3(r) = (3r^2 - 2r^3)$$
$$H_4(r) = (r^3 - r^2)\, L$$

b) C^2 : $U = H_1 U_1 + H_2 U_1' + H_3 U_1'' + H_4 U_2 + H_5 U_2' + H_6 U_2''$

$$H_1 = (1 - 10r^3 + 15r^4 - 6r^5)$$
$$H_2 = (r - 6r^3 + 8r^4 - 3r^5)\, L$$
$$H_3 = (r^2 - 3r^3 + 3r^4 - r^5)\, L^2/2$$
$$H_4 = (10r^3 - 15r^4 + 6r^5)$$
$$H_5 = (7r^4 - 3r^5 - 4r^3)\, L$$
$$H_6 = (r^3 - 2r^4 + r^5)\, L^2/2$$

c) C^3 : $U = H_1 U_1 + H_2 U_1' + H_3 U_1'' + H_4 U_1''' + H_5 U_2 + H_6 U_2' + H_7 U_2'' + H_8 U_2'''$

$$H_1 = (1 - 35r^4 + 84r^5 - 70r^6 + 20r^7)$$
$$H_2 = (r - 20r^4 + 45r^5 - 36r^6 + 10r^7)/L$$
$$H_3 = (r^2 - 10r^4 + 20r^5 - 15r^6 + 4r^7)\, L^2/2$$
$$H_4 = (r^3 - 4r^4 + 6r^5 - 4r^6 + r^7)\, L^3/6$$
$$H_5 = (35r^4 - 84r^5 + 70r^6 - 20r^7)$$
$$H_6 = (10r^7 - 34r^6 + 39r^5 - 15r^4)\, L$$
$$H_7 = (5r^4 - 14r^5 + 13r^6 - 4r^7)\, L^2/2$$
$$H_8 = (r^7 - 3r^6 + 3r^5 - r^4)\, L^3/6$$

Figure 4.5.1 Hermitian interpolation in unit coordinates

elements. Typical C^2 equations of this type are also given in Fig. 4.5.1 and elsewhere. Since derivatives have also been introduced as nodal parameters, the previous statement that $\Sigma H_i = 1$ is no longer true.

4.6 Hierarchical Interpolation

Recently some alternate types of interpolation have become popular. They are called *hierarchical functions*. The unique feature of these polynomials is that the higher order polynomials contain the lower order ones. This concept is shown in Fig. 4.6.1. Thus, to get new functions you simply add some terms to the old functions. To illustrate this concept let us return to the linear element in local natural coordinates. In that


```
SUBROUTINE SHAPE_C1_L (R, L, H)                                          ! 1
! *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-*       ! 2
! SHAPE FUNCTIONS FOR CUBIC HERMITE IN UNIT COORDINATES                  ! 3
! *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-*       ! 4
Use Precision_Module                                                     ! 5
 IMPLICIT NONE                                                           ! 6
 REAL(DP), INTENT(IN)  :: R, L                                           ! 7
 REAL(DP), INTENT(OUT) :: H (4)                                          ! 8
                                                                         ! 9
! L = PHYSICAL LENGTH OF ELEMENT     1----------2 ---> R                 !10
! R = LOCAL COORDINATE OF POINT     R=0         R=1                       !11
! H = SHAPE FUNCTIONS ARRAY                                             !12
! DOF ARE FUNCTION AND SLOPE, WRT X, AT EACH NODE (N_G_DOF=2)           !13
! D()/DX = D()/DR DR/DX = 1/L * D()/DR                                  !14
                                                                        !15
  H(1) = 1.d0 - 3.0*R*R + 2.0*R*R*R                                     !16
  H(2) = (R - 2.0*R*R + R*R*R)*L                                        !17
  H(3) =  3.0*R*R - 2.0*R*R*R                                           !18
  H(4) = (R*R*R - R*R)*L                                                !19
END SUBROUTINE SHAPE_C1_L                                               !20
                                                                        !21
SUBROUTINE DERIV_C1_L (R, L, DH)                                        !22
! *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-*       !23
! FIRST DERIVATIVES OF CUBIC HERMITE IN UNIT COORDINATES                !24
! *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-*       !25
Use Precision_Module                                                    !26
 IMPLICIT NONE                                                          !27
 REAL(DP), INTENT(IN)  :: R, L                                          !28
 REAL(DP), INTENT(OUT) :: DH (4)                                        !29
                                                                        !30
! L = PHYSICAL LENGTH OF ELEMENT       1 -------- 2 --> R               !31
! R = LOCAL COORDINATE OF POINT        R=0        R=1                   !32
! DH = FIRST PHYSICAL DERIVATIVES OF H                                  !33
                                                                        !34
  DH (1) = 6.d0 * (R * R - R) / L                                       !35
  DH (2) = 1.d0 - 4.d0 * R + 3.d0 * R * R                               !36
  DH (3) = 6.d0 * (R - R * R) / L                                       !37
  DH (4) = 3.d0 * R * R - 2.d0 * R                                      !38
END SUBROUTINE DERIV_C1_L                                               !39
                                                                        !40
SUBROUTINE DERIV2_C1_L (R, L, D2H)                                      !41
! *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-*       !42
! 2ND DERIVATIVES OF CUBIC HERMITE IN UNIT COORDINATES                  !43
! *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-* *-*       !44
Use Precision_Module                                                    !45
 IMPLICIT NONE                                                          !46
 REAL(DP), INTENT(IN)  :: R, L                                          !47
 REAL(DP), INTENT(OUT) :: D2H (4)                                       !48
                                                                        !49
! L = PHYSICAL LENGTH OF ELEMENT       1 -------- 2 --> R               !50
! R = LOCAL COORDINATE OF POINT        R=0        R=1                   !51
! D2H = SECOND DERIVATIVES OF H                                         !52
                                                                        !53
  D2H (1) = 6.d0 * (R + R - 1.d0) / L**2                                !54
  D2H (2) = ( - 4.d0 + 6.d0 * R) / L                                    !55
  D2H (3) = 6.d0 * (1.d0 - R - R) / L**2                                !56
  D2H (4) = (6.d0 * R - 2.d0) / L                                       !57
END SUBROUTINE DERIV2_C1_L                                              !58
```

