ASEN 5007-Homework 10

Zach Dischner

Helpful Modules

Helpful modules

Cell 7: Simple function to print output for solutions in a stylazed way

```
In[1]:= PrintWithStyle[x_] :=
         Module[{color = LightGreen}, Framed[Style[x, 18, Bold, Background → color],
           Background → color]
         ]
```

```
Quad4IsoPMembraneStiffness[ncoor_,Emat_,th_,options_]:=
  Module[{i,k,p=2,numer=False,h=th,qcoor,c,w,Nf,
    dNx,dNy,Jdet,Be,Ke=Table[0,{8},{8}]},
  If [Length[options]==2, {numer,p}=options, {numer}=options];
  If [p<1||p>4, Print["p out of range"]; Return[Null]];
  For [k=1, k<=p*p, k++,
       {qcoor,w}= QuadGaussRuleInfo[{p,numer},k];
       {Nf,dNx,dNy,Jdet}=Quad4IsoPShapeFunDer[ncoor,qcoor];
        If [Length[th]==4, h=th.Nf]; c=w*Jdet*h;
        Be={Flatten[Table[{dNx[[i]],       0},{i,4}]],
            Flatten[Table[{0,       dNy[[i]]},{i,4}]],
            Flatten[Table[{dNy[[i]],dNx[[i]]},{i,4}]]};
        Ke+=Simplify[c*Transpose[Be].(Emat.Be)];
      ]; Return[Simplify[Ke]]
   ];

Quad4IsoPMembraneBodyForces[ncoor_,rho_,th_,options_,bfor_]:=
  Module[{i,k,p=2,numer=False,h=th,
    bx,by,bx1,by1,bx2,by2,bx3,by3,bx4,by4,bxc,byc,qcoor,
    c,w,Nf,dNx,dNy,Jdet,B,qctab,fe=Table[0,{8}]},
  If [Length[options]==2, {numer,p}=options, {numer}=options];
  If [Length[bfor]==2,{bx,by}=bfor;bx1=bx2=bx3=bx4=bx;by1=by2=by3=by4=by];
  If [Length[bfor]==4,{{bx1,by1},{bx2,by2},{bx3,by3},{bx4,by4}}=bfor];
  If [p<1||p>4, Print["p out of range"]; Return[Null]];
  bxc={bx1,bx2,bx3,bx4}; byc={by1,by2,by3,by4};
  For [k=1, k<=p*p, k++,
       {qcoor,w}= QuadGaussRuleInfo[{p,numer},k];
       {Nf,dNx,dNy,Jdet}=Quad4IsoPShapeFunDer[ncoor,qcoor];
       bx=Nf.bxc; by=Nf.byc; If [Length[th]==4, h=th.Nf];
