4

4- and 8-Node Iso-P Quadrilateral Ring Elements

TABLE OF CONTENTS

§4.1. Introduction

This Chapter illustrates the computer implementation of isoparametric *quadrilateral* elements for the axisymmetric problem. These are called *ring* elements. Triangles, which present some programming quirks, are not described in these Notes. For details on those the reader may consult Chapter 24 of the Introduction to Finite Elements Notes.

Two specific elements are covered. They are identified by the following type labels.

Quad4 The standard 4-node isoparametric quadrilateral. This is usually processed with a 2×2 Gauss integration rule, which represents full integration.

Quad8RI The 8-node isoparametric quadrilateral. This is often processed by Reduced Integration: a 2×2 Gauss rule, whence the label. This rule results in rank deficiency, but this is generally harmless. It can also integrated with a 3×3 rule for safety, but performance suffers.

A third element was supposed to be described here:

Quad4SRI The 4-node isoparametric quadrilateral processed by Selective Reduced Integration (SRI). Implementation, however, has not been completed.

The element description that follows covers the computation of the element stiffness matrix, consistent node force vector for a body force field, consistent node force for surface tractions, and recovery of element stresses from displacements.

We consider the implementation of the 4-node and 8-node quadrilateral ring elements for axisymmetric solid analysis. The element cross sections are depicted in Figure 4.1.

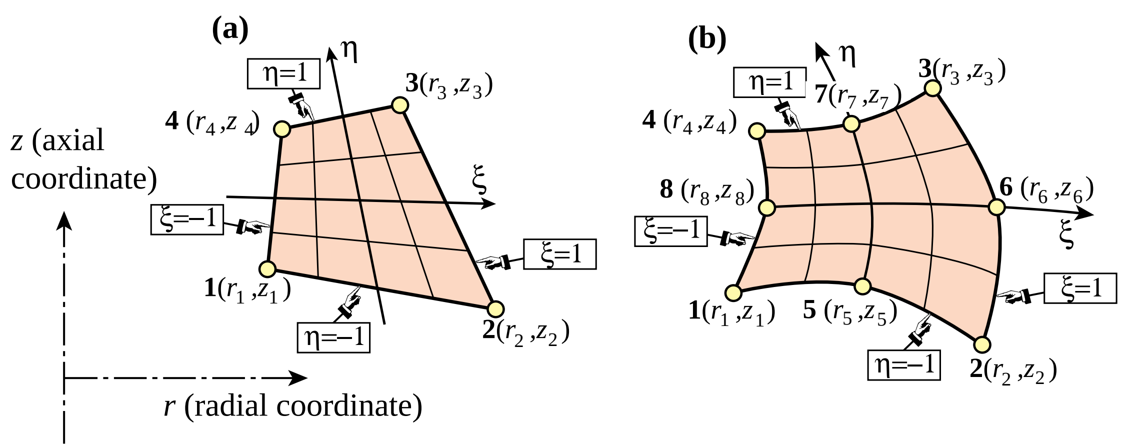

FIGURE 4.1. The 4-node and 8-node iso-P quadrilateral ring elements described in this Chapter.

For 2D and 3D elements iso-P elements it is convenient to break up the implementation into *application dependent* and *application independent* modules, as sketched in Figure 4.2. The application independent modules can be "reused" in other FEM applications, for example to form thermal, fluid or electromagnetic elements.

For the 4-node quadrilateral studied here, the configuration shown in Figure 4.2 is done through the following modules:

```
Quad4IsoPRingStiffness -       forms Ke of standard isoP 4-node quad ring
   QuadGaussRuleInfo -          returns Gauss quadrature product rules of order 1-4
   IsoQuad4ShapeFunDer -        evaluates shape functions and their x/y derivatives

Quad4isoPRingForces -       forms traction force fe of 4-node standard isoP quad ring
   QuadGaussRuleInfo -          returns Gauss quadrature product rules of order 1-4
   IsoQuad4ShapeFun -           evaluates shape functions

Quad4isoPRingTracForces -       forms traction force fe of 4-node standard isoP quad
ring
   QuadGaussRuleInfo -          returns Gauss quadrature product rules of order 1-4
   IsoQuad4ShapeFun -           evaluates shape functions

Quad4isoPRingStresses -       evaluates stresses fe of 4-node standard isoP quad ring
   QuadGaussRuleInfo -          returns Gauss quadrature product rules of order 1-4
   IsoQuad4ShapeFunDer -           evaluates shape functions
```

A diagrammatic representation of the module organization is provided in Figure 4.2.

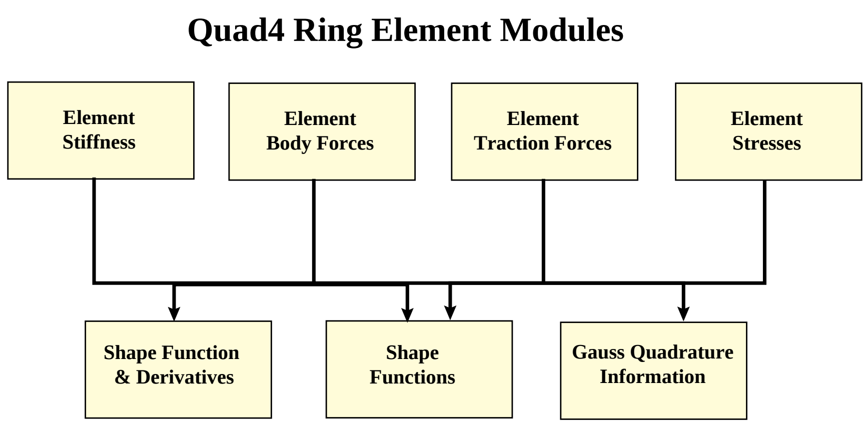

FIGURE 4.2. Hierarchical organization of ring element modules.

These modules are presented in the following subsections, except for the Gauss quadrature information modules, which were described in the previous Chapter.

```
Quad4IsoPRingShapeFunDer[ncoor_,qcoor_,Jcons_]:= Module[
  { r1,r2,r3,r4,z1,z2,z3,z4,ξ,η,Nf,dNr,dNz,A0,A1,A2,Jdet},
  {{r1,z1},{r2,z2},{r3,z3},{r4,z4}}=ncoor; {ξ,η}=qcoor;
   Nf={(1-ξ)*(1-η),(1+ξ)*(1-η),(1+ξ)*(1+η),(1-ξ)*(1+η)}/4;
   A0=((r3-r1)*(z4-z2)-(r4-r2)*(z3-z1))/2;
   A1=((r3-r4)*(z1-z2)-(r1-r2)*(z3-z4))/2;
   A2=((r2-r3)*(z1-z4)-(r1-r4)*(z2-z3))/2;
   Jdet=(A0+A1*ξ+A2*η)/4; If [Jcons,Jdet=A0/4];
   dNr={z2-z4+(z4-z3)*ξ+(z3-z2)*η,z3-z1+(z3-z4)*ξ+(z1-z4)*η,
    z4-z2+(z1-z2)*ξ+(z4-z1)*η,z1-z3+(z2-z1)*ξ+(z2-z3)*η}/(8*Jdet);
   dNz={r4-r2+(r3-r4)*ξ+(r2-r3)*η,r1-r3+(r4-r3)*ξ+(r4-r1)*η,
    r2-r4+(r2-r1)*ξ+(r1-r4)*η,r3-r1+(r1-r2)*ξ+(r3-r2)*η}/(8*Jdet);
   Return[{Nf,dNr,dNz,Jdet}]
];
```

FIGURE 4.3. Shape function module for 4-node bilinear quadrilateral ring element.

§4.2. The 4-Node Quadrilateral Ring Element

This is the axisymmetric solid version of the well known isoparametric quadrilateral with bilinear shape functions. The element has 4 nodes and 8 displacement degrees of freedom arranged as

$$\mathbf{u}^e = [\, u_{r1} \quad u_{z1} \quad u_{r2} \quad u_{z2} \quad u_{r3} \quad u_{z3} \quad u_{r4} \quad u_{z4} \,]^T . \tag{4.1}$$

§4.2.1. Shape Function Module

Module `Quad4IsoPRingShapeFunDer`, listed in Figure 4.3, computes the shape functions N_i^e, $i = 1, 2, 3, 4$ and their partial derivatives with respect to r and z at a specified point in the element. Usually this module is called at sample points of a Gauss quadrature rule, but it may also be used with symbolic inputs to get information for an arbitrary point at $\{\xi, \eta\}$. The element geometry is defined by the 8 coordinates $\{r_i, z_i\}$, $i = 1, 2, 3, 4$. These are collected in arrays

$$\mathbf{r} = [\, r_1 \quad r_2 \quad r_3 \quad r_4 \,]^T , \qquad \mathbf{z} = [\, z_1 \quad z_2 \quad z_3 \quad z_4 \,]^T . \tag{4.2}$$

We will use the abbreviations $r_{ij} = r_i - r_j$ and $z_{ij} = z_i - z_j$ for coordinate differences. The shape functions and their partial derivatives with respect to the quadrilateral coordinates are collected in the arrays

$$\begin{aligned} \mathbf{N} &= \tfrac{1}{4}\,[\,(1-\xi)(1-\eta) \quad (1+\xi)(1-\eta) \quad (1+\xi)(1+\eta) \quad (1-\xi)(1+\eta)\,], \\ \mathbf{N}_{,\xi} &= \frac{\partial \mathbf{N}}{\partial \xi} = \tfrac{1}{4}\,[\,-1+\eta \quad 1-\eta \quad 1+\eta \quad -1-\eta\,], \\ \mathbf{N}_{,\eta} &= \frac{\partial \mathbf{N}}{\partial \eta} = \tfrac{1}{4}\,[\,-1+\xi \quad -1-\xi \quad 1+\xi \quad 1-\xi\,]. \end{aligned} \tag{4.3}$$