Figure 4.5.2 The C^1 Hermite cubic

element

$$u^e(n) \;=\; H_1(n)\, u_1^e + H_2(n)\, u_2^e \tag{4.20}$$

where the two H_i are given in Eq. (4.10). We want to generate a quadratic interpolation form that will not destroy these H_i as Eq. (4.17) did. The key to accomplishing this goal is to note that the second derivative of (4.10) is everywhere zero. Thus, if we introduce an additional degree of freedom related to the second derivative of u it will not affect the linear terms. Figure 4.6.1 shows the linear element where we have added a third midpoint ($n = 0$) control node to be associated with the quadratic additions. At the third node let the degree of freedom be the second local derivative, d^2u/dr^2. Upgrade the approximation by setting

$$u(n) \;=\; H_1(n)\, u_1^e + H_2(n)\, u_2^e + Q_3(n)\, d^2\, u^e/dn^2 \tag{4.21}$$

where the hierarchical quadratic addition is: $H_3(n) = c_1 + c_2\, n + c_3\, n^2$. The three constants are found from the conditions that it vanishes at the two original nodes, so as not to change H_1 and H_2, and the second derivative is unity at the new midpoint node. The result is

$$H_3(n) \;=\; (n^2 - 1)/2\,. \tag{4.22}$$

The concept is extended to a cubic hierarchical element by using the new function in conjunction with the third tangential derivative at the center.

The higher order hierarchical functions are becoming increasingly popular. They utilize the higher derivatives at the center node. We introduce the notation $m \to n$ to denote the presence of consecutive tangential derivatives from order m to order n. The value of the function is implied by $m = 0$. These functions must vanish at the end nodes. Finally, we usually want the function $H_{p+1}(n)$, $p \ge 2$ to have its p-th derivative take on a value of unity at the center node. The resulting functions are not unique. A common set of hierarchical functions in natural coordinates $-1 \le n \le 1$ are

$$H_p(n) \;=\; (n^p - b)/\,p!\quad , \qquad p \ge 2 \tag{4.23}$$

where $b = 1$ if p is even, and $b = n$ if p is odd. The first six members of this family are shown in Fig. 4.6.2. Note that the even functions approach a rectangular shape as $p \to \infty$, but there is not much change in their form beyond the fourth order polynomial. Likewise, the odd functions approach a sawtooth as $p \to \infty$, but they change relatively little after the cubic order polynomial. These observations suggest that for the above hierarchical choice it may be better to stop at the fourth order polynomial and refine the mesh rather than adding more hierarchical degrees of freedom. However, this form might capture shape boundary layers or shocks better than other choices. These relations are zero at the ends, $n = \pm 1$. The first derivatives of these functions are

$$H'_{p+1} \;=\; [\, pn^{(p-1)} - b']/\,p!$$

and since b'' is always zero, the second derivatives are

$$H''_{p+1} \;=\; p\,(p-1)\, n^{(p-2)}/\,p! \;=\; n^{(p-2)}/(p-2)\,!\,.$$

Proceeding in this manner it is easy to show by induction that the m-th derivative for $m \ge 2$ is

