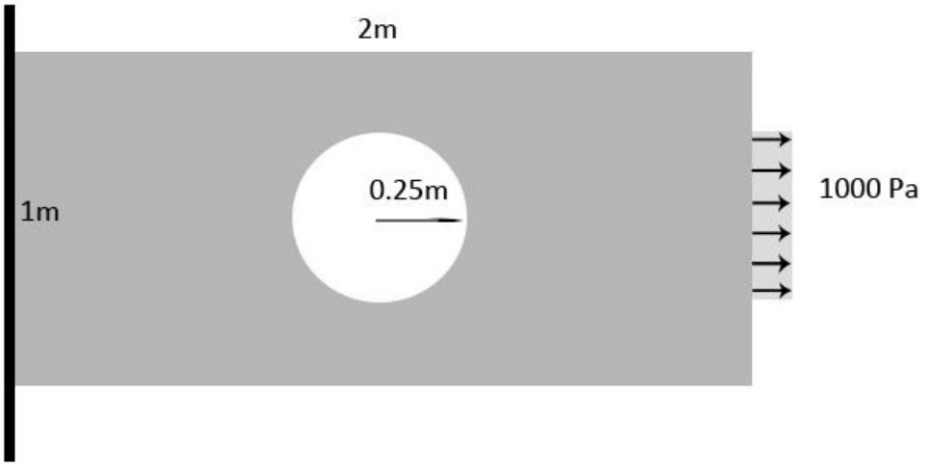
فهرست

موضوع پروژه

تحلیل استاتیک قطعه‌ای از پیش تعریف شده مشابه شکل به روش المان‌های محدود

هدف از انجام پروژه

قطعه‌ای به صورت شکل زیر تعریف شده است. این قطعه را به روش المانهای محدود حل کرده و مقادیر جابجایی ها، نیروهای تکیه گاهی، کرنش و تنش را محاسبه خواهیم کرد. در ادامه به شرح این مراحل خواهیم پرداخت.



E = 200 GPa , v = 0.3 , h = 0.05 mm

تصویر ۱ تصویر شماتیک قطعه مورد تحلیل

روابط تئوری مورد استفاده

در روند حل این مسئله، از روابط زیر استفاده گردیده است که در این بخش ذکر می‌کنیم.

$$E = \begin{bmatrix} \frac{E}{1-\nu^2} & \frac{\nu E}{1-\nu^2} & 0 \\ \frac{\nu E}{1-\nu^2} & \frac{E}{1-\nu^2} & 0 \\ 0 & 0 & \frac{E}{2(1+\nu)} \end{bmatrix}$$

$$\sigma = \begin{pmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \end{pmatrix}$$

$$\varepsilon = \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ 2\varepsilon_{xy} \end{pmatrix}$$

$$\varepsilon = \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ 2\varepsilon_{xy} \end{pmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix} \begin{pmatrix} u(x,y) \\ v(x,y) \end{pmatrix}$$

$$\sigma = \begin{pmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \end{pmatrix} = \begin{bmatrix} \frac{E}{1-\nu^2} & \frac{\nu E}{1-\nu^2} & 0 \\ \frac{\nu E}{1-\nu^2} & \frac{E}{1-\nu^2} & 0 \\ 0 & 0 & \frac{E}{2(1+\nu)} \end{bmatrix} \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ 2\varepsilon_{xy} \end{pmatrix}$$

$$\underline{U} = \begin{pmatrix} u(x,y) \\ v(x,y) \end{pmatrix} = \begin{bmatrix} N^e{}_1 & 0 & N^e{}_2 & 0 & N^e{}_3 & 0 \\ 0 & N^e{}_1 & 0 & N^e{}_2 & 0 & N^e{}_3 \end{bmatrix} \underline{u^e} = \underline{N u^e}$$