The Jacobian matrix is

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial r}{\partial \xi} & \dfrac{\partial z}{\partial \xi} \\ \dfrac{\partial r}{\partial \eta} & \dfrac{\partial z}{\partial \eta} \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{N}_{,\xi}\,\mathbf{r} & \mathbf{N}_{,\xi}\,\mathbf{z} \\ \mathbf{N}_{,\eta}\,\mathbf{r} & \mathbf{N}_{,\eta}\,\mathbf{z} \end{bmatrix}. \tag{4.4}$$

Expanding the inner products yields the explicit expressions

$$\begin{aligned}
J_{11} &= \tfrac{1}{4}(r_{21}+r_{34}+(r_{12}+r_{34})\,\eta), \quad J_{12} = \tfrac{1}{4}(z_{21}+z_{34}+(z_{12}+z_{34})\,\eta),\\
J_{21} &= \tfrac{1}{4}(r_{32}+r_{41}+(r_{12}+r_{34})\,\xi), \quad J_{22} = \tfrac{1}{4}(z_{32}+z_{41}+(z_{12}+z_{34})\,\xi),\\
J &= \det(\mathbf{J}) = J_{11}J_{22}-J_{12}J_{21} = \tfrac{1}{4}(A_0+A_1\,\xi+A_2\,\eta),
\end{aligned} \tag{4.5}$$

in which

$$A_0 = \tfrac{1}{2}(r_{31}z_{42}-r_{42}z_{31}), \quad A_1 = \tfrac{1}{2}(r_{34}z_{12}-r_{12}z_{34}), \quad A_2 = \tfrac{1}{2}(r_{23}z_{14}-r_{14}z_{23}). \tag{4.6}$$

The inverse Jacobian is obtained by explicit inversion. Finally the $\{r, z\}$ derivatives produced by the chain rule emerge as the explicit formulas

$$\mathbf{N}_{,r} = \frac{\partial \mathbf{N}}{\partial r} = \frac{1}{8J}\begin{bmatrix} z_{24}+z_{43}\,\xi+z_{32}\,\eta \\ z_{31}+z_{34}\,\xi+z_{14}\,\eta \\ z_{42}+z_{12}\,\xi+z_{41}\,\eta \\ z_{13}+z_{21}\,\xi+z_{23}\,\eta \end{bmatrix}, \quad \mathbf{N}_{,z} = \frac{\partial \mathbf{N}}{\partial z} = \frac{1}{8J}\begin{bmatrix} r_{42}+r_{34}\,\xi+r_{23}\,\eta \\ r_{13}+r_{43}\,\xi+r_{41}\,\eta \\ r_{24}+r_{21}\,\xi+r_{14}\,\eta \\ r_{31}+r_{12}\,\xi+r_{32}\,\eta \end{bmatrix}. \tag{4.7}$$

The logic of `Quad4IsoPRingShapeFunDer`, listed in Figure 4.3, implements the foregoing equations. The module is invoked as

```
{ Nf,dNr,dNz,Jdet }= Quad4IsoPRingShapeFunDer[ncoor,qcoor,Jcons]
```
(4.8)

where the arguments are

`ncoor` Quadrilateral node coordinates arranged in two-dimensional list form: `{ { r1,z1 },{ r2,z2 },{ r3,z3 },{ r4,z4 } }`.

`qcoor` Quadrilateral coordinates { ξ , η } of the point at which shape functions and derivatives are to be evaluated.

`Jcons` A logical flag. If `True`, the Jacobian determinant J is set to its value at the element center, namely, $A_0/4$, for any $\{\xi, \eta\}$. That setting is useful in certain research studies.

The module returns the list `{ Nf,dNr,dNz,Jdet }`, where

`Nf` Shape function values[1] arranged as list `{ N1,N2,N3,N4 }`.

`dNr` r shape function derivatives (4.7) arranged as `{ dNr1,dNr2,dNr3,dNr4 }`.

`dNz` z shape function derivatives (4.7) arranged as `{ dNz1,dNz2,dNz3,dNz4 }`.

`Jdet` Jacobian determinant.

§4.2.2. Element Stiffness Module

Module `Quad4IsoPRingStiffness`, listed in Figure 4.4, computes the stiffness matrix of a 4-noded iso-P quadrilateral ring element. The computation is carried out using numerical quadrature. It essentially follows the procedure outlined in the previous Chapter.

The module is invoked as

```
Ke = Quad4IsoPRingStiffness[ncoor,Emat,options]
```
(4.9)

[1] Note that `N` cannot be used as name of the list of shape function values, because that symbol is reserved.

```
Quad4IsoPRingStiffness[ncoor_,Emat_,options_]:=Module[
  {p=2,numer=False,Jcons=False,Kfac=1,qcoor,k,
   r1,r2,r3,r4,z1,z2,z3,z4,Nf,N1,N2,N3,N4,A0,Jdet,Be,
   dNr1,dNr2,dNr3,dNr4,dNz1,dNz2,dNz3,dNz4,rk,w,c,Ke,
  Ke0=Table[0,{8},{8}],modname="Quad4IsoPRingStiffness: "},
  If [Length[options]==1,{numer}=options];
  If [Length[options]==2,{numer,p}=options];
  If [Length[options]==3,{numer,p,Jcons}=options];
  If [Length[options]==4,{numer,p,Jcons,Kfac}=options];
  If [p<1||p>5, Print[modname,"illegal p:",p]; Return[Ke0]];
  {{r1,z1},{r2,z2},{r3,z3},{r4,z4}}=ncoor;
  A0=((r3-r1)*(z4-z2)-(r4-r2)*(z3-z1))/2;
  If [numer&&(A0<=0), Print[modname,"Neg or zero area"];
      Return[Ke0]];  Ke=Ke0;
  For [k=1,k<=p*p,k++,
      {qcoor,w}= QuadGaussRuleInfo[{p,numer},k];
     {Nf,{dNr1,dNr2,dNr3,dNr4},{dNz1,dNz2,dNz3,dNz4},
      Jdet}=Quad4IsoPRingShapeFunDer[ncoor,qcoor,Jcons];
      If [numer&&(Jdet<=0), Print[modname,"Neg or zero",
         " Gauss point Jacobian at k=",k]; Return[Ke0]];
      {N1,N2,N3,N4}=Nf; rk=r1*N1+r2*N2+r3*N3+r4*N4;
      Be={{ dNr1,   0, dNr2,   0, dNr3,   0, dNr4,   0},
          {    0,dNz1,    0,dNz2,    0,dNz3,    0,dNz4},
          {N1/rk,   0,N2/rk,   0,N3/rk,   0,N4/rk,   0},
          { dNz1,dNr1, dNz2,dNr2, dNz3,dNr3, dNz4,dNr4}};
      c=Kfac*w*rk*Jdet; If [numer,Be=N[Be]; c=N[c]];
      Ke+=c*Transpose[Be].(Emat.Be);
      ]; ClearAll[Ke0,Be]; Return[Ke] ];
```

FIGURE 4.4. Element stiffness formation module for 4-node iso-P quadrilateral ring.

The arguments are:

`ncoor` Quadrilateral node coordinates arranged in two-dimensional list form: `{ { r1,z1 },{ r2,z2 },{ r3,z3 },{ r4,z4 } }`.

`Emat` The 4×4 matrix of elastic moduli:

$$\mathbf{E} = \begin{bmatrix} E_{11} & E_{12} & E_{13} & E_{14} \\ E_{12} & E_{22} & E_{23} & E_{24} \\ E_{13} & E_{23} & E_{33} & 0 \\ E_{14} & E_{24} & 0 & E_{44} \end{bmatrix}, \tag{4.10}$$

arranged as a two-dimensional list array: `{ { E11,E12,E13,E14 }, { E12,E22,E23,E24 },{ E13,E23,E33,0 },{ E14,E24,0,E44 } }`.

Note that $E_{34} = 0$ to satisfy axisymmetric behavior assumptions. If the material is isotropic, with elastic modulus E and Poisson ratio ν,

$$\mathbf{E} = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & 0 \\ \nu & 1-\nu & \nu & 0 \\ \nu & \nu & 1-\nu & 0 \\ 0 & 0 & 0 & \frac{1}{2}-\nu \end{bmatrix}. \tag{4.11}$$

`options` A list of processing options. This list may be either empty or contain up to 4 items. Possible configurations are `{ }`, `{ numer }`, `{ numer,p }`, `{ numer,p,Jcons }`, or `{ numer,p,Jcons,Kfac }`.

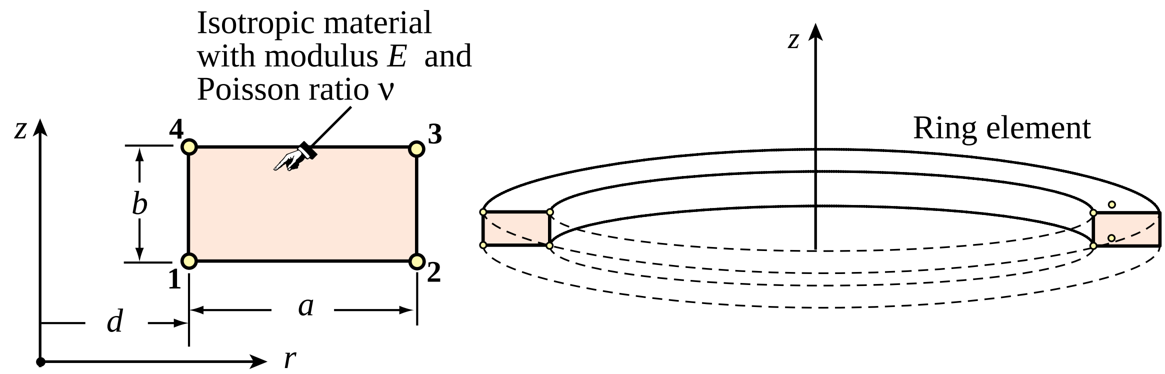

FIGURE 4.5. Test quadrilateral ring element geometry.