$$H^{(m)}_{p+1}(n) \;=\; n^{(p-m)}/(p-m)\,!\,. \tag{4.24}$$

At the center point, $n = 0$, the derivative has a value of

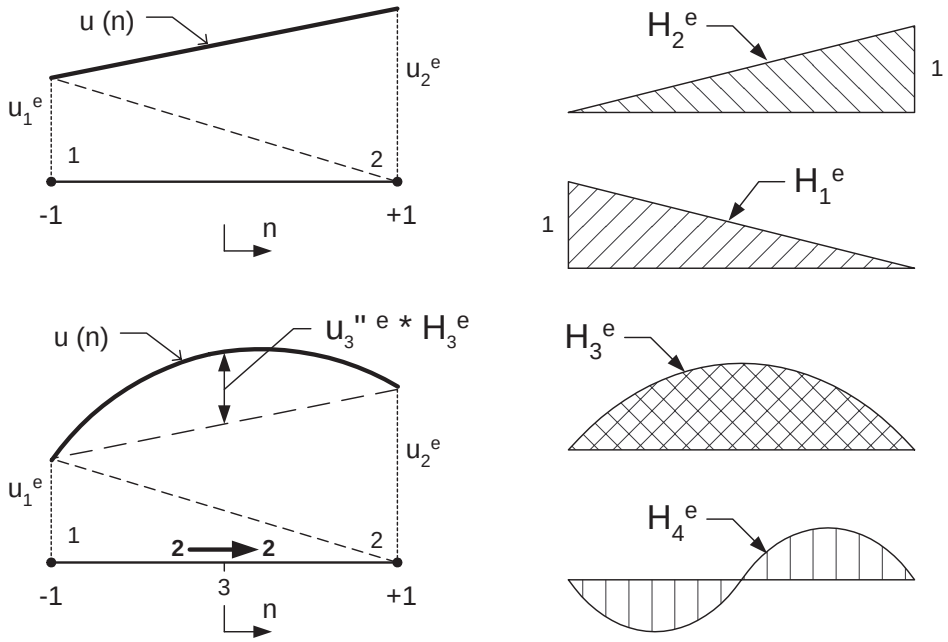

Figure 4.6.1 Concept of hierarchical shape functions

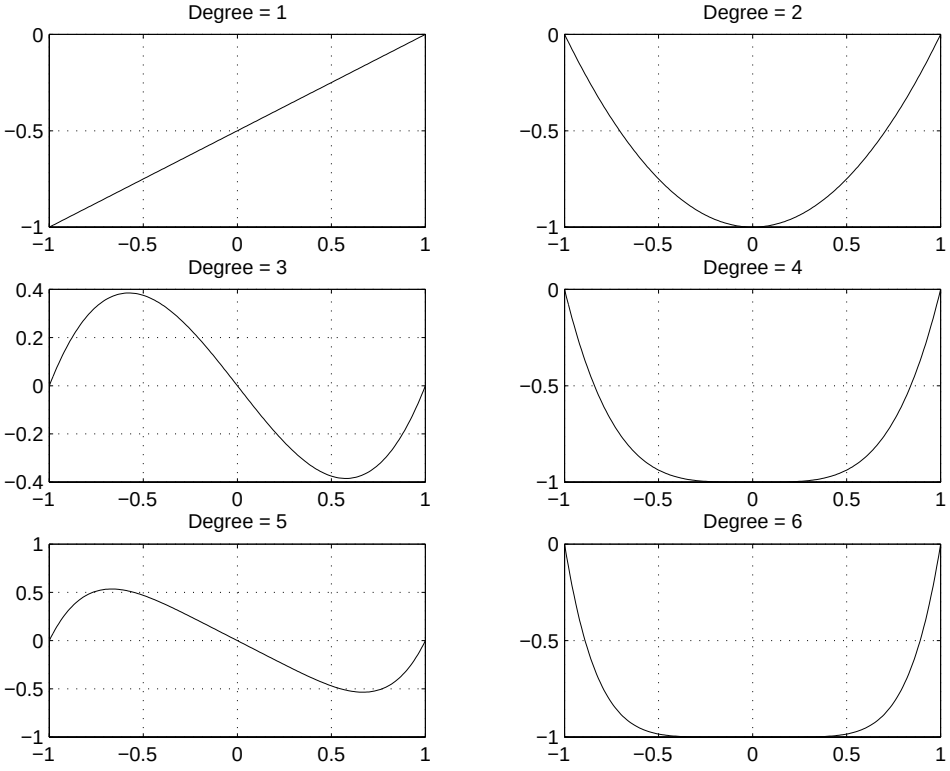

Figure 4.6.2 A C^0 hierarchical family

$$H_{p+1}^{(m)}(0) = \begin{cases} 0 & \text{if} \quad m \neq p \\ 1 & \text{if} \quad m = p \,. \end{cases}$$

We will see later that when hierarchical functions are utilized, the element matrices for a p-th order polynomial are partitions of the element matrices for a $(p+1)$ order polynomial. A typical cubic element, shown in Fig. 4.6.3, would be built by using the first two hierarchical functions shown in the previous figure.

The element square matrix will always involve an integral of the product of the derivatives of the interpolation functions. If those derivatives were orthogonal then they would result in a diagonal square matrix. That would be very desirable. Thus, it is becoming popular to search for interpolation functions whose derivatives are close to being orthogonal. It is well known that integrals of products of *Legendre polynomials* are orthogonal. This suggests that a useful trick would be to pick interpolation functions that are integrals of Legendre polynomials so that their derivatives are Legendre polynomials. Such a trick is very useful in the so-called *p-method* and *hp-method* of adaptive finite element analysis. For future reference we will observe that the first six Legendre polynomials on the domain of $-1 \le x \le 1$ are [1, 6]:

$$\begin{aligned} P_0(x) &= 1 \\ P_1(x) &= x \\ P_2(x) &= (3x^2 - 1)/2 \\ P_3(x) &= (5x^3 - 3x)/2 \\ P_4(x) &= (35x^4 - 30x^2 + 3)/8 \\ P_5(x) &= (63x^5 - 70x^3 + 15x)/8 \\ P_6(x) &= (231x^6 - 315x^4 + 105x^2 - 5)/16 \end{aligned} \tag{4.25}$$

Legendre polynomials can be generated from the *recursion formula*:

$$(n+1)\, P_{n+1}(x) = (2n+1)\, x\, P_n(x) - nP_{n-1}(x), \qquad n \ge 1$$

and

$$n\, P'_{n+1}(x) = (2n+1)\, x\, P'_n(x) - (n+1)\, P'_{n-1}(x). \tag{4.26}$$

To avoid roundoff error and unnecessary calculations, these recursion relations should be used instead of Eq. (4.25) when computing these polynomials. They have the orthogonality property:

$$\int_{-1}^{+1} P_i(x)\; P_j(x)\; dx = \begin{cases} \dfrac{2}{2i+1} & \text{for} \quad i = j \\ 0 & \text{for} \quad i \neq j \,. \end{cases} \tag{4.27}$$