       c=w*Jdet*h;
       bk=Flatten[Table[{Nf[[i]]*bx,Nf[[i]]*by},{i,4}]];
       fe+=c*bk;
      ]; Return[fe]
   ];

Quad4IsoPMembraneStresses[ncoor_,Emat_,th_,options_,udis_]:=
  Module[{i,k,numer=False,qcoor,Nf,
    dNx,dNy,Jdet,Be,qctab,ue=udis,sige=Table[0,{4},{3}]},
  qctab={{-1,-1},{1,-1},{1,1},{-1,1}};
  numer=options[[1]];
  If [Length[udis]==4, ue=Flatten[udis]];
  For [k=1, k<=Length[sige], k++,
       qcoor=qctab[[k]]; If [numer, qcoor=N[qcoor]];
       {Nf,dNx,dNy,Jdet}=Quad4IsoPShapeFunDer[ncoor,qcoor];
        Be={ Flatten[Table[{dNx[[i]],       0},{i,4}]],
            Flatten[Table[{0,       dNy[[i]]},{i,4}]],
            Flatten[Table[{dNy[[i]],dNx[[i]]},{i,4}]]};
        sige[[k]]=Emat.(Be.ue);
      ]; Return[sige]
   ];

Quad4IsoPShapeFunDer[ncoor_,qcoor_]:= Module[
  {Nf,dNx,dNy,dNξ,dNη,i,J11,J12,J21,J22,Jdet,ξ,η,x,y,
  x1,x2,x3,x4,y1,y2,y3,y4},
  {ξ,η}=qcoor;  {{x1,y1},{x2,y2},{x3,y3},{x4,y4}}=ncoor;
   Nf={(1-ξ)*(1-η),(1+ξ)*(1-η),(1+ξ)*(1+η),(1-ξ)*(1+η)}/4;
   dNξ ={-(1-η), (1-η),(1+η),-(1+η)}/4;
   dNη= {-(1-ξ),-(1+ξ),(1+ξ), (1-ξ)}/4;
   x={x1,x2,x3,x4}; y={y1,y2,y3,y4};
   J11=dNξ.x; J12=dNξ.y; J21=dNη.x; J22=dNη.y;
   Jdet=Simplify[J11*J22-J12*J21];
   dNx= ( J22*dNξ-J12*dNη)/Jdet;  dNx=Simplify[dNx];
   dNy= (-J21*dNξ+J11*dNη)/Jdet;  dNy=Simplify[dNy];
   Return[{Nf,dNx,dNy,Jdet}]
];

QuadGaussRuleInfo[{rule_,numer_},point_]:= Module[
 {ξ,η,p1,p2,i,j,w1,w2,m,info={{Null,Null},0}},
  If [Length[rule]==2,  {p1,p2}=rule, p1=p2=rule];
  If [Length[point]==2, {i,j}=point, m=point;
      j=Floor[(m-1)/p1]+1; i=m-p1*(j-1) ];
  {ξ,w1}=  LineGaussRuleInfo[{p1,numer},i];
  {η,w2}=  LineGaussRuleInfo[{p2,numer},j];
  info={{ξ,η},w1*w2};
  If [numer, Return[N[info]], Return[Simplify[info]]];
];
```