`numer` is a logical flag with value `True` or `False`. If `True`, the computations are forced to proceed in floating point arithmetic. For symbolic or exact arithmetic work set `numer` to `False`. If omitted, `False` is assumed.

`p` is an integer specifying that the Gauss product rule used in computing $\mathbf{K}^e$ is to have `p` points in each direction. It may be 1 through 4. For rank sufficiency, `p` must be 2 or higher. If `p` is 1 the element will be rank deficient by three. If omitted `p` = 2 is assumed.

`Jcons` is a logical flag with value `True` or `False`. If `True` the Jacobian determinant at the element center is assumed to be constant over the element, even if it has arbitrary geometry. This is useful in centain research studies. If omitted, `False` is assumed.

`Kfac` is a ring-circumference-span factor by which the stiffness matrix will be scaled. Typically `Kfac=1` to make the ring element span one radian, `Kfac=2`π to make a complete circle. If omitted, `Kfac` = 1 is assumed.

As function value the module returns

`Ke` a 8×8 symmetric matrix pertaining to the arrangement (4.2) of element node displacements. If an error is detected during processing, a zero matrix is returned.

Example 4.1. The stiffness module is tested on the geometry identified in Figure 4.5. The cross section is a rectangle dimensioned $a \times b$ with sides parallel to the $\{r, z\}$ axes. The distance of the leftmost side to the z axis is d. The material is isotropic with modulus E and Poisson's ratio ν.

The script of Figure 4.8 computes and prints the stiffness of the test element shown in for $E = 96$, $\nu = 1/3$, $a = 4$, $b = 2$ and $d = 0$. The default `Kfac` = 1 is used. Nodes 1 and 2 sit on the z axes. The value of `p` is changed in a loop. The flag `numer` is set to `True` to use floating-point computation for speed. The computed entries of $\mathbf{K}^e$ are exact integers for all values of p:

```
ClearAll[Em,ν,a,b,d,h,p,num];
Em=96; ν=1/3; a=4; b=2; d=0; Kfac=2*Pi; Kfac=1;
ncoor={{d,0},{a+d,0},{a+d,b},{d,b}}; num=False;
Emat=Em/((1+ν)*(1-2*ν))*{{1-ν,ν,ν,0},{ν,1-ν,ν,0},
         {ν,ν,1-ν,0},{0,0,0,1/2-ν}};
Print["Emat=",Emat//MatrixForm];
For [p=1,p<=4,p++,  Print["Gauss rule p=",p];
     Ke=Quad4IsoPRingStiffness[ncoor,Emat,Kfac,{num,p}];
     Ke=Simplify[Ke];  Print["Ke=",Ke//MatrixForm];
     Print["Eigenvalues of Ke=",Chop[Eigenvalues[N[Ke]]]];
];
```

FIGURE 4.6. Driver for exercising the `Quad4IsoPRingStiffness` module of Figure 4.4 using the ring element geometry of Figure 4.5, with $E = 96$, $\nu = 1/3$, $a = 4$, $b = 2$ $d = 0$ and four Gauss product integration rules.

$$\mathbf{K}^e_{1\times 1} = \begin{bmatrix} 72 & 18 & 36 & -18 & -36 & -18 & 0 & 18 \\ 18 & 153 & -54 & 135 & -90 & -153 & -18 & -135 \\ 36 & -54 & 144 & -90 & 72 & 54 & -36 & 90 \\ -18 & 135 & -90 & 153 & -54 & -135 & 18 & -153 \\ -36 & -90 & 72 & -54 & 144 & 90 & 36 & 54 \\ -18 & -153 & 54 & -135 & 90 & 153 & 18 & 135 \\ 0 & -18 & -36 & 18 & 36 & 18 & 72 & -18 \\ 18 & -135 & 90 & -153 & 54 & 135 & -18 & 153 \end{bmatrix} \tag{4.12}$$

$$\mathbf{K}^e_{2\times 2} = \begin{bmatrix} 168 & -12 & 24 & 12 & -24 & -36 & 48 & 36 \\ -12 & 108 & -24 & 84 & -72 & -102 & -36 & -90 \\ 24 & -24 & 216 & -120 & 0 & 72 & -24 & 72 \\ 12 & 84 & -120 & 300 & -72 & -282 & 36 & -102 \\ -24 & -72 & 0 & -72 & 216 & 120 & 24 & 24 \\ -36 & -102 & 72 & -282 & 120 & 300 & -12 & 84 \\ 48 & -36 & -24 & 36 & 24 & -12 & 168 & 12 \\ 36 & -90 & 72 & -102 & 24 & 84 & 12 & 108 \end{bmatrix} \tag{4.13}$$

$$\mathbf{K}^e_{3\times 3} = \begin{bmatrix} 232 & -12 & 24 & 12 & -24 & -36 & 80 & 36 \\ -12 & 108 & -24 & 84 & -72 & -102 & -36 & -90 \\ 24 & -24 & 216 & -120 & 0 & 72 & -24 & 72 \\ 12 & 84 & -120 & 300 & -72 & -282 & 36 & -102 \\ -24 & -72 & 0 & -72 & 216 & 120 & 24 & 24 \\ -36 & -102 & 72 & -282 & 120 & 300 & -12 & 84 \\ 80 & -36 & -24 & 36 & 24 & -12 & 232 & 12 \\ 36 & -90 & 72 & -102 & 24 & 84 & 12 & 108 \end{bmatrix} \tag{4.14}$$

$$\mathbf{K}^e_{4\times 4} = \begin{bmatrix} 280 & -12 & 24 & 12 & -24 & -36 & 104 & 36 \\ -12 & 108 & -24 & 84 & -72 & -102 & -36 & -90 \\ 24 & -24 & 216 & -120 & 0 & 72 & -24 & 72 \\ 12 & 84 & -120 & 300 & -72 & -282 & 36 & -102 \\ -24 & -72 & 0 & -72 & 216 & 120 & 24 & 24 \\ -36 & -102 & 72 & -282 & 120 & 300 & -12 & 84 \\ 104 & -36 & -24 & 36 & 24 & -12 & 280 & 12 \\ 36 & -90 & 72 & -102 & 24 & 84 & 12 & 108 \end{bmatrix} \tag{4.15}$$

As can be seen entries change substantially in going from $p = 1$ to $p = 2$. From then on only four entries, associated with the r stiffness at nodes 1 and 4, change. The eigenvalues of these matrices are:

Rule	Eigenvalues of $\mathbf{K}^e$ for varying integration rule							
1×1	667.794	180.000	124.206	72.000	0	0	0	0
2×2	745.201	261.336	248.750	129.451	100.389	88.598	10.275	0
3×3	745.446	330.628	266.646	133.236	126.343	98.690	11.011	0
4×4	745.716	397.372	272.092	144.542	135.004	101.908	11.365	0

(4.16)

The stiffness matrix computed by the one-point rule is rank deficient by three. For $p = 2$ and up it has the correct rank of 7. The eigenvalues do not change appreciably after $p = 2$. Because the nonzero eigenvalues measure the internal energy taken up by the element in deformation eigenmodes, it can be seen that raising the order of the integration stiffens the element.

§4.2.3. Body Force Module

Module `Quad4IsoPRingBodyForces`, listed in Figure 4.7 computes the consistent force vector associated with a body force field $\vec{\mathbf{b}} = \{b_x, b_y\}$ specified over a four-node iso-P quadrilateral ring element. The field is assumed to be given per unit of volume, in radial-axial component-wise form. Two common scenarios for this kind of forcing effect are:

1. The element is subjected to a gravity acceleration field g due to self-weight in the $-z$ direction. Then $b_r = 0$ and $b_z = -\rho\, g$, where ρ is the mass density of the element material.
2. The element rotates at constant angular velocity ω (radians per second) around the axis of revolution z. Then $b_r = \rho\, \omega^2\, r$ and $b_z = 0$.

The force vector is computed by Gauss numerical integration as described in the previous chapter.

The module is invoked as

```
Ke = Quad4IsoPRingBodyForces[ncoor,bfor,options]
```
(4.17)

The arguments are:

`ncoor` Same as in `Quad4IsoPRingStiffness`

`bfor` Specifies body force field (forces per unit of volume) over the element. Two specification forms are allowed.

One-dimensional list: `{ br,bz }`

Two-dimensional list: `{ { br1,bz1 },{ br2,bz2 },{ br3,bz3 },{ br4,bz4 } }`

In the first form the body force field is taken to be uniform over the element, with radial component `br` and axial component `bz`.

The second form assumes body forces to vary over the element. Radial and axial components are specified at the four corners; for example `{ br1,bz1 }` are the values of b_r and b_z at corner 1. From this information the field is interpolated over the element using the iso-P bilinear shape functions.

`options` Same as in `Quad4IsoPRingStiffness`.