To create a family of functions for potential use as hierarchical interpolation functions we next consider the integral of the above polynomials. Define a new function

$$\gamma_j(x) = \int_{-1}^{x} P_{j-1}(t)\, dt\,. \tag{4.28}$$

A handbook of mathematical functions shows the useful relation for Legendre polynomials that

$$(2j-1)\,P_{j-1}(t) = P'_j(t) - P'_{j-2}(t) \tag{4.29}$$

where $(\ \ )'$ denotes dP/dt. The integral of the derivative is evaluated by inspection so

$$\gamma_j(x) = [\,P_j(x) - P_{j-2}(x)\,]/(2j-1) \tag{4.30}$$

since the lower limit terms cancel each other because

$$P_j(-1) = \begin{cases} 1 & j \text{ even} \\ -1 & j \text{ odd}\,. \end{cases}$$

We may want to multiply by a constant to scale such a function in a special way. For example, to make its second derivative unity at $x = 0$. Thus, for use as interpolation functions we will consider the family of functions defined as

$$\phi_j(x) = [\,P_j(x) - P_{j-2}(x)\,]/\lambda_j \equiv \psi_j(x)/\lambda_j \tag{4.31}$$

where λ_j is a constant to be selected later. From the definition of the Legendre polynomials, we see that the first few values of $\psi_j(x)$ that are of interest are:

$$\begin{aligned} \psi_2(x) &= 3(x^2-1)/2 \\ \psi_3(x) &= 5(x^3-x)/2 \\ \psi_4(x) &= 7(5x^4-6x^2+1)/8 \\ \psi_5(x) &= 9(7x^5-10x^3+3x)/8 \\ \psi_6(x) &= 11(21x^6-35x^4+15x^2-1)/16 \end{aligned} \tag{4.32}$$

These functions are shown in Fig. 4.6.3 along with a linear polynomial. Note that each function has its number of roots (zero values) equal to the order of the polynomial. The previous set had only two roots for the even order polynomials and three roots for the odd order polynomials (excluding the linear one). Thus, this is clearly a different type of function for hierarchical use. These would be more expensive to integrate numerically since there are more terms in each function. Note that the $\psi_j(x)$ have the property that they vanish at the ends of the domain: $\psi_j(\pm 1) \equiv 0,\ j \geq 2$. A popular choice for the midpoint hierarchical interpolation functions is to pick

$$H_j(x) = \phi_{j-1}(x), \qquad j \geq 3 \tag{4.33}$$

where the scaling is chosen to be

$$\lambda_j \equiv \sqrt{4j-2}\,. \tag{4.34}$$

The reader should note for future reference that if the above domain of $-1 \leq x \leq 1$ was the edge of a two-dimensional element then the above derivatives would be viewed as tangential derivatives on that edge. The same is true for edges of solid elements. Hierarchical enrichment is not just restricted to C^0 functions, but have also been used with Hermite functions as well. Earlier we saw the C^1 cubic Hermite and the C^2 fifth order Hermite polynomials. The cubic has nodal dof that are the value and slope of the