Out[2]=

```
Quad4IsoPMembraneStiffness[ncoor_,Emat_,th_,options_]:=
  Module[{i,k,p=2,numer=False,h=th,qcoor,c,w,Nf,
    dNx,dNy,Jdet,Be,Ke=Table[0,{8},{8}]},
  If [Length[options]==2, {numer,p}=options, {numer}=options];
  If [p<1||p>4, Print["p out of range"]; Return[Null]];
  For [k=1, k<=p*p, k++,
       {qcoor,w}= QuadGaussRuleInfo[{p,numer},k];
       {Nf,dNx,dNy,Jdet}=Quad4IsoPShapeFunDer[ncoor,qcoor];
        If [Length[th]==4, h=th.Nf]; c=w*Jdet*h;
        Be={Flatten[Table[{dNx[[i]],       0},{i,4}]],
            Flatten[Table[{0,       dNy[[i]]},{i,4}]],
            Flatten[Table[{dNy[[i]],dNx[[i]]},{i,4}]]};
        Ke+=Simplify[c*Transpose[Be].(Emat.Be)];
      ]; Return[Simplify[Ke]]
   ];

Quad4IsoPMembraneBodyForces[ncoor_,rho_,th_,options_,bfor_]:=
  Module[{i,k,p=2,numer=False,h=th,
    bx,by,bx1,by1,bx2,by2,bx3,by3,bx4,by4,bxc,byc,qcoor,
    c,w,Nf,dNx,dNy,Jdet,B,qctab,fe=Table[0,{8}]},
  If [Length[options]==2, {numer,p}=options, {numer}=options];
  If [Length[bfor]==2,{bx,by}=bfor;bx1=bx2=bx3=bx4=bx;by1=by2=by3=by4=by];
  If [Length[bfor]==4,{{bx1,by1},{bx2,by2},{bx3,by3},{bx4,by4}}=bfor];
  If [p<1||p>4, Print["p out of range"]; Return[Null]];
  bxc={bx1,bx2,bx3,bx4}; byc={by1,by2,by3,by4};
  For [k=1, k<=p*p, k++,
       {qcoor,w}= QuadGaussRuleInfo[{p,numer},k];
       {Nf,dNx,dNy,Jdet}=Quad4IsoPShapeFunDer[ncoor,qcoor];
       bx=Nf.bxc; by=Nf.byc; If [Length[th]==4, h=th.Nf];
       c=w*Jdet*h;
       bk=Flatten[Table[{Nf[[i]]*bx,Nf[[i]]*by},{i,4}]];
       fe+=c*bk;
      ]; Return[fe]
   ];

Quad4IsoPMembraneStresses[ncoor_,Emat_,th_,options_,udis_]:=
  Module[{i,k,numer=False,qcoor,Nf,
    dNx,dNy,Jdet,Be,qctab,ue=udis,sige=Table[0,{4},{3}]},
  qctab={{-1,-1},{1,-1},{1,1},{-1,1}};
  numer=options[[1]];
  If [Length[udis]==4, ue=Flatten[udis]];
  For [k=1, k<=Length[sige], k++,
       qcoor=qctab[[k]]; If [numer, qcoor=N[qcoor]];
       {Nf,dNx,dNy,Jdet}=Quad4IsoPShapeFunDer[ncoor,qcoor];
        Be={ Flatten[Table[{dNx[[i]],       0},{i,4}]],