As function value the module returns

```
Quad4IsoPRingBodyForces[ncoor_,bfor_,options_]:=Module[
  {p=2,numer=False,Jcons=False,Kfac=1,qcoor,k,m,
  r1,r2,r3,r4,z1,z2,z3,z4,N1,N2,N3,N4,dNr,dNz,Jdet,Be,
  br1,bz1,br2,bz2,br3,bz3,br4,bz4,brc,bzc,bk,rk,w,c,fe,
  fe0=Table[0,{8}],modname="Quad4IsoPRingBodyForces: "},
  If [Length[options]==1,{numer}=options];
  If [Length[options]==2,{numer,p}=options];
  If [Length[options]==3,{numer,p,Jcons}=options];
  If [Length[options]==4,{numer,p,Jcons,Kfac}=options];
  If [p<1||p>5, Print[modname,"p out of range"]; Return[fe0]];
  {{r1,z1},{r2,z2},{r3,z3},{r4,z4}}=ncoor;
  A0=((r3-r1)*(z4-z2)-(r4-r2)*(z3-z1))/2;
  If [numer&&(A0<=0), Print[modname,"Neg or zero area"];
      Return[fe0]]; fe=fe0; m=Length[bfor];
  If [m!=2&&m!=4, Print[modname," Illegal bfor"]; Return[fe0]];
  If [m==2, br1=br2=br3=br4=bfor[[1]];bz1=bz2=bz3=bz4=bfor[[2]]];
  If [m==4,{{br1,bz1},{br2,bz2},{br3,bz3},{br4,bz4}}=bfor];
  For [k=1,k<=p*p,k++,
      {qcoor,w}= QuadGaussRuleInfo[{p,numer},k];
     {{N1,N2,N3,N4},dNr,dNz,Jdet}=
           Quad4IsoPRingShapeFunDer[ncoor,qcoor,Jcons];
      If [numer&&(Jdet<=0), Print[modname,"Neg or zero",
         " Gauss point Jacobian at k=",k]; Return[fe0]];
      rk=r1*N1+r2*N2+r3*N3+r4*N4;  c=Kfac*w*Jdet*rk;
      brk=br1*N1+br2*N2+br3*N3+br4*N4;
      bzk=bz1*N1+bz2*N2+bz3*N3+bz4*N4;
      bk={N1*brk,N1*bzk,N2*brk,N2*bzk,
          N3*brk,N3*bzk,N4*brk,N4*bzk};
      If [numer,bk=N[bk]]; fe+=c*bk;
      ];  If[!numer, fe=Simplify[fe]];
      Return[fe] ];
```

FIGURE 4.7. Module that computes consistent node forces for a 4-noded quadrilateral ring element given a body force field.

`fe` Consistent force vector arranged { `fr1,fz1,fr2,fz2,fr3,fz3,fr4,fz4` } to represent

$$\mathbf{f}^e = [\, f_{r1} \quad f_{z1} \quad f_{r2} \quad f_{z2} \quad f_{r3} \quad f_{z3} \quad f_{r4} \quad f_{z4} \,]^T. \tag{4.18}$$

Example 4.2. Consider again the ring element of Figure 4.5. This is now exercised for body force computation, using the script listed in Figure 4.8. These specify $a = 6$, $b = 2$, $d = 1$, two body force distributions and two integration rules: `p=1` and `p=2`.

The uniform body force distribution $b_r = 3$ and $b_z = -1$ gives for the 1×1 and 2×2 integration rules:

$$\begin{aligned} \mathbf{f}^e_{1\times1} &= [\,36 \quad -12 \quad 36 \quad -12 \quad 36 \quad -12 \quad 36 \quad -12\,]^T \\ \mathbf{f}^e_{2\times2} &= [\,27 \quad -9 \quad 45 \quad -15 \quad 45 \quad -15 \quad 27 \quad -9\,]^T \end{aligned} \tag{4.19}$$

Note that $f_{r1} + f_{r2} + f_{r3} + f_{r4} = 144$ for both rules. Likewise for the $-z$ component. Thus the total force is conserved. The varying body force distribution $b_r = r$ and $b_z = 0$, which mimics a centrifugal force, gives for the 1×1 and 2×2 integration rules:

$$\begin{aligned} \mathbf{f}^e_{1\times1} &= [\,42 \quad 0 \quad 42 \quad 0 \quad 42 \quad 0 \quad 42 \quad 0\,]^T \\ \mathbf{f}^e_{2\times2} &= [\,29 \quad 0 \quad 70 \quad 0 \quad 70 \quad 0 \quad 29 \quad 0\,]^T \end{aligned} \tag{4.20}$$

```
ClearAll[a,b,d,h,p,numer];
a=6; b=2; d=1; br=3; bz=1;
Jcons=False; numer=True;
ncoor={{d,0},{a+d,0},{a+d,b},{d,b}};
For [p=1,p<=2,p++, Print["Gauss rule p=",p];
     fe=Quad4IsoPRingBodyForces[ncoor,{3,-1},{numer,p}];
     Print["fe =",Partition[fe,2]//MatrixForm];
     bfor={{1,0},{6,0},{6,0},{1,0}};
     fe=Quad4IsoPRingBodyForces[ncoor,bfor,{numer,p}];
     Print["fe =",Partition[fe,2]//MatrixForm];
     ];
```

FIGURE 4.8. Test statements to exercise body force module of Figure 4.7.

Here $f_{r1} + f_{r2} + f_{r3} + f_{r4} = 168$ for p = 1 but that sum is 198 for p = 2. The 2×2 rule captures a variable body force radial distribution better, as may be expected.

Trying with p = 3 or greater reproduces the results of the 2×2 product rule.

§4.2.4. Stress Recovery Module

Module `Quad4IsoPRingStresses`, listed in Figure 4.9, recovers stresses at the 4 corner nodes of the iso-P 4-node quadrilateral ring element, given its node displacements.

The procedure is as follows. The stresses are recovered at five sample points $k = 0, 1, 2, 3, 4$ with quadrilateral coordinates $\{\xi, \eta\} = \{0, 0\}, \{-g, -g\}, \{g, -g\}, \{g, g\}, \{-g, g\}$, in which $0 < g \le 1$, using the direct evaluation $\bar{\boldsymbol{\sigma}}_k = \mathbf{E}\,\mathbf{B}_k^e\,\mathbf{u}^e$. Here a bar over the stress symbol is used to mark a sample value. Perform a least-square bilinear fit over the 5 sample points assigning weight w_0 to sample at $\{\xi, \eta\} = \{0, 0\}$ and weight 1 to each of the samples at $\{\xi, \eta\} = \{\pm g, \pm g\}$. Evaluation of the fit at the corners $\{\xi, \eta\} = \{\pm 1, \pm 1\}$ yields

$$\begin{bmatrix} \sigma_{rr1} & \sigma_{zz1} & \sigma_{\theta\theta1} & \sigma_{rz1} \\ \sigma_{rr2} & \sigma_{zz2} & \sigma_{\theta\theta2} & \sigma_{rz2} \\ \sigma_{rr3} & \sigma_{zz3} & \sigma_{\theta\theta3} & \sigma_{rz3} \\ \sigma_{rr4} & \sigma_{zz4} & \sigma_{\theta\theta4} & \sigma_{rz4} \end{bmatrix} = \frac{1}{T_d} \begin{bmatrix} T_1 & T_2 & T_3 & T_4 & T_3 \\ T_1 & T_3 & T_2 & T_3 & T_4 \\ T_1 & T_4 & T_3 & T_2 & T_3 \\ T_1 & T_3 & T_4 & T_3 & T_2 \end{bmatrix} \begin{bmatrix} \bar{\sigma}_{rr0} & \bar{\sigma}_{zz0} & \bar{\sigma}_{\theta\theta0} & \bar{\sigma}_{rz0} \\ \bar{\sigma}_{rr1} & \bar{\sigma}_{zz1} & \bar{\sigma}_{\theta\theta1} & \bar{\sigma}_{rz1} \\ \bar{\sigma}_{rr2} & \bar{\sigma}_{zz2} & \bar{\sigma}_{\theta\theta2} & \bar{\sigma}_{rz2} \\ \bar{\sigma}_{rr3} & \bar{\sigma}_{zz3} & \bar{\sigma}_{\theta\theta3} & \bar{\sigma}_{rz3} \\ \bar{\sigma}_{rr4} & \bar{\sigma}_{zz4} & \bar{\sigma}_{\theta\theta4} & \bar{\sigma}_{rz4} \end{bmatrix}, \tag{4.21}$$

in which $T_1 = 4\,g^2\,w_0$, $T_2 = 4 + 4\,g^2 + w_0 + 2\,g\,(4 + w_0)$, $T_3 = 4\,g^2 - 4 - w_0$, $T_4 = 4 + 4\,g^2 + w_0 - 2\,g\,(4 + w_0)$ and $T_d = 4\,g^2\,(4 + w_0)$. The default values used in the least-square fit are $w_0 = 0$ and $g = 1/\sqrt{3}$, in which case $\{\xi, \eta\} = \{\pm g, \pm g\}$ are located at the sample points of the 2×2 Gauss product rule.

The module is invoked as

$$\texttt{Ke = Quad4IsoPRingStresses[ncoor,Emat,ue,options]} \tag{4.22}$$

The arguments are:

`ncoor`	Same as in `Quad4IsoPRingStiffness`
`Emat`	Same as in `Quad4IsoPRingStiffness`

`ue` The element node displacements arranged as a one-dimensional list: { `ur1,uz1, ur2,uz2,ur3,uz3,ur4,uz4` } representing the displacement vector

$$\mathbf{u}^e = [\,u_{r1} \quad u_{z1} \quad u_{r2} \quad u_{z2} \quad u_{r3} \quad u_{z3} \quad u_{r4} \quad u_{z4}\,]^T. \tag{4.23}$$

`options` A list of processing options. This list may be either empty or contain up to 4 items. Possible configurations are { }, { `numer` }, { `numer,g` } or { `numer,g,w0` }.

`numer` is a logical flag with value `True` or `False`. If `True`, the computations are forced to proceed in floating point arithmetic. For symbolic or exact arithmetic work set `numer` to `False`. If omitted, `False` is assumed.

`g` Defines location of 4 sample points within element from which stresses are extrapolated to the corners according to (4.21). If omitted the default g = $1/\sqrt{3}$ is assumed.

`w0` Weight used in the least-square extrapolator (4.21). If omitted the default $w_0 = 0$ is assumed.