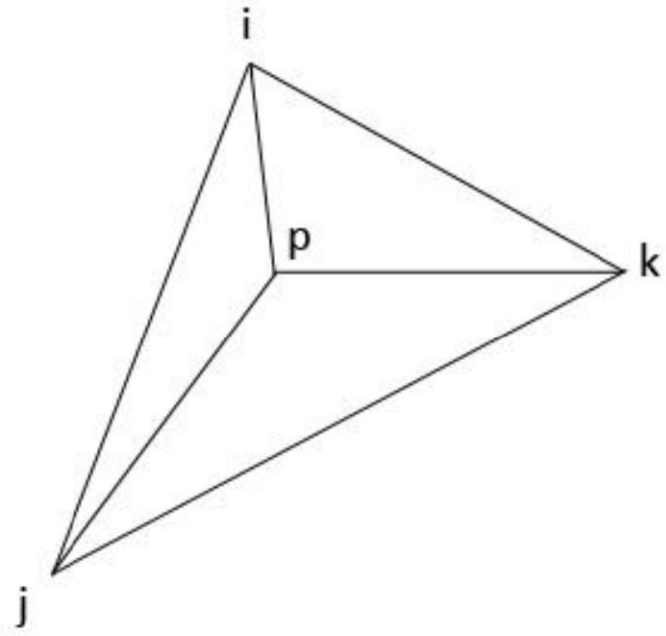
$$\begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ 2\varepsilon_{xy} \end{pmatrix} = \begin{bmatrix} \frac{\partial N^e{}_1}{\partial x} & 0 & \frac{\partial N^e{}_2}{\partial x} & 0 & \frac{\partial N^e{}_3}{\partial x} & 0 \\ 0 & \frac{\partial N^e{}_1}{\partial y} & 0 & \frac{\partial N^e{}_2}{\partial y} & 0 & \frac{\partial N^e{}_3}{\partial y} \\ \frac{\partial N^e{}_1}{\partial y} & \frac{\partial N^e{}_1}{\partial x} & \frac{\partial N^e{}_2}{\partial y} & \frac{\partial N^e{}_2}{\partial x} & \frac{\partial N^e{}_3}{\partial y} & \frac{\partial N^e{}_3}{\partial x} \end{bmatrix} \underline{u^e} = \underline{Bu^e}$$

$$\varepsilon^e = \underline{Bu^e}$$

$$\sigma^e = \underline{EBu^e}$$

$$\underline{k^e} = \frac{1}{2}\int h\underline{B^T EB}\, d\Omega^e$$

Area Coordinates



تصویر ۲ نمایی از یک المان مثلثی

$$N_i = L_i = \frac{Area(P_{jk})}{A}$$

$$N_j = L_j = \frac{Area(P_{ik})}{A}$$

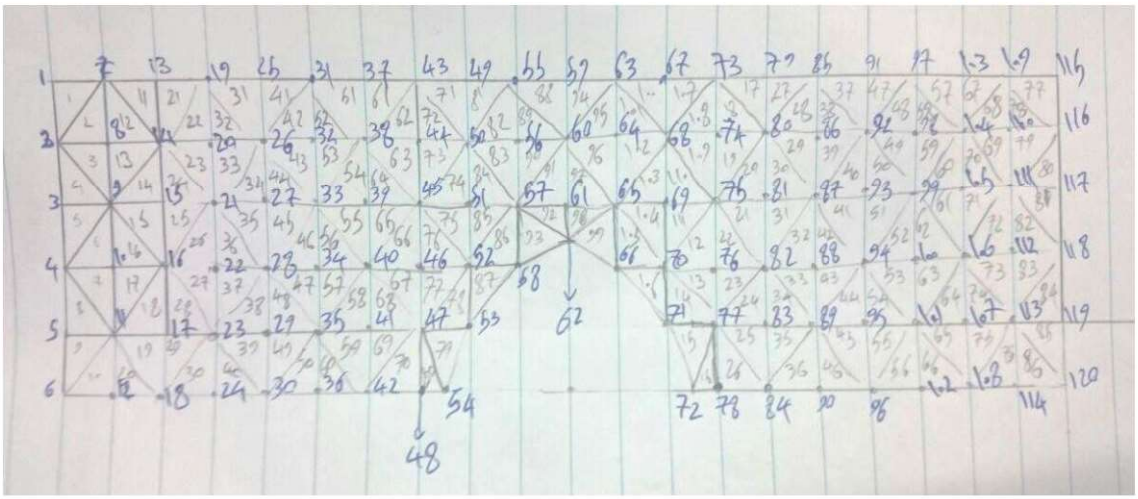
$$N_k = L_k = \frac{Area(P_{ji})}{A}$$

$$\int {L^\alpha}_i \, {L^\beta}_j {L^\gamma}_k \, dA = \frac{\alpha!\ \beta!\ \gamma!}{(\alpha + \beta + \gamma + 2)!}\ 2A$$

نحوه ی کد نویسی برای حل مسئله

در این پروژه از نرم افزار Mathematica برای نوشتن کدها استفاده شده است.