            Flatten[Table[{0,       dNy[[i]]},{i,4}]],
            Flatten[Table[{dNy[[i]],dNx[[i]]},{i,4}]]};
        sige[[k]]=Emat.(Be.ue);
      ]; Return[sige]
   ];

Quad4IsoPShapeFunDer[ncoor_,qcoor_]:= Module[
  {Nf,dNx,dNy,dNξ,dNη,i,J11,J12,J21,J22,Jdet,ξ,η,x,y,
  x1,x2,x3,x4,y1,y2,y3,y4},
  {ξ,η}=qcoor;  {{x1,y1},{x2,y2},{x3,y3},{x4,y4}}=ncoor;
   Nf={(1-ξ)*(1-η),(1+ξ)*(1-η),(1+ξ)*(1+η),(1-ξ)*(1+η)}/4;
   dNξ ={-(1-η), (1-η),(1+η),-(1+η)}/4;
```

```
    dNη= {-(1-ξ),-(1+ξ),(1+ξ), (1-ξ)}/4;
    x={x1,x2,x3,x4}; y={y1,y2,y3,y4};
    J11=dNξ.x; J12=dNξ.y; J21=dNη.x; J22=dNη.y;
    Jdet=Simplify[J11*J22-J12*J21];
    dNx= ( J22*dNξ-J12*dNη)/Jdet;  dNx=Simplify[dNx];
    dNy= (-J21*dNξ+J11*dNη)/Jdet;  dNy=Simplify[dNy];
    Return[{Nf,dNx,dNy,Jdet}]
];

QuadGaussRuleInfo[{rule_,numer_},point_]:= Module[
 {ξ,η,p1,p2,i,j,w1,w2,m,info={{Null,Null},0}},
  If [Length[rule]==2,  {p1,p2}=rule, p1=p2=rule];
  If [Length[point]==2, {i,j}=point, m=point;
      j=Floor[(m-1)/p1]+1; i=m-p1*(j-1) ];
  {ξ,w1}=  LineGaussRuleInfo[{p1,numer},i];
  {η,w2}=  LineGaussRuleInfo[{p2,numer},j];
  info={{ξ,η},w1*w2};
  If [numer, Return[N[info]], Return[Simplify[info]]];
];


LineGaussRuleInfo[{rule_,numer_},point_]:= Module[
  {g2={-1,1}/Sqrt[3],w3={5/9,8/9,5/9},
   g3={-Sqrt[3/5],0,Sqrt[3/5]},
   w4={(1/2)-Sqrt[5/6]/6, (1/2)+Sqrt[5/6]/6,
       (1/2)+Sqrt[5/6]/6, (1/2)-Sqrt[5/6]/6},
   g4={-Sqrt[(3+2*Sqrt[6/5])/7],-Sqrt[(3-2*Sqrt[6/5])/7],
        Sqrt[(3-2*Sqrt[6/5])/7], Sqrt[(3+2*Sqrt[6/5])/7]},
   g5={-Sqrt[5+2*Sqrt[10/7]],-Sqrt[5-2*Sqrt[10/7]],0,
        Sqrt[5-2*Sqrt[10/7]], Sqrt[5+2*Sqrt[10/7]]}/3,
   w5={322-13*Sqrt[70],322+13*Sqrt[70],512,
       322+13*Sqrt[70],322-13*Sqrt[70]}/900,
   i=point,p=rule,info={{Null,Null},0}},
  If [p==1, info={0,2}];
  If [p==2, info={g2[[i]],1}];
  If [p==3, info={g3[[i]],w3[[i]]}];
  If [p==4, info={g4[[i]],w4[[i]]}];
  If [p==5, info={g5[[i]],w5[[i]]}];
  If [numer, Return[N[info]], Return[Simplify[info]]];
];
```