As function value the module returns

sige computed corner stresses stored in a 4-entry, two-dimensional list: { { `sigrr1,sigzz1,sigtt1,sigrz1` }, { `sigrr2,sigzz2,sigtt2,sigrz2` }, { `sigrr3,sigzz3,sigtt3,sigrz3` }, { `sigrr4,sigzz4,sigtt4,sigrz4` } } to represent the array shown on the left hand side of (4.23).

```
ClearAll[Em,ν,a,b,d,err,ezz,grz,ur,uz,r,z];
Em=2500; ν=1/4;
{err,ezz,err,grz}={3/80,-1/40,3/80,4/50};
ncoor={{d,0},{a+d,0},{a+d,b},{d,b}}; num=False;
Emat=Em/((1+ν)*(1-2*ν))*{{1+ν,ν,ν,0},{ν,1+ν,ν,0},
       {ν,ν,1+ν,0},{0,0,0,1/2-ν}};
{err,ezz,err,grz}={3/80,-1/40,3/80,4/50};
ur[r_,z_]:=err*r; uz[r_,z_]:=ezz*z+grz*r;
ue=Table[{0,0},{4}];
For [n=1,n<=4,n++,{rn,zn}=ncoor[[n]];
     ue[[n]]={ur[rn,zn],uz[rn,zn]} ];
ue=Flatten[ue]; Print["ue=",ue];
sige=Quad4IsoPRingStresses[ncoor,Emat,ue,{}];
Print["Corner stresses=",sige//MatrixForm];
```

FIGURE 4.10. Test statements and results for stress recovery module `Quad4IsoPRingStresses`.

Example 4.3. The stress recovery module is tested by the statements listed in Figure 4.10. The technique used is to generate the element node displacements by evaluating a test displacement field

$$u_r(r,z) = e_{rr} r, \qquad u_z(r,z) = e_{zz} r + \gamma_{rz} z \tag{4.24}$$

in which $\{e_{rr}, e_{zz}, \gamma_{rz}\}$ are specified strains assumed constant over the element. Note that the hoop strain is $e_{\theta\theta} = u_r/r = e_{rr}$. The geometry is that of the rectangular cross-section element of Figure 4.5.

```
Quad4IsoPRingStresses[ncoor_,Emat_,ue_,options_]:=
  Module[{numer=False,g=1/Sqrt[3],Jcons=False,w0=0,
     eps=10.^(-9),r1,r2,r3,r4,z1,z2,z3,z4,Nf,N1,N2,N3,N4,
     dNr1,dNr2,dNr3,dNr4,dNz1,dNz2,dNz3,dNz4,
     T1,T2,T3,T4,Td,Tg4,Jdet,qcoor,w,c,Be,
     gctab={{0,0}},k,kg,rk,sigg,sige,udis=ue,
     modname="Quad4IsoPRingStresses: "},
  If [Length[options]==1,{numer}=options];
  If [Length[options]==2,{numer,g}=options];
  If [Length[options]==3,{numer,g,w0}=options];
  If [Head[g]==Symbol||g>0, Td=4*g^2*(4+w0);
     T1=4*g^2*w0; T2=4+4*g^2+w0+2*g*(4+w0);
     T3=-4+4*g^2-w0; T4=4+4*g^2+w0-2*g*(4+w0);
     Tg4={{T1,T2,T3,T4,T3},{T1,T3,T2,T3,T4},
          {T1,T4,T3,T2,T3},{T1,T3,T4,T3,T2}}/Td;
     gctab={{0,0},{-1,-1},{1,-1},{1,1},{-1,1}}*g];
  kg=Length[gctab]; sigg=Table[{0,0,0,0},{kg}];
  If [numer, gctab=N[gctab]; Tg4=N[Tg4]; udis=N[ue] ];
  {{r1,z1},{r2,z2},{r3,z3},{r4,z4}}=ncoor;
  For [k=1,k<=kg,k++, qcoor=gctab[[k]];
      {Nf,{dNr1,dNr2,dNr3,dNr4},{dNz1,dNz2,dNz3,dNz4},
       Jdet}=Quad4IsoPRingShapeFunDer[ncoor,qcoor,Jcons];
      {N1,N2,N3,N4}=Nf; rk=r1*N1+r2*N2+r3*N3+r4*N4;
      Be={{ dNr1,   0, dNr2,   0, dNr3,   0, dNr4,   0},
          {    0,dNz1,    0, dNz2,   0,dNz3,    0,dNz4},
          {N1/rk,   0,N2/rk,   0,N3/rk,   0,N4/rk,   0},
          { dNz1,dNr1, dNz2,dNr2, dNz3,dNr3, dNz4,dNr4}};
      If [numer,Be=N[Be]]; sigg[[k]]=Emat.(Be.udis);
      ] ;
  If [kg==1, sige=Table[sigg[[1]],{4}], sige=Tg4.sigg];
  If [numer, sige=Chop[sige,eps]];
  If [!numer,sige=Simplify[sige]]; Return[sige] ];
```

FIGURE 4.9. Module for recovery of Quad4 ring element corner stresses from displacements.

The displacement field (4.24) is evaluated at the corner nodes to construct the node displacement vector, which is then fed to the stress recovery module. Dimensions a, b and d are kept arbitrary as symbolic variables. Numeric data: $e_{rr} = e_{\theta\theta} = 3/80$, $e_{zz} = -1/40$, $\gamma_{rz} = 4/50$, $E = 2500$ and $\nu = 1/4$. The associated displacement field as per (4.24) is

$$\mathbf{u}^e = \left[\ \frac{3d}{80}\quad \frac{2d}{25}\quad \frac{3(a+d)}{80}\quad \frac{2(a+d)}{25}\quad \frac{3(a+d)}{80}\quad \frac{b}{40}+\frac{2(a+d)}{25}\quad \frac{3d}{80}\quad -\frac{b}{40}+\frac{2d}{25}\ \right] \tag{4.25}$$

The computed corner stresses returned by the module (note that logical flag `numer` is `False` since that is the default) are

$$\boldsymbol{\sigma}^e = \begin{bmatrix} 200 & -50 & 200 & 80 \\ 200 & -50 & 200 & 80 \\ 200 & -50 & 200 & 80 \\ 200 & -50 & 200 & 80 \end{bmatrix} \tag{4.26}$$

which may be verify to be correct.

§4.3. The 8-Node Quadrilateral Ring Element

This is the axisymmetric solid version of the isoparametric quadrilateral with serendipity shape functions. The element has 8 nodes and 16 displacement degrees of freedom arranged as

$$\mathbf{u}^e = [\, u_{r1}\quad u_{z1}\quad u_{r2}\quad u_{z2}\quad u_{r3}\quad u_{z3}\quad \ldots\quad u_{r8}\quad u_{z8}\,]^T\,. \tag{4.27}$$

```
Quad8IsoPRingShapeFunDer[ncoor_,qcoor_,Jcons_]:=Module[
  {r1,r2,r3,r4,r5,r6,r7,r8,z1,z2,z3,z4,z5,z6,z7,z8,
   ξ,η,rv,zv,A0,dNξ,dNη,N1B,N2B,N3B,N4B,J11,J12,J21,J22,
   Nf,dNr,dNz,Jdet}, {ξ,η}=qcoor;
  {{r1,z1},{r2,z2},{r3,z3},{r4,z4},{r5,z5},{r6,z6},{r7,z7},
  {r8,z8}}=ncoor; A0=((r5-r7)*(z6-z8)-(r6-r8)*(z5-z7))/4;
   N1B=(1-ξ)*(1-η)/4;  N2B=(1+ξ)*(1-η)/4;
   N3B=(1+ξ)*(1+η)/4;  N4B=(1-ξ)*(1+η)/4;
   Nf= {-N1B*(1+ξ+η),-N2B*(1-ξ+η),-N3B*(1-ξ-η),
        -N4B*(1+ξ-η),  2*N1B*(1+ξ),2*N3B*(1-η),
         2*N3B*(1-ξ),  2*N4B*(1-η)};
   dNξ ={(1- η)*(2*ξ+η),(1-η)*(2*ξ-η),(1+η)*(2*ξ+η),
         (1+η)*(2*ξ-η),  4*ξ*(η-1),2*(1-η^2),
         -4*ξ*(1+η),-2*(1-η^2)}/4;
   dNη ={(1- ξ)*(ξ+2*η),-(1+ξ)*(ξ-2*η),(1+ξ)*(ξ+2*η),
         -(1-ξ)*(ξ-2*η),  -2*(1-ξ^2),-4*(1+ξ)*η,
          2*(1-ξ^2),-4*(1-ξ)*η}/4;
   rv={r1,r2,r3,r4,r5,r6,r7,r8}; zv={z1,z2,z3,z4,z5,z6,z7,z8};
   J11=dNξ.rv; J12=dNξ.zv; J21=dNη.rv; J22=dNη.zv;
   Jdet=Simplify[J11*J22-J12*J21]; If [Jcons,Jdet=A0];
   dNr=( J22*dNξ-J12*dNη)/Jdet;
   dNz=(-J21*dNξ+J11*dNη)/Jdet;
   Return[{Nf,dNr,dNz,Jdet}] ];
```

FIGURE 4.11. Shape function module for 8-node bilinear quadrilateral ring element.