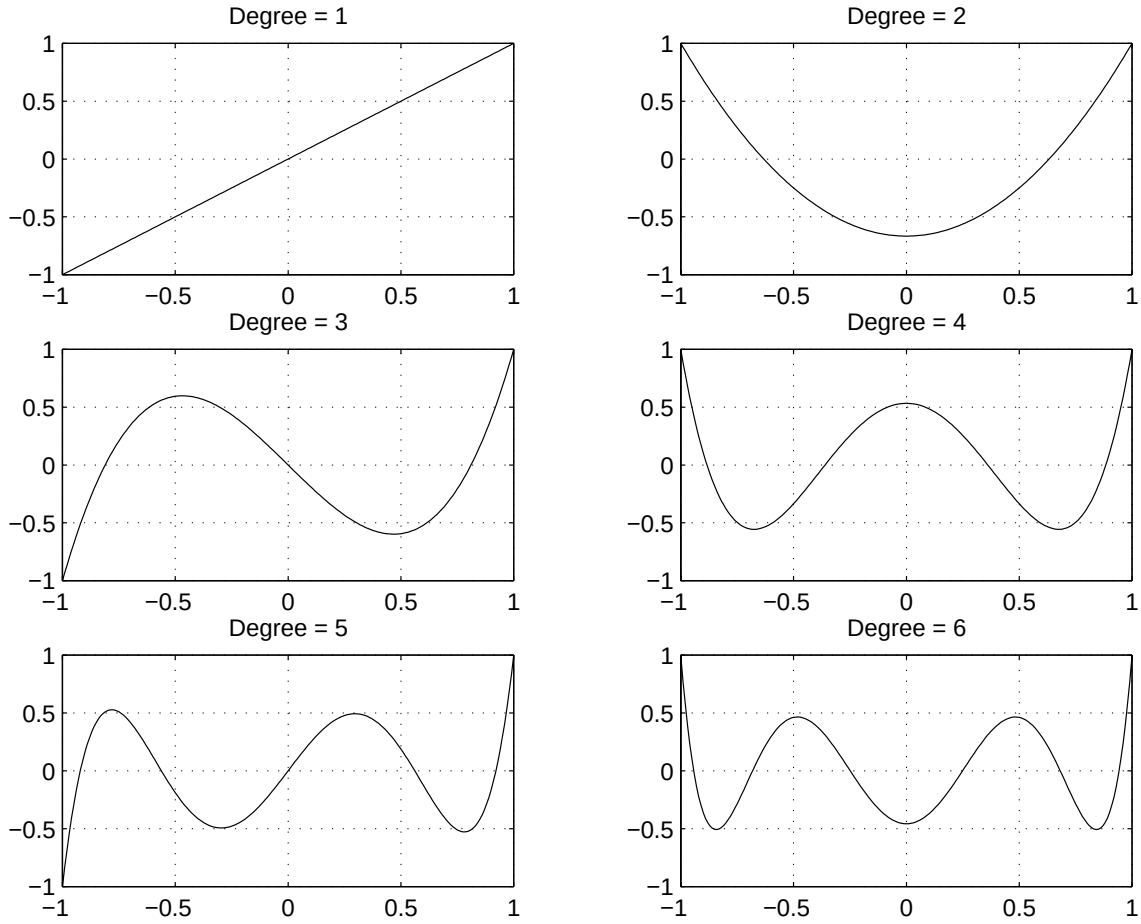

Figure 4.6.3 Integrals of Legendre polynomials 5.0in

solution at each end. If we desire to add a center hierarchical enrichment, then that function should have a zero value and slope at each end. In addition, since the fourth derivative of the cubic polynomial is zero, we select that quantity as the first hierarchical dof. In natural coordinates $-1 \le a \le 1$, we have $p-3$ internal functions for $p \ge 3$. One possible choice is

$$H_p^{(0)} = \begin{cases} 1/p!\left[\, a^{p/2} - 1 \,\right]^2, & p \ge 4, \ \text{even} \\ \dfrac{1}{p!}\left[\, a^{(p-1)/2} - 1 \,\right]^2, & p \ge 5, \ \text{odd}. \end{cases} \tag{4.35}$$

For example, for $p = 4$, $H_p^{(0)} = [\, a^4 - 2a^2 + 1 \,]24$, which is zero at both ends as is its first derivative $\dfrac{d\, H_p^{(0)}}{d\, a} = [\, 4a^3 - 4a \,]$. while its fourth local derivative is unity for all a. We associate that constant dof with the center point, $a = 0$. A similar set of enhancements that have zero second derivatives at the ends can be used for the C^2 elements.

4.7 Nodally Exact Interpolations *

Recall that the analytic solution to a differential equation is generally viewed as the sum of a homogeneous solution and a particular solution. It has been proved by Tong [7] that if the finite element interpolation functions are the exact solution to the homogeneous differential equation ($q = 0$), then the finite element solution of a non-homogeneous (non-zero) source term will *always* be exact at the nodes. Clearly, this also means that if the source is zero, then this type of solution would be exact everywhere. It is well known that the exact solution of the homogeneous equations for the bar on an elastic foundation (or a rod conducting and convecting heat) will generally involve hyperbolic functions. Therefore, if we replaced the polynomial interpolations with the homogeneous hyperbolic functions we can assure ourselves of results that are at least exact at the nodes. For the problems considered here, it can be shown that the typical element matrices obtained from interpolating with the exact homogeneous solutions are summarized in Tables 4.1 and 4.2. In practice, using hyperbolic functions with large arguments can break down due to the way their values are computed in the operating system mathematics library.

Table 4.1 Homogeneous solution interpolation for semi-infinite axial bar on a foundation

a) PDE : $\dfrac{d}{dx}\left(EA\,\dfrac{d\,u}{d\,x} \right) - k\,u \;=\; q, \qquad m \;=\; k \,/\, EA, \qquad k > 0$

b) Homogeneous Interpolation : $H_1 \;=\; e^{-m\,x}$

c) Stiffness Matrix : $K_{11} \;=\; \dfrac{m\,EA}{2} \;+\; \dfrac{k}{2\,m}$

d) Force Vector : $F_1 \;=\; q \,/\, m, \qquad q \;=\;$ constant

e) Mass Matrix : $M_{11} \;=\; \rho \,/\, 2\,m$

Table 4.2 Homogeneous solution interpolation for finite axial bar on a foundation

a) PDE : $\dfrac{d}{dx}\left(EA\,\dfrac{d\,u}{d\,x}\right) - k\,u \;=\; q, \qquad m \;=\; k \,/\, EA, \qquad k > 0$

b) Homogeneous Interpolation : $S \;=\; \sinh(\,mL^e\,), \qquad C \;=\; \cosh(\,mL^e\,)$

$$\mathbf{H} \;=\; \frac{1}{S}\,[\;\sinh[\,m(L-x)\,] \qquad \sinh(\,mx\,)\;]$$

c) Stiffness Matrix : $a \;=\; k + EA\,m^2, \qquad b \;=\; k - EA\,m^2$

$$\mathbf{K} \;=\; \frac{1}{2m\,S^2}\begin{bmatrix} (\,a\,\sinh(\,2\,mL^e\,) - b\,mL^e\,) & (\,b\,\sinh(\,2\,mL^e\,) - a\,S\,) \\ \text{symmetric} & (\,a\,\sinh(\,2\,mL^e\,) - b\,mL^e\,) \end{bmatrix}$$

d) Force Vector : $q \;=\; q_1\,(\,1 - x/L^e\,) + q_2\;x/L^e$

$$\mathbf{F} \;=\; \frac{1}{m}\left\{ \begin{array}{c} \dfrac{q_1\,(\,C-1\,)}{S} \;+\; \dfrac{(\,q_2 - q_1\,)\;(\,1 - mL^e\,/\,S\,)}{m\,L^e} \\ \dfrac{q_1\,(\,C-1\,)}{S} \;+\; \dfrac{(\,q_2 - q_1\,)\;(\,mL^e\,\coth(\,mL^e\,) - 1\,)}{m\,L^e} \end{array} \right\}$$

e) Mass Matrix : $\mathbf{M} \;=\; \dfrac{\zeta}{2m\,S^2}\begin{bmatrix} (\,\sinh(\,2\,mL^e\,) - mL^e\,) & (\,S + \sinh(\,2\,mL^e\,)\,) \\ \text{symmetric} & (\,\sinh(\,2\,mL^e\,) - mL^e\,) \end{bmatrix}$