Problem 1 - Book Exercise 23.7

```
In[3]:= Quad8IsoPMembraneStiffness[ncoor_, Emat_, th_, options_] :=
         Module[{i, k, p = 2, numer = False, h = th, qcoor, c, w, Nf,
          dNx, dNy, Jdet, Be, Ke = Table[0, {16}, {16}]},
         If [Length[options] == 2, {numer, p} = options, {numer} = options];
         If [p < 1 || p > 4, Print["p out of range"]; Return[Null]];
         For [k = 1, k <= p * p, k++,
           {qcoor, w} = QuadGaussRuleInfo[{p, numer}, k];
           {Nf, dNx, dNy, Jdet} = Quad8IsoPShapeFunDer[ncoor, qcoor];
            If [Length[th] == 4, h = th.Nf]; c = w * Jdet * h;
            Be = {Flatten[Table[{dNx[[i]],    0}, {i, 8}]],
              Flatten[Table[{0,    dNy[[i]]}, {i, 8}]],
              Flatten[Table[{dNy[[i]], dNx[[i]]}, {i, 8}]]};
            Ke += Simplify[c * Transpose[Be].(Emat.Be)];
          ]; Return[Simplify[Ke]]
         ];
```

```
In[4]:= Quad8IsoPShapeFunDer[ncoor_, qcoor_] := Module[
         {Nf, dNx, dNy, dNξ, dNη, i, J11, J12, J21, J22, Jdet, ξ, η, x, y,
         x1, x2, x3, x4, x5, x6, x7, x8, y1, y2, y3, y4, y5, y6, y7, y8},
         {ξ, η} = qcoor;
           {{x1, y1}, {x2, y2}, {x3, y3},
              {x4, y4}, {x5, y5}, {x6, y6}, {x7, y7}, {x8, y8}} = ncoor;
          Nf = {(ξ - 1) * (η - 1) * (1 + ξ + η) / 4,
              (-ξ - 1) * (η - 1) * (1 - ξ + η) 4,
              (-ξ - 1) * (-η - 1) * (1 - ξ - η) / 4,
              (ξ - 1) * (-η - 1) * (1 + ξ - η) / 4,
              (1 - ξ^2) * (1 - η) / 2,
              (1 - η^2) * (1 + ξ) / 2,
              (1 - ξ^2) * (1 + η) / 2,
              (1 - η^2) * (1 - ξ) / 2
            };
          (*dNξ ={-(1-η)/4,  (1-η)/4,(1+η)/4,-(1+η)/4
            };
           dNη= {-(1-ξ),-(1+ξ),(1+ξ), (1-ξ)}/4;*)
           dNξ = D[Nf, ξ];
           dNη = D[Nf, η];
          x = {x1, x2, x3, x4, x5, x6, x7, x8};
            y = {y1, y2, y3, y4, y5, y6, y7, y8};
          J11 = dNξ.x; J12 = dNξ.y; J21 = dNη.x; J22 = dNη.y;
          Jdet = Simplify[J11 * J22 - J12 * J21];
          dNx = ( J22 * dNξ - J12 * dNη) / Jdet; dNx = Simplify[dNx];
          dNy = (-J21 * dNξ + J11 * dNη) / Jdet; dNy = Simplify[dNy];
          Return[{Nf, dNx, dNy, Jdet}]
        ];
        QuadGaussRuleInfo[{rule_, numer_}, point_] :=
          Module[{ξ, η, p1, p2, i, j, w1, w2, m, info = {{Null, Null}, 0}},
           If[Length[rule] == 2, {p1, p2} = rule, p1 = p2 = rule];
           If[Length[point] == 2, {i, j} = point, m = point;
            j = Floor[(m - 1) / p1] + 1; i = m - p1 * (j - 1)];
           {ξ, w1} = LineGaussRuleInfo[{p1, numer}, i];
           {η, w2} = LineGaussRuleInfo[{p2, numer}, j];
           info = {{ξ, η}, w1 * w2};
           If[numer, Return[N[info]], Return[Simplify[info]]];];

In[8]:= ClearAll[Em, nu, h, a, p]; h = 1 / 3;
        Emat = {{17 837 820, 5 945 940, 0}, {5 945 940, 17 837 820, 0}, {0, 0, 5 945 940}};
        ncoor = {{0, 0}, {2 * a, 0}, {2 * a, a}, {0, a}, {a, 0}, {2 * a, a / 2}, {a, a}, {0, a / 2}};
        PrintWithStyle[
         "First, I derived the shape functions for the rest of the element, based on what
           was provided in Chapter 18. Those are included in the module above.
           Due to symmetry and transitivity, I was able to do so by inspection."]
        PrintWithStyle["There is an error somewhere here. I don't know what it is.
           The code I made is close though!!! :-("]
        For[p = 1, p ≤ 4, p++, Ke = Quad8IsoPMembraneStiffness[ncoor, Emat, h, {True, p}];
         Ke = Rationalize[Ke, 0.0000001]; Print["Ke=", Ke // MatrixForm];
        ]
```