§4.3.1. Shape Function Module

Module `Quad8IsoPRingShapeFunDer`, listed in Figure 4.11, computes the shape functions N_i^e, $i = 1, 2, \ldots 8$ and their partial derivatives with respect to r and z at a specified point in the element. Usually this module is called at sample points of a Gauss quadrature rule, but it may also be used with symbolic inputs to get information for an arbitrary point at $\{\xi, \eta\}$. The element geometry is defined by the 16 coordinates $\{r_i, z_i\}$, $i = 1, 2, \ldots 8$. These are collected in the arrays

$$\mathbf{r} = [\, r_1 \quad r_2 \quad \ldots \quad r_8 \,]^T, \qquad \mathbf{z} = [\, z_1 \quad z_2 \quad \ldots \quad r_8 \,]^T. \tag{4.28}$$

We will use the abbreviations $r_{ij} = r_i - r_j$ and $z_{ij} = z_i - z_j$ for coordinate differences The shape functions and their partial derivatives with respect to the quadrilateral coordinates are collected in the following arrays. Using the abbreviations $N_1^B = \frac{1}{4}(1-\xi)(1-\eta)$, $N_2^B = \frac{1}{4}(1+\xi)(1-\eta)$, $N_3^B = \frac{1}{4}(1+\xi)(1+\eta)$ and $N_4^B = \frac{1}{4}(1-\xi)(1+\eta)$ for the shape functions of the 4-noded bilinear quadrilateral, we have

$$\mathbf{N} = \begin{bmatrix} -N_1^B(1+\xi+\eta) \\ -N_2^B(1-\xi+\eta) \\ -N_3^B(1-\xi-\eta) \\ -N_4^B(1+\xi-\eta) \\ 2N_1^B(1+\xi) \\ 2N_2^B(1-\eta) \\ 2N_3^B(1-\xi) \\ 2N_4^B(1-\eta) \end{bmatrix}, \quad \mathbf{N}_{,\xi} = \tfrac{1}{4} \begin{bmatrix} (1-\eta)(2\xi+\eta) \\ (1-\eta)(2\xi-\eta) \\ (1+\eta)(2\xi+\eta) \\ (1+\eta)(2\xi-\eta) \\ 2\xi(\eta-1) \\ 2(1-\eta^2) \\ -2\xi(1+\eta) \\ -2(1-\eta^2) \end{bmatrix}, \quad \mathbf{N}_{,\eta} = \tfrac{1}{4} \begin{bmatrix} (1-\xi)(\xi+2\eta) \\ -(1+\xi)(\xi-2\eta) \\ (1+\xi)(\xi+2\eta) \\ -(1-\xi)(\xi-2\eta) \\ -2(1-\xi^2) \\ -2(1+\xi)\eta \\ 2(1-\xi^2) \\ -2(1-\xi)\eta \end{bmatrix}. \tag{4.29}$$

The Jacobian matrix is

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial r}{\partial \xi} & \dfrac{\partial z}{\partial \xi} \\ \dfrac{\partial r}{\partial \eta} & \dfrac{\partial z}{\partial \eta} \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{N}_{,\xi}\,\mathbf{r} & \mathbf{N}_{,\xi}\,\mathbf{z} \\ \mathbf{N}_{,\eta}\,\mathbf{r} & \mathbf{N}_{,\eta}\,\mathbf{z} \end{bmatrix}. \tag{4.30}$$

with determinant $J = \det(\mathbf{J}) = J_{11}\,J_{22} - J_{12}\,J_{21}$. Finally the $\{r, z\}$ partials are obtained from

$$\begin{bmatrix} \mathbf{N}_{,r} \\ \mathbf{N}_{,z} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \mathbf{N}}{\partial r} \\ \dfrac{\partial \mathbf{N}}{\partial z} \end{bmatrix} = \mathbf{J}^{-1} \begin{bmatrix} \mathbf{N}_{,\xi} \\ \mathbf{N}_{,\eta} \end{bmatrix} = \frac{1}{J} \begin{bmatrix} J_{22} & -J_{12} \\ -J_{21} & J_{11} \end{bmatrix} \begin{bmatrix} \mathbf{N}_{,\xi} \\ \mathbf{N}_{\eta} \end{bmatrix}. \tag{4.31}$$

or $\mathbf{N}_{,r} = (J_{11}\mathbf{N}_{,\xi} - J_{12}\mathbf{N}_{,\eta})/J$ and $\mathbf{N}_{,z} = (-J_{21}\mathbf{N}_{,\xi} + J_{12}\mathbf{N}_{,\eta})/J$. Unlike the 4-node quadrilateral, explicit expressions for $\mathbf{N}_{,r}$ and $\mathbf{N}_{,z}$ are difficult to work out because of the increased complexity of the polynomials. The module listed in Figure 4.11 does not attempt to do so. The module is invoked as

```
{ Nf,dNr,dNz,Jdet }= Quad4IsoPRingShapeFunDer[ncoor,qcoor,Jcons]      (4.32)
```

where the arguments are

- `ncoor` Quadrilateral node coordinates arranged in two-dimensional list form: `{ { r1,z1 },{ r2,z2 },{ r3,z3 }, ... { r8,z8 } }`.
- `qcoor` Quadrilateral coordinates { ξ , η } of the point at which shape functions and derivatives are to be evaluated.
- `Jcons` A logical flag. If `True`, the Jacobian determinant J is set to its value at the element center: $A_0 = (r_{57}z_{68} - r_{68}z_{57})/4$, for any $\{\xi, \eta\}$. That option is useful in certain research studies.

The module returns the list { `Nf,dNr,dNz,Jdet` }, where

- `Nf` Shape function values[2] arranged as the list { `N1,N2,N3, ... N8` }.
- `dNr` r shape function derivatives (4.31) arranged as the list { `dNr1,dNr2,dNr3, ... dNr8` }.
- `dNz` z shape function derivatives (4.31) arranged as the list { `dNz1,dNz2,dNz3, ... dNz8` }.
- `Jdet` Jacobian determinant.

§4.3.2. Element Stiffness Module

Module `Quad8IsoPRingStiffness`, listed in Figure 4.13, computes the stiffness matrix of a 8-noded iso-P quadrilateral ring element. The computation is carried out using numerical quadrature.

The module is invoked as

```
Ke = Quad8IsoPRingStiffness[ncoor,Emat,options]      (4.33)
```

[2] Note that `N` cannot be used as name of the list of shape function values, because that symbol is reserved.

```
Quad8IsoPRingStiffness[ncoor_,Emat_,options_]:=Module[
  {p=2,numer=False,Jcons=False,Kfac=1,qcoor,k,
   r1,r2,r3,r4,r5,r6,r7,r8,z1,z2,z3,z4,z5,z6,z7,z8,
   Nf,N1,N2,N3,N4,N5,N6,N7,N8,
   dNr1,dNr2,dNr3,dNr4,dNr5,dNr6,dNr7,dNr8,
   dNz1,dNz2,dNz3,dNz4,dNz5,dNz6,dNz7,dNz8,
   rk,w,c,A0,Jdet,Be,Ke,Ke0=Table[0,{16},{16}],
   modname="Quad8IsoPRingStiffness: "}, Ke=Ke0;
  If [Length[options]==1,{numer}=options];
  If [Length[options]==2,{numer,p}=options];
  If [Length[options]==3,{numer,p,Jcons}=options];
  If [Length[options]==4,{numer,p,Jcons,Kfac}=options];
  If [p<1||p>5, Print[modname,"illegal p:,p"]; Return[Ke0]];
  {{r1,z1},{r2,z2},{r3,z3},{r4,z4},
   {r5,z5},{r6,y6},{r7,z7},{r8,z8}}=ncoor;
  A0=((r5-r7)*(z6-z8)-(r6-r8)*(z5-z7))/4;
  If [numer&&(A0<=0), Print[modname,"Neg or zero area"];
      Return[Ke0]];
  For [k=1,k<=p*p,k++,
      {qcoor,w}= QuadGaussRuleInfo[{p,numer},k];
     {{N1,N2,N3,N4,N5,N6,N7,N8},
      {dNr1,dNr2,dNr3,dNr4,dNr5,dNr6,dNr7,dNr8},
      {dNz1,dNz2,dNz3,dNz4,dNz5,dNz6,dNz7,dNz8},
      Jdet}=Quad8IsoPRingShapeFunDer[ncoor,qcoor,Jcons];
      If [numer&&(Jdet<=0), Print[modname,"Neg or zero",
         " Gauss point Jacobian at k=",k]; Return[Ke0]];
      rk=r1*N1+r2*N2+r3*N3+r4*N4+r5*N5+r6*N6+r7*N7+r8*N8;
      Be={{ dNr1,   0, dNr2,   0, dNr3,   0, dNr4,   0,
            dNr5,   0, dNr6,   0, dNr7,   0, dNr8,   0},
          {    0,dNz1,    0,dNz2,    0,dNz3,    0,dNz4,
               0,dNz5,    0,dNz6,    0,dNz7,    0,dNz8},
          {N1/rk,   0,N2/rk,   0,N3/rk,   0,N4/rk,   0,
           N5/rk,   0,N6/rk,   0,N7/rk,   0,N8/rk,   0},
          { dNz1,dNr1, dNz2,dNr2, dNz3,dNr3, dNz4,dNr4,
            dNz5,dNr5, dNz6,dNr6, dNz7,dNr7, dNz8,dNr8}};
      c=Kfac*w*rk*Jdet; If[!numer, Be=Simplify[Be]];
      If [numer,Be=N[Be]; c=N[c]];
      Ke+=c*Transpose[Be].(Emat.Be);
      ]; Return[Ke] ];
```

FIGURE 4.12. Element stiffness formation module for 8-node iso-P quadrilateral ring.

The arguments are:

`ncoor`	Quadrilateral node coordinates arranged in two-dimensional list form: `{{r1,z1},{r2,z2},{r3,z3}, ... {r8,z8}}`.
`Emat`	Same as for `Quad4IsoPRingStiffness`.
`options`	Same as for `Quad4IsoPRingStiffness`.

As function value the module returns

`Ke`	a 16×16 symmetric matrix pertaining to a node by node arrangement of element node displacements. If an error is detected during processing, a zero matrix is returned.