ابتدا باید بتوانیم قطعه را مش زده و اطلاعات مربوط به آن را ثبت نماییم. برای این کار از المان های مثلثی بهره برده ایم. در ابتدا فرم کلی قطعه را روی کاغذ رسم و مش دلخواه و مناسب را ترسیم کردیم. در ترسیم مش ها به نکات لازم توجه شد، از جمله کیفیت مش ها و مختصات نقاط هر نود، تا ما را برای رسیدن به جواب مطلوب و سادگی در حل کمک کنند. تصویر شماره ۳، نمایی از روش در پیش گرفته شده برای بدست آوردن مقادیر نودها است. البته این شکل تقریبی و برای درک مفهومی مسئله است. نکته‌ی قابل توجه آن است که ما نیمی از قطعه را که دارای تقارن با بخش دیگر است تحلیل کرده و در نهایت میتوان نتیجه را به طور متقارن به کل قطعه تعمیم داد. تعداد کل المان‌ها ۱۸۶ و تعداد کل نودها ۱۲۰ عدد می‌باشد.



تصویر ۳ نحوه شماره گذاری نودها و المان‌ها

حال در جهت نوشتن کد قدم برمیداریم. در ابتدا نیاز به یک ماتریس داریم که نشان دهنده ی چگونگی اتصال نود ها است. به این شکل که ستون اول شماره ی المان, و سه ستون بعدی به ترتیب شماره ی نود ها است . (همه طبق ترتیب معین و ساعتگرد.)

قطعه کد زیر نشان دهنده ی این مرحله میباشد :

```
NodalConnect={{1,1,7,2},{2,2,7,8},{3,2,8,9},{4,2,9,3},{5,3,9,4},{6,4,9,10},
```

در مرحله ی بعد نیاز به مختصات هر نود داریم. برای این کار از حلقه استفاده مینماییم و به صورت دستی مقادیر وارد نمی‌شوند. همان طور که با توجه به نظم موجود مشخص است، برای هر ستون از نود ها، میتوان نظمی برای بدست آورد مختصات نودها در نظر گرفت. به این شکل که در هر ستون ۶ نود دارای طول یکسان بوده و به ترتیب افزایش شماره نود، عرض آن ۱۰سانتی متر کاهش می‌یابد. قبل و بعد از دایره که تعداد نودها در جهت عمودی برابر است، از دو کد استفاده می‌شود. کد زیر مختصات نودها را قبل از دایره می‌دهد که مربوط به نود های ۱ الی ۴۸ میباشد :

```
NodeNum=1;
x=0;
    For[i=1,i<=8,i++,
        y=50/100;
        For[j=1,j<=6,j++,
            NodeCoord[[NodeNum,1]]=NodeNum;
            NodeCoord[[NodeNum,2]]=x;
            NodeCoord[[NodeNum,3]]=y;
            NodeNum++;
            y-=10/100
        ];
    x+=10/100
    ]
```

کد زیر نیز عینا مانند کد بالاست، اما برای بدست آورد مختصات نودهای ۷۳ الی ۱۲۰ که به صورت زیر است:

```
NodeNum=73;
x=130/100;
    For[i=1,i<=8,i++,
        y=50/100;
        For[j=1,j<=6,j++,
            NodeCoord[[NodeNum,1]]=NodeNum;
            NodeCoord[[NodeNum,2]]=x;
            NodeCoord[[NodeNum,3]]=y;
            NodeNum++;
            y-=10/100
        ];
    x+=10/100
    ]
```

برای نقاط بالایی دایره نیز که عینا همین کد ها به صورت زیر نوسته میشود, با این تفاوت که دیگر ۶ نود در یک ستون وجود ندارد و برای هر ستون میبایست از حلقه های جداگانه استفاده نمودوکد زیر نمونه ی یکی از این ستون ها است :

```
NodeNum=59;
x=100/100;
    For[i=1,i<=1,i++,
        y=50/100;
        For[j=1,j<=3,j++,
            NodeCoord[[NodeNum,1]]=NodeNum;
            NodeCoord[[NodeNum,2]]=x;
            NodeCoord[[NodeNum,3]]=y;
            NodeNum++;
            y-=10/100
        ];
    x+=10/100
    ]
```

که برای ثبت مختصات نود ۵۹ الی ۶۱ استفاده شده است.

برای بقیه ستون ها نیز به صورت مشابه عمل شده است. البته به این نکته توجه شود که قبل از این مراحل، ماتریس صفری ساخته شد تا در این حلقه ها مقادیر بدست آمده در آن جایگذاری شود.

حال سه نود باقی می‌ماند که در هیچ یک از حلقه ها موجود نیست، به علت اینکه دارای نظمی نبوده‌اند و مقدار طول آنها و یا عرضشان به صورت متفاوت تغییر کرده است. این سه نود را به صورت دستی وارد ماتریس مربوطه میکنیم.