Out[11]=

```
First, I derived the shape functions for the rest of the
  element, based on what was provided in Chapter 18.
  Those are included in the module above. Due to symmetry
  and transitivity, I was able to do so by inspection.
```

Out[12]=

```
There is an error somewhere here. I don't know what
  it is. The code I made is close though!!! :-(
```

```
Set::shape : Lists {ξ$792, w1$792} and LineGaussRuleInfo[{1, True}, 1] are not the same shape. ≫

Set::shape : Lists {η$792, w2$792} and LineGaussRuleInfo[{1, True}, 1] are not the same shape. ≫

      495 495 w1$792 w2$792 (5824 η$792^6+8 η$792^5 (-1307+276 ξ$792+1920 ξ$792^2)+η$792^3 (514-740 ξ$792+34 720 ξ$792^2-5464 ξ$792^3-36 912

                                                                    495 495 w1$792 w2$792 (-4 η$792^3+2 ξ$792 (-1

             - 1 981 980 w1$792 w2$792 (48 η$792^6-8 η$792^5 (91+190 ξ$792+132 ξ$792^2)-4 η$792 ξ$792^2 (-72-151 ξ$792+333 ξ$792^2+364 ξ$

                        3 963 960 w1$792 w2$792 (152 η$792^6-6 η$792^5 (57+104 ξ$792+8 ξ$792^2)-8 ξ$792^3 (9-21 ξ$792-32 ξ$792^2+8 ξ$792

      - 495 495 w1$792 w2$792 (48 η$792^6+8 η$792^5 (643+1639 ξ$792+951 ξ$792^2)+4 ξ$792^2 (-4+56 ξ$792-1125 ξ$792^2-1034 ξ$792^3+136 ξ$792^4

                      495 495 w1$792 w2$792 (152 η$792^6+2 η$792^5 (57+182 ξ$792+90 ξ$792^2)+η$792^4 (-310+77 ξ$792+514 ξ$792^2+270 ξ$792

            495 495 w1$792 w2$792 (5824 η$792^6+24 η$792^5 (-191-277 ξ$792+279 ξ$792^2)-24 η$792 ξ$792^2 (11+6 ξ$792+274 ξ$792^2+465 ξ$792

                        - 495 495 w1$792 w2$792 (304 η$792^6-4 η$792^5 (49-107 ξ$792+69 ξ$792^2)+4 ξ$792^3 (17-27 ξ$792+48 ξ$792^2-68 ξ$

Ke=    495 495 w1$792 w2$792 (η$792^5 (722+2936 ξ$792+2166 ξ$792^2)+η$792^3 (396+4937 ξ$792+1050 ξ$792^2-13 124 ξ$792^3-9828 ξ$792^4)+η$792

              - 495 495 w1$792 w2$792 (76 η$792^5 (1+12 ξ$792+3 ξ$792^2)+η$792^4 (-155-1500 ξ$792+549 ξ$792^2+1272 ξ$792^3-342 ξ$792^4)+2 η$

      - 495 495 w1$792 w2$792 (48 η$792^6+12 η$792^5 (184+483 ξ$792+273 ξ$792^2)+η$792^3 (2261+7966 ξ$792-3940 ξ$792^2-25 758 ξ$792^3-15 690 ξ$

                 495 495 w1$792 w2$792 (304 η$792^6+4 η$792^5 (-57-65 ξ$792+33 ξ$792^2)-2 ξ$792 (1-50 ξ$792-43 ξ$792^2+100 ξ$792^3+52 ξ$792

       - 495 495 w1$792 w2$792 (η$792^5 (722+2936 ξ$792+2166 ξ$792^2)-η$792^3 (-396+375 ξ$792+5102 ξ$792^2+6836 ξ$792^3+2076 ξ$792^4)+η$792

                  495 495 w1$792 w2$792 (76 η$792^5 (1+12 ξ$792+3 ξ$792^2)+η$792^3 (86-840 ξ$792+352 ξ$792^2-578 ξ$792^3-708 ξ$792^4)+η$79

     495 495 w1$792 w2$792 (5824 η$792^6+4 η$792^5 (-1880-555 ξ$792+2757 ξ$792^2)+η$792^3 (3045-2570 ξ$792+13 500 ξ$792^2+10 402 ξ$792^3-15 69

              - 495 495 w1$792 w2$792 (608 η$792^6-4 η$792^5 (212-423 ξ$792+81 ξ$792^2)-2 ξ$792 (1-50 ξ$792-43 ξ$792^2+100 ξ$792^3+52 ξ$792

Set::shape : Lists {ξ$1127, w1$1127} and LineGaussRuleInfo[{2, True}, 1] are not the same shape. ≫

General::stop : Further output of Set::shape will be suppressed during this calculation. ≫
```

Problem 2 - Book Exercise 24.2

In[6]:=

```
PrintWithStyle["First, all shape functions for the
   element were derived by inspection from the ones given in 24.1"]
PrintWithStyle["I'm going to drop this homework because I just don't
   have time or the gumption to put all I can into it. Sorry!"]
```

Out[6]=

```
First, all shape functions for the element were
  derived by inspection from the ones given in 24.1
```

Out[7]=

```
I'm going to drop this homework because I just don't have
  time or the gumption to put all I can into it. Sorry!
```