Example 4.4. The stiffness module is tested on the geometry identified in Figure 4.5. The cross section is a rectangle dimensioned $a \times b$ with sides parallel to the $\{r, z\}$ axes. The distance of the leftmost side to the z

```
ClearAll[Em,n,a,b,d,p,numer];
a=4; b=2; d=0; Em=96; n=1/3;
ncoor={{d,0},{a+d,0},{a+d,b},{d,b},{a/2+d,0},{a+d,b/2},
       {a/2+d,b},{d,b/2}};
Emat=Em/((1+n)*(1-2*n))*{{1-n,n,n,0},{n,1-n,n,0},
                {n,n,1-n,0},{0,0,0,1/2-n}};
Print["Emat=",Emat//MatrixForm];
For [p=1,p<=4,p++,
     Ke=Quad8IsoPRingStiffness[ncoor,Emat,{True,p}];
     Ke=Simplify[Chop[Ke]];  Print["Ke=",Ke//MatrixForm];
     Print["Eigenvalues of Ke=",Chop[Eigenvalues[N[Ke]],.000001]];
     ];
```

FIGURE 4.13. Driver for exercising the `Quad8IsoPRingStiffness` module of Figure 4.13 using the ring element geometry shown in Figure 4.5, with $E = 96$, $\nu = 1/3$, $a = 4$, $b = 2$ $d = 0$ and four Gauss product integration rules.

axis is d. The material is isotropic with modulus E and Poisson's ratio ν.

The script of Figure 4.8 computes and prints the stiffness of the test element shown in for $E = 96$, $\nu = 1/3$, $a = 4$, $b = 2$ $d = 0$. The default `Kfac` = 1 is used. Nodes 1 and 2 sit on the z axes. The value of `p` is changed in a loop. The flag `numer` is set to `True` to use floating-point computation for speed. The computed entries of $\mathbf{K}^e$ are exact integers for all values of p:

The eigenvalues of these matrices are:

Rule	Eigenvalues of $\mathbf{K}^e$ for varying integration rule							
1×1	667.794	180.000	124.206	72.000	0	0	0	0
2×2	745.201	261.336	248.750	129.451	100.389	88.598	10.275	0
3×3	745.446	330.628	266.646	133.236	126.343	98.690	11.011	0
4×4	745.716	397.372	272.092	144.542	135.004	101.908	11.365	0

(4.34)

The stiffness matrix computed by the one-point rule is rank deficient by 11. For $p = 2$ it is rank deficient by one, but the element is useful[3] since the spurious mode is not usually propagated over the mesh. For $p = 3$ and higher the element has full rank of 15. The eigenvalues do not change appreciably after $p = 2$.

§4.3.3. Body Force Module

Module `Quad8IsoPRingBodyForces`, listed in Figure 4.14 computes the consistent force vector associated with a body force field $\vec{\mathbf{b}} = \{b_x, b_y\}$ specified over an 8-node iso-P quadrilateral ring element. The field is assumed to be given per unit of volume, in radial-axial component-wise form.

The force vector is computed by Gauss numerical integration as described in the previous chapter. The module is invoked as

```
Ke = Quad8IsoPRingBodyForces[ncoor,bfor,options]
```
(4.35)

The arguments are:

`ncoor` Same as in `Quad8IsoPRingStiffness`

[3] It will be seen in the benchmarks of Chapter 14 that the 8-node quadrilateral integrated with the 2×2 Gauss rue outperforms the fully integrated version, especially for near-incompressible material behavior.

`bfor` Specifies body force field (forces per unit of volume) over the element. Three specification formats are allowed.

One-dimensional list: `{ br,bz }`

Two-dimensional list with corner values only: `{ { br1,bz1 },{ br2,bz2 }, ... { br4,bz4 } }`

Two-dimensional list with values at corners and midnodes: `{ { br1,bz1 }, ... { br8,bz8 } }`

In the first form the body force field is taken to be uniform over the element, with radial component `br` and axial component `bz`.

The second and third forms assume body forces to vary over the element. If only the corner values are given, the value at midnodes is determined from the adjacent corner nodes by averaging. From this information the field is interpolated over the element using the 8-node shape functions.

`options` Same as in `Quad8IsoPRingStiffness`

As function value the module returns

`fe` Consistent force vector arranged `{ fr1,fz1,fr2,fz2,fr3,fz3,fr4,fz4 }` to represent

$$\mathbf{f}^e = [\ f_{r1} \quad f_{z1} \quad f_{r2} \quad f_{z2} \quad f_{r3} \quad f_{z3} \quad \ldots \quad f_{r8} \quad f_{z8}\ ]^T. \tag{4.36}$$

Example 4.5. To be added later.

```
Quad8IsoPRingBodyForces[ncoor_,bfor_,options_]:=Module[
  {p=2,numer=False,Jcons=False,Kfac=1,qcoor,k,m,mOK,
   r1,r2,r3,r4,r5,r6,r7,r8,z1,z2,z3,z4,z5,z6,z7,z8,
   Nf,N1,N2,N3,N4,N5,N6,N7,N8,dNr,dNz,
   br1,br2,br3,br4,br5,br6,br7,br8,
   bz1,bz2,bz3,bz4,bz5,bz6,bz7,bz8,
   rk,w,c,A0,Jdet,fe,fe0=Table[0,{16}],
   modname="Quad8IsoPRingBodyForces: "}, fe=fe0;
  If [Length[options]==1,{numer}=options];
  If [Length[options]==2,{numer,p}=options];
  If [Length[options]==3,{numer,p,Jcons}=options];
  If [Length[options]==4,{numer,p,Jcons,Kfac}=options];
  If [p<1||p>5, Print[modname,"illegal p:",p]; Return[fe0]];
  {{r1,z1},{r2,z2},{r3,z3},{r4,z4},
   {r5,z5},{r6,z6},{r7,z7},{r8,z8}}=ncoor;
  A0=((r5-r7)*(z6-z8)-(r6-r8)*(z5-z7))/4;
  If [numer&&(A0<=0), Print[modname,"Neg or zero area"];
      Return[fe0]]; m=Length[bfor]; mOK=MemberQ[{2,4,8},m];
  If [!mOK, Print[modname," Illegal bfor"]; Return[fe0]];
  If [m==2, br1=br2=br3=br4=br5=br6=br7=br8=bfor[[1]];
            bz1=bz2=bz3=bz4=bz5=bz6=bz7=bz8=bfor[[2]]];
  If [m==4,{{br1,bz1},{br2,bz2},{br3,bz3},{br4,bz4}}=bfor;
            {br5,bz5,br6,bz6,br7,bz7,br8,bz8}={br1+br2,
             bz1+bz2,br2+br3,bz2+bz3,br3+br4,bz3+bz4,
             br4+br1,bz4+bz1}/2];
  If [m==8,{{br1,bz1},{br2,bz2},{br3,bz3},{br4,bz4},
            {br5,bz5},{br6,bz6},{br7,bz7},{br8,bz8}}=bfor];
  For [k=1,k<=p*p,k++,
      {qcoor,w}= QuadGaussRuleInfo[{p,numer},k];
     {{N1,N2,N3,N4,N5,N6,N7,N8},dNr,dNz,Jdet}=
           Quad8IsoPRingShapeFunDer[ncoor,qcoor,Jcons];
      If [numer&&(Jdet<=0), Print[modname,"Neg or zero",
         " Gauss point Jacobian at k=",k]; Return[fe0]];
      rk=r1*N1+r2*N2+r3*N3+r4*N4+r5*N5+r6*N6+r7*N7+r8*N8;
      brk=br1*N1+br2*N2+br3*N3+br4*N4+br5*N5+br6*N6+br7*N7+br8*N8;
      bzk=bz1*N1+bz2*N2+bz3*N3+bz4*N4+bz5*N5+bz6*N6+bz7*N7+bz8*N8;
      bk={N1*brk,N1*bzk,N2*brk,N2*bzk,N3*brk,N3*bzk,N4*brk,N4*bzk,
          N5*brk,N5*bzk,N6*brk,N6*bzk,N7*brk,N7*bzk,N8*brk,N8*bzk};
      c=Kfac*w*Jdet*rk; If [numer,bk=N[bk];c=N[c]]; fe+=c*bk;
      ];  Return[fe] ];
```

FIGURE 4.14. Module that computes consistent node forces for a 8-noded quadrilateral ring element given a body force field.

§4.3.4. Stress Recovery Module

Module `Quad8IsoPRingStresses`, listed in Figure 4.16, recovers stresses at the 4 corner nodes and 4 midpoints of the iso-P 8-node quadrilateral ring element, given its node displacements.

The procedure is similar to that used for the 4-node quadrilateral explained in §4.2.4. The stresses are recovered at five sample points $k = 0, 1, 2, 3, 4$ with quadrilateral coordinates $\{\xi, \eta\} = \{0, 0\}$, $\{-g, -g\}$, $\{g, -g\}$, $\{g, g\}$, $\{-g, g\}$, in which $0 < g \leq 1$, using the direct evaluation $\bar{\boldsymbol{\sigma}}_k = \mathbf{E}\,\mathbf{B}_k^e\,\mathbf{u}^e$. (A bar over the stress symbol is used to mark a sample value.) Perform a least-square bilinear fit over the 5 sample points assigning weight $0 \leq w_0$ to sample at $\{\xi, \eta\} = \{0, 0\}$ and weight 1 to each

```
ClearAll[a,b,d,p];
a=3; b=2; d=1;
ncoor={{d,0},{a+d,0},{a+d,b},{d,b},{a/2+d,0},{a+d,b/2},{a/2+d,b},
       {d,b/2}};
For [p=1,p<=3,p++, For [case=1,case<=2,case++,
     If [case==1, bfor={36,-18}];
     If [case==2, bfor=Table[{60*ncoor[[i,1]],0},{i,8}]];
     fe=Quad8IsoPRingBodyForces[ncoor,bfor,{True,p}];
     fe=Simplify[Chop[fe]];
     Print["fe=",Transpose[Partition[fe,2]]//MatrixForm];
     frsum=Sum[fe[[2*i-1]],{i,8}]; fzsum=Sum[fe[[2*i]],{i,8}];
     Print["frsum=",frsum," fzsum=",fzsum];
     ]];
```