قدم بعدی بدست آورد مساحت و Shape Functionهای مربوط به هر المان است. برای این کار یک ماتریس با نام SF و سایز ۱۸۶ در ۸ تعریف میکنیم که دارای ۸ ستون است. ستون اول مربوط به شماره المان، سه ستون بعدی شماره‌ی نودها، سه ستون بعدی Shape Function های مربوطه است که با استفاده از Area Coordinate بدست آمده‌اند. در این مرحله r طول و s عرض در نظر گرفته شده است. حلقه‌ی زیر تمام Shape Function ها و مساحت‌ها را برای المان‌ها محاسبه میکند:

```
For[i=1,i<=EleCount,i++,
    For[j=1,j<=4,j++,
        SF[[i,j]]=NodalConnect[[i,j]]
        ];
    SF[[i,8]]=-0.5*Det[
                    {{1,NodeCoord[[NodalConnect[[i,2]],2]],NodeCoord[[NodalConnect[[i,2]],3]]},
                    {1,NodeCoord[[NodalConnect[[i,3]],2]],NodeCoord[[NodalConnect[[i,3]],3]]},
                    {1,NodeCoord[[NodalConnect[[i,4]],2]],NodeCoord[[NodalConnect[[i,4]],3]]}}
                    ];

    SF[[i,5]]=-0.5*Det[
                    {{1,r,s},
                    {1,NodeCoord[[NodalConnect[[i,3]],2]],NodeCoord[[NodalConnect[[i,3]],3]]},
                    {1,NodeCoord[[NodalConnect[[i,4]],2]],NodeCoord[[NodalConnect[[i,4]],3]]}}
                    ]/SF[[i,8]];

    SF[[i,6]]=-0.5*Det[
                    {{1,NodeCoord[[NodalConnect[[i,2]],2]],NodeCoord[[NodalConnect[[i,2]],3]]},
                    {1,r,s},
                    {1,NodeCoord[[NodalConnect[[i,4]],2]],NodeCoord[[NodalConnect[[i,4]],3]]}}
                    ]/SF[[i,8]];

    SF[[i,7]]=-0.5*Det[
                    {{1,NodeCoord[[NodalConnect[[i,2]],2]],NodeCoord[[NodalConnect[[i,2]],3]]},
                    {1,NodeCoord[[NodalConnect[[i,3]],2]],NodeCoord[[NodalConnect[[i,3]],3]]},
                    {1,r,s}}
                    ]/SF[[i,8]]
    ]
```

حال می‌باسیت بر اساس کد زیر، مراحل بعدی صورت بگیرد. ابتدا ماتریس سختی هر المان محاسبه شده، سپس هر درایه از آن ماتریس، با توجه به شماره ی نودها و المان استفاده شده در مکان مناسب خود در ماتریس سختی کل جاگذاری میشود. قبل از آن ماتریس kg با سایز ۲۴۰ در ۲۴۰ تعریف میشود که پس از محاسبه ی مقادیر در آن جایگذاری شوند. این ماتریس ماتریس سختی کلی نیست، ماتریس سختی‌ها باید در نصف ضخامت ضرب شوند که پس از آن ماتریس KgFinal نشان دهنده‌ی ماتریس سختی کل میباشد که همان kg است که در نصف ضخامت ضرب شده است. این کد به شرح زیر است:

```
For[EleNum=1,EleNum<=186,EleNum++,
    B={{D[SF[[EleNum,5]],r],0,D[SF[[EleNum,6]],r],0,D[SF[[EleNum,7]],r],0},
       {0,D[SF[[EleNum,5]],s],0,D[SF[[EleNum,6]],s],0,D[SF[[EleNum,7]],s]},
       {D[SF[[EleNum,5]],s],D[SF[[EleNum,5]],r],D[SF[[EleNum,6]],s],D[SF[[EleNum,6]],r],D[SF[[EleNum,7]
       Ke=Transpose[B].EMat.B*SF[[EleNum,8]];

       Kg[[NodalConnect[[EleNum,2]]*2-1,NodalConnect[[EleNum,2]]*2-1]]+=Ke[[1,1]];
       Kg[[NodalConnect[[EleNum,2]]*2-1,NodalConnect[[EleNum,2]]*2]]+=Ke[[1,2]];
       Kg[[NodalConnect[[EleNum,2]]*2,NodalConnect[[EleNum,2]]*2-1]]+=Ke[[2,1]];
       Kg[[NodalConnect[[EleNum,2]]*2,NodalConnect