4.8 Interpolation Error *

To obtain a physical feel for the typical errors involved, we consider a one-dimensional model. A hueristic argument will be used. Recall the Taylor's series of a function v at a point x :

$$v(x+h) \;=\; v(x) + h\,\frac{\partial v}{\partial x}\,(x) + \frac{h^2}{2}\;\frac{\partial^2 v}{\partial x^2}\,(x) \;+\; \cdots \tag{4.36}$$

The objective here is to show that if the third term is neglected, then the relations for the linear line element are obtained. That is, the third term is a measure of the interpolation error in the linear element. For an element with nodes at i and j, we use Eq. (4.36) to estimate the function at node j when h is the length of the element :

$$v_j \;=\; v_i + h\,\frac{\partial v}{\partial x}\,(\,x_i\,)\,.$$

Solving for the gradient at node i yields

$$\frac{\partial v}{\partial x}(x_i) = \frac{(v_j - v_i)}{h} = \frac{\partial v}{\partial x}(x_j)$$

which is the constant previously obtained for the derivative in the linear line element. Thus, we can think of this type of element as representing the first two terms of the Taylor series. The omitted third term is a measure of the error associated with the element. Its value is proportional to the product of the second derivative and the square of the element size.

If the exact solution is linear so that the first derivative is constant, then the second derivative, $\partial^2 v/\partial x^2$, is zero and there is no error in the element. Otherwise, the second derivative and element error do not vanish. If the user wishes to exercise control over this relative error, then the element size, h, must be varied, or we must use a higher degree interpolation for the element. If we think in terms of the bar element, then v and $\partial v/\partial x$ represent the displacement and strain, respectively. The contribution to the error represents the strain gradient (and stress gradient). Therefore, we must use our engineering judgment to make the element size, h, small in regions of large strain gradients (stress concentrations). Conversely, where the strain gradients are small, we can increase the element size, h, to reduce the computational cost. A similar argument can be stated for the heat conduction problem. Then, v is the temperature, $\partial v/\partial x$ describes the temperature gradient (heat flux), and the error is proportional to the flux gradient. If one does not wish to vary the element sizes, h, then to reduce the error, one must add higher order polynomial terms of the element interpolation functions so that the second derivative is present in the element. These two approaches to error control are known as the *h-method* and the *p-method,* respectively.

The previous comments have assumed the use of a uniform mesh, that is, h was the same for all elements in the mesh. Thus, the above error discussions have not considered the interaction of adjacent elements. The effects of adjacent element sizes have been evaluated for the case of a continuous bar subject to an axial load. An error term, in the governing differential equation, due to the finite element approximation at node j has been shown to be

$$E = -\frac{h}{3}(1-a)\frac{\partial^3 v}{\partial x^3}(x_j) + \frac{h^2}{12}\left(\frac{1+a^3}{1+a}\right)\frac{\partial^4 v}{\partial x^4}(x_j) + \cdots \qquad (4.37)$$

where h is the size of one element and ah is the size of the adjacent element. Here it is seen that for a smooth variation ($a \doteq 1$) or a uniform mesh ($a = 1$), the error in the approximated ODE is of order h squared. However, if the adjacent element sizes differ greatly ($a \neq 1$), then a larger error of order h is present. This suggests that it is desirable to have a gradual change in element sizes when possible. One should avoid placing a small element adjacent to one that is many times larger. Today the process of error estimation in a finite element analysis is a well established field of applied mathematics. This knowledge can be incorporated into a finite element software system. The MODEL code has this ability.

4.9 Gradient Estimates *

In our finite element calculations we often have a need for accurate estimates of the derivatives of the primary variable. For example, in plane stress or plane strain analysis,

the primary unknowns which we compute are the displacement components of the nodes. However, we often are equally concerned about the strains and stresses which are computed from the derivatives of the displacements. Likewise, when we model an ideal fluid with a velocity potential, we actually have little or no interest in the computed potential; but we are very interested in the velocity components which are the derivatives of the potential. A logical question at this point is: what location in the element will give me the most accurate estimate of derivatives? Such points are called *optimal points* or *Barlow points* [3] or *superconvergent points*. A heuristic argument for determining their location can be easily presented. Let us begin by recalling some of our previous observations. In Sec.s 2.6.2 and 3.4, we found that our finite element solution example was an *interpolate* solution, that is, it was exact at the node points and approximate elsewhere. Such accuracy is rare but, in general, one finds that the computed values of the primary variable are most accurate at the node points. Thus, for the sake of simplicity we will assume that the element's nodal values are exact.

Recall that we have taken our finite element approximation to be a polynomial of some specific order, say m. If the exact solution is also a polynomial of order m, then our finite element solution will be exact everywhere in the element. In addition, the finite element derivative estimates will also be exact. It is rare to have such good luck. In general, we must expect our results to only be approximate. However, we can hope for the next best thing to an exact solution. That would be where the exact solution is a polynomial that is one order higher, say $n = m + 1$, than our finite element polynomial. Let the subscripts E and F denote the exact and finite element solutions, respectively. Consider a one-dimensional formulation in natural coordinates, $-1 < a < +1$. Then the exact solution could be written as

$$U_E(a) = \mathbf{P}_E(a)\,\mathbf{V}_E = \begin{bmatrix} 1 & a & a^2 \cdots a^m & a^n \end{bmatrix} \mathbf{V}_E ,$$