FIGURE 4.15. Test statements to exercise body force module of Figure 4.7.

of the samples at $\{\xi,\eta\}=\{\pm g,\pm g\}$. Evaluation of the fit at the corner and midpoint nodes yields

$$\begin{bmatrix} \sigma_{rr1} & \sigma_{zz1} & \sigma_{\theta\theta 1} & \sigma_{rz1} \\ \sigma_{rr2} & \sigma_{zz2} & \sigma_{\theta\theta 2} & \sigma_{rz2} \\ \sigma_{rr3} & \sigma_{zz3} & \sigma_{\theta\theta 3} & \sigma_{rz3} \\ \sigma_{rr4} & \sigma_{zz4} & \sigma_{\theta\theta 4} & \sigma_{rz4} \\ \sigma_{rr5} & \sigma_{zz5} & \sigma_{\theta\theta 5} & \sigma_{rz5} \\ \sigma_{rr6} & \sigma_{zz6} & \sigma_{\theta\theta 6} & \sigma_{rz6} \\ \sigma_{rr7} & \sigma_{zz7} & \sigma_{\theta\theta 7} & \sigma_{rz7} \\ \sigma_{rr8} & \sigma_{zz8} & \sigma_{\theta\theta 8} & \sigma_{rz8} \end{bmatrix} = \frac{1}{T_d} \begin{bmatrix} T_1 & T_2 & T_3 & T_4 & T_3 \\ T_1 & T_3 & T_2 & T_3 & T_4 \\ T_1 & T_4 & T_3 & T_2 & T_3 \\ T_1 & T_3 & T_4 & T_3 & T_2 \\ T_1 & T_5 & T_5 & T_6 & T_6 \\ T_1 & T_6 & T_5 & T_5 & T_6 \\ T_1 & T_6 & T_6 & T_5 & T_5 \\ T_1 & T_5 & T_6 & T_6 & T_5 \end{bmatrix} \begin{bmatrix} \bar\sigma_{rr0} & \bar\sigma_{zz0} & \bar\sigma_{\theta\theta 0} & \bar\sigma_{rz0} \\ \bar\sigma_{rr1} & \bar\sigma_{zz1} & \bar\sigma_{\theta\theta 1} & \bar\sigma_{rz1} \\ \bar\sigma_{rr2} & \bar\sigma_{zz2} & \bar\sigma_{\theta\theta 2} & \bar\sigma_{rz2} \\ \bar\sigma_{rr3} & \bar\sigma_{zz3} & \bar\sigma_{\theta\theta 3} & \bar\sigma_{rz3} \\ \bar\sigma_{rr4} & \bar\sigma_{zz4} & \bar\sigma_{\theta\theta 4} & \bar\sigma_{rz4} \end{bmatrix}, \qquad (4.37)$$

in which T_1 T_2, T_3, T_4 and T_d are the same as in the 4-node quadrilateral module whereas $T_5 = g(4+4g+w_0)$ and $T_6 = g(-4+4g-w_0)$. The default values used in the least-square fit are $w_0 = 0$ and $g = 1/\sqrt{3}$, in which case $\{\xi,\eta\}=\{\pm g,\pm g\}$ are located at the sample points of the 2×2 Gauss product rule.

```
ClearAll[Em,ν,a,b,d,err,ezz,grz,ur,uz,r,z];
Em=2500; ν=1/4; d=1; a=3; b=2;
{err,ezz,err,grz}={3/80,-1/40,3/80,4/50};
ncoor={{d,0},{a+d,0},{a+d,b},{d,b},{a/2+d,0},{a+d,b/2},{a/2+d,b},
       {d,b/2}};
Emat=Em/((1+ν)*(1-2*ν))*{{1+ν,ν,ν,0},{ν,1+ν,ν,0},
         {ν,ν,1+ν,0},{0,0,0,1/2-ν}};
{err,ezz,err,grz}={3/80,-1/40,3/80,4/50};
ur[r_,z_]:=err*r; uz[r_,z_]:=ezz*z+grz*r;
ue=Table[{0,0},{8}];
For [n=1,n<=8,n++, {rn,zn}=ncoor[[n]];
     ue[[n]]={ur[rn,zn],uz[rn,zn]} ];
ue=Flatten[ue]; Print["ue=",ue];
sige=Quad8IsoPRingStresses[ncoor,Emat,ue,{True}];
Print["Corner stresses=",sige//MatrixForm];
```

FIGURE 4.17. Test statements for stress recovery module `Quad8IsoPRingStresses`.

The module is invoked as

```
Ke = Quad8IsoPRingStresses[ncoor,Emat,ue,options]          (4.38)
```

```
Quad8IsoPRingStresses[ncoor_,Emat_,ue_,options_]:=
  Module[{numer=False,g=1/Sqrt[3],Jcons=False,w0=0,
   eps=10.^(-9),r1,r2,r3,r4,r5,r6,r7,r8,z1,z2,z3,z4,
   z5,z6,z7,z8,Nf,N1,N2,N3,N4,N5,N6,N7,N8,
   dNr1,dNr2,dNr3,dNr4,dNr5,dNr6,dNr7,dNr8,
   dNz1,dNz2,dNz3,dNz4,dNz5,dNz6,dNz7,dNz8,
   T1,T2,T3,T4,T5,T6,Td,Tg8,Jdet,qcoor,w,c,Be,
   gctab={{0,0}},k,kg,rk,sigg,sige,udis=ue,
   modname="Quad8IsoPRingStresses: "},
  If [Length[options]==1,{numer}=options];
  If [Length[options]==2,{numer,g}=options];
  If [Length[options]==3,{numer,g,w0}=options];
  If [Head[g]==Symbol||g>0, Td=4*g^2*(4+w0);
     T1=4*g^2*w0; T2=4+4*g^2+w0+2*g*(4+w0);
     T3=-4+4*g^2-w0; T4=4+4*g^2+w0-2*g*(4+w0);
     T5=g*(4+4*g+w0); T6=g*(-4+4*g-w0);
     Tg8={{T1,T2,T3,T4,T3},{T1,T3,T2,T3,T4},
          {T1,T4,T3,T2,T3},{T1,T3,T4,T3,T2},
          {T1,T5,T5,T6,T6},{T1,T6,T5,T5,T6},
          {T1,T6,T6,T5,T5},{T1,T5,T6,T6,T5}}/Td;
     gctab={{0,0},{-1,-1},{1,-1},{1,1},{-1,1}}*g];
  kg=Length[gctab]; sigg=Table[{0,0,0,0},{kg}];
  If [numer, gctab=N[gctab]; Tg8=N[Tg8]; udis=N[ue]];
  {{r1,z1},{r2,z2},{r3,z3},{r4,z4},
   {r5,z5},{r6,z6},{r7,z7},{r8,z8}}=ncoor;
  For [k=1,k<=kg,k++, qcoor=gctab[[k]];
     {{N1,N2,N3,N4,N5,N6,N7,N8},
      {dNr1,dNr2,dNr3,dNr4,dNr5,dNr6,dNr7,dNr8},
      {dNz1,dNz2,dNz3,dNz4,dNz5,dNz6,dNz7,dNz8},
      Jdet}=Quad8IsoPRingShapeFunDer[ncoor,qcoor,Jcons];
      rk=r1*N1+r2*N2+r3*N3+r4*N4+r5*N5+r6*N6+r7*N7+r8*N8;
      Be={{ dNr1,   0, dNr2,   0, dNr3,   0, dNr4,   0,
            dNr5,   0, dNr6,   0, dNr7,   0, dNr8,   0},
          {    0,dNz1,    0,dNz2,    0,dNz3,    0,dNz4,
               0,dNz5,    0,dNz6,    0,dNz7,    0,dNz8},
          {N1/rk,   0,N2/rk,   0,N3/rk,   0,N4/rk,   0,
           N5/rk,   0,N6/rk,   0,N7/rk,   0,N8/rk,   0},
          { dNz1,dNr1, dNz2,dNr2, dNz3,dNr3, dNz4,dNr4,
            dNz5,dNr5, dNz6,dNr6, dNz7,dNr7, dNz8,dNr8}};
      If [numer,Be=N[Be]]; sigg[[k]]=Emat.(Be.udis)
      ] ;
  If [kg==1, sige=Table[sigg[[1]],{4}], sige=Tg8.sigg];
  If [numer, sige=Chop[sige,eps]];
  If [!numer,sige=Simplify[sige]]; Return[sige] ];
```

FIGURE 4.16. Module for recovery of Quad8 ring element corner stresses from displacements.

The arguments are:

`ncoor` Node coordinates: same as in `Quad8IsoPRingStiffness`

`Emat` Elasticity matrix: same as in `Quad8IsoPRingStiffness`

`ue` The element node displacements arranged as a one-dimensional list: { `ur1,uz1, ur2,uz2,ur3,uz3, ... ur8,uz8` } representing the displacement vector

$$\mathbf{u}^e = [\, u_{r1} \quad u_{z1} \quad u_{r2} \quad u_{z2} \quad u_{r3} \quad u_{z3} \quad \ldots \quad u_{r8} \quad u_{z8} \,]^T. \qquad (4.39)$$

`options` Same as in `Quad4IsoPRingStresses`. The same defaults for omitted values apply.

As function value the module returns

sige — computed corner stresses stored in a 8-entry, two-dimensional list: `{{sigrr1,sigzz1,sigtt1,sigrz1}, {sigrr2,sigzz2,sigtt2,sigrz2}, ... {sigrr8,sigzz8,sigtt8,sigrz8}}` to represent the array shown on the left hand side of (4.37).

Example 4.6. To be added later.

Homework Exercises for Chapter 4

4- and 8-Node Iso-P Quadrilateral Ring Elements

No Exercises constructed for this Chapter yet. The elements are used in Exercises of following chapters.