and our approximate finite element polynominal solution would be

$$U_F(a) = \mathbf{P}_F(a)\,\mathbf{V}_F = \begin{bmatrix} 1 & a & a^2 \cdots a^m \end{bmatrix} \mathbf{V}_F$$

where $n = (m + 1)$, as assumed above. In the above $\mathbf{V}_E$ and $\mathbf{V}_F$ represent different vectors of unknown constants. In the domain of a typical element, these two forms should be almost equal. If we assume that they are equal at the nodes, then we can equate $u_E(a_j) = u_F(a_j)$ where a_j is the local coordinate of node j. Then the following identities are obtained: $\mathbf{P}_F(a_j)\,\mathbf{V}_F = \mathbf{P}_E(a_j)\,\mathbf{V}_E$, $1 \le k \le m$, or symbolically

$$\mathbf{A}_F\,\mathbf{V}_F = \mathbf{A}_E\,\mathbf{V}_E \tag{4.38}$$

where the rectangular array $\mathbf{A}_E$ has one more column than the square matrix $\mathbf{A}_F$, but otherwise they are the same. Indeed, upon closer inspection we should observe that $\mathbf{A}_E$ can be partitioned into a square matrix that is the same as $\mathbf{A}_F$ and an additional column so that $\mathbf{A}_E = [\,\mathbf{A}_F \mid \mathbf{C}_E\,]$ where the column is $\mathbf{C}_E^T = [\,a_1^n \;\; a_2^n \;\; a_3^n \cdots a_m^n\,]$. If we solve Eq. (4.38) we can relate the finite element constants, $\mathbf{V}_F$, to the exact constants, $\mathbf{V}_E$, at the nodes of the element. Then inverting matrxi $\mathbf{A}_F$, Eq. (4.38) gives $\mathbf{V}_F = \mathbf{A}_F^{-1}\,\mathbf{A}_E\,\mathbf{V}_E = [\,\mathbf{I} \mid \mathbf{A}_F^{-1}\,\mathbf{C}_E\,]\,\mathbf{V}_E = [\,\mathbf{I} \mid \mathbf{E}\,]\,\mathbf{V}_E$ or simply

$$\mathbf{V}_F = \mathbf{K}\,\mathbf{V}_E \tag{4.39}$$

where $\mathbf{K} = \mathbf{A}_F^{-1}\,\mathbf{A}_E$ is a rectangular matrix with constant coefficients. Therefore, we can return to Eq. (4.38) and relate everything to $\mathbf{V}_E$. This gives $u_F(a) = \mathbf{P}_F(a)\,\mathbf{K}\mathbf{V}_E = \mathbf{P}_E(a)\,\mathbf{V}_E = u_E(a)$ so that for arbitrary $\mathbf{V}_E$, one probably has the finite element polynomial and the exact polynomial related by $\mathbf{P}_F(a)\,\mathbf{K} = \mathbf{P}_E(a)$. Likewise, the derivatives of this relation should be approximately equal.

As an example, assume a quadratic finite element in one-dimensional natural coordinates, $-1 < a < +1$. The exact solution is assumed to be cubic. Therefore,

$$\mathbf{P}_F = \left\lfloor\, 1 \quad a \quad a^2 \,\right\rfloor, \qquad \mathbf{V}_F^T = \lfloor\, V_1 \quad V_2 \quad V_3 \,\rfloor_F\,,$$
$$\mathbf{P}_E = \left\lfloor\, 1 \quad a \quad a^2 \;\middle|\; a^3 \,\right\rfloor, \qquad \mathbf{V}_E^T = \lfloor\, V_1 \quad V_2 \quad V_3 \mid V_4 \,\rfloor_E\,.$$

Selecting the nodes at the standard positions of $a_1 = -1$, $a_2 = 0$, and $a_3 = 1$ gives:

$$\mathbf{A}_F = \begin{bmatrix} 1 & -1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \qquad \mathbf{A}_F^{-1} = \frac{1}{2}\begin{bmatrix} 0 & 2 & 0 \\ -1 & 0 & 1 \\ 1 & -2 & 1 \end{bmatrix},$$

$$\mathbf{A}_E = \left[\begin{array}{ccc|c} 1 & -1 & 1 & -1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{array}\right], \qquad \mathbf{C}_E = \begin{Bmatrix} -1 \\ 0 \\ 1 \end{Bmatrix},$$

$$\mathbf{A}_F^{-1}\,\mathbf{C}_E = \begin{Bmatrix} 0 \\ 1 \\ 0 \end{Bmatrix} \equiv \mathbf{E}, \qquad \mathbf{K} = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{array}\right].$$

For an interpolate solution, the two equivalent forms are exact at the three nodes ($a = \pm 1$, $a = 0$) and inside the element. Then the product expands to $\mathbf{P}_F\,\mathbf{K} = [\,1 \quad a \quad a^2 \quad a\,]$. Only the last polynomial term differs from $\mathbf{P}_E$. By inspection we see that term is $a\,V_4 = a^3\,V_4$ which is valid when a is evaluated at any of the three nodes. Equating the first derivatives at the optimum point a_0,

$$\lfloor\, 0 \quad 1 \quad 2a_0 \quad 1 \,\rfloor = \left\lfloor\, 0 \quad 1 \quad 2a_0 \quad 3a_0^2 \,\right\rfloor,$$

or simply $1 = 3a_0^2$ so that $a_0 = \pm 1/\sqrt{3}$. These are the usual Gauss points used in the two point integration rule. Similarly, the *optimal location*, a_s, for the second derivative is found from

$$\lfloor\, 0 \quad 0 \quad 2 \quad 0 \,\rfloor = \lfloor\, 0 \quad 0 \quad 2 \quad 6a_s \,\rfloor,$$

so that $a_s = 0$, the center of the element. The same sort of procedure can be applied to 2-D and 3-D elements. Generally, we find that derivative estimates are least accurate at the nodes. The derivative estimates are usually most accurate at the tabulated integration points. That is indeed fortunate, since it means we get a good approximation of the element square matrix. The typical sampling positions for the C^0 quadratic elements are shown in Fig. 4.9.1. The C^1 line elements have the same points except that the function and slope are most accurate at the end points while the best second and third derivative

locations are at the marked interior points. It is easy to show that the center of the linear element is the optimum position for sampling the first derivative. Since the front of partition **K** is an identity matrix, **I**, we are really saying that an exact nodal interpolate solution implies that $\mathbf{P}_F(a)\,\mathbf{A}_F^{-1}\,\mathbf{C}_E = a^n$. Let the vector $\mathbf{A}_F^{-1}\,\mathbf{C}_E$ denote an extrapolation vector, say **E**. Then, the derivatives would be the same in the two systems at points where

$$\left(\frac{d^k}{da^k}\ \mathbf{P}_F(a)\right)\mathbf{E} \;=\; \left(\frac{d^k}{da^k}\ a^n\right), \qquad 0 \le k \le n-1. \tag{4.40}$$

For example, the above quadratic element interpolate of a cubic solution gave

$$\begin{array}{lll} k=0, & [\,1 & a & a^2\,] \\ k=1, & [\,0 & 1 & 2a\,] \\ k=2, & [\,0 & 0 & 2\,] \end{array} \left\{\begin{array}{c} 0 \\ 1 \\ 0 \end{array}\right\} \;=\; \begin{array}{r} a^3 \\ 3a^2 \\ 6a \end{array}$$

which are only satisfied for

k	a_k
0	$-1,\ 0,\ +1$
1	$\pm 1/\sqrt{3}$
2	0

4.10 Exercises

1. For a one-dimensional quadratic element use the unit coordinate interpolation functions in Fig. 4.4.1 to evaluate the matrices:

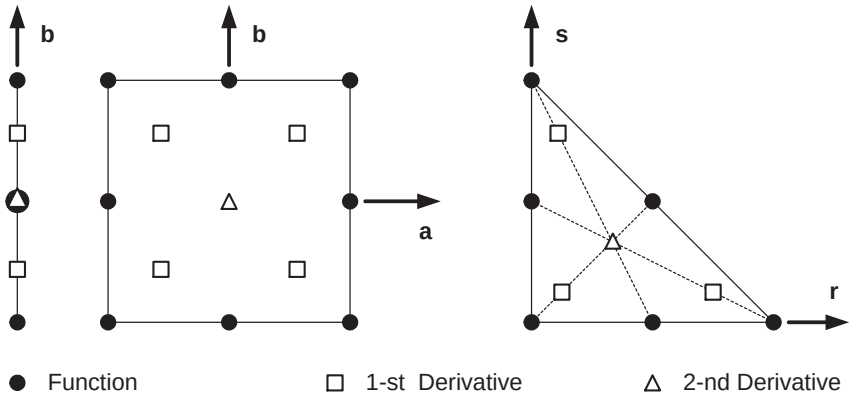

Figure 4.9.1 Barlow points for quadratic elements

$$a)\quad \mathbf{C}^e = \int_{L^e} \mathbf{H}^T\, dx, \quad b)\quad \mathbf{M}^e = \int_{L^e} \mathbf{H}^T\, \mathbf{H}\, dx\,,$$

$$c)\quad \mathbf{S}^e = \int_{L^e} \frac{d\mathbf{H}^T}{dx}\,\frac{d\mathbf{H}}{dx}\, dx\,, \quad d)\quad \mathbf{U}^e = \int_{L^e} \mathbf{H}^T\,\frac{d\mathbf{H}}{dx}\, dx\,.$$

Also give the sum of all of the coefficients of each matrix.

2. Solve the above problem by using the natural coordinate version, $-1 \le n \le 1$, from Fig, 4.4.1.

3. Solve Problem 3.8 using the least squares finite element method instead.

4. Solve Problem 3.9 using the least squares finite element method instead.

5. Solve Problem 3.10 using the least squares finite element method instead.

4.11 References

[1] Abramowitz, M. and Stegun, I.A., *Handbook of Mathematical Functions*, National Bureau of Standards (1964).

[2] Babuska, I., Griebel, M., and Pitkaranta, J., "The Problem of Selecting Shape Functions for a *p*-Type Finite Element," *Int. J. Num. Meth. Eng.*, **28**, pp. 1891−1908 (1989).

[3] Barlow, J., "Optimal Stress Locations in Finite Element Models," *Int. J. Num. Meth. Eng.*, **10**, pp. 243−251 (1976).

[4] Desai, C.S. and , T. Kundu, *Introduction to the Finite Element Method*, Boca Raton, FL: CRC Press (2001).

[5] Krishnamoorthy, C.S., *Finite Element Analysis: Theory and Programming*, New York: McGraw-Hill (1994).

[6] Szabo, B. and Babuska, I., *Finite Element Analysis*, New York: John Wiley (1991).

[7] Tong, P. and Rossettos, J.N., *Finite Element Method: Basic Techniques and Implementation*, MIT Press (1977).

[8] Zienkiewicz, O.C. and Taylor, R.L., *The Finite Element Method*, 5th Edition, London: Arnold (2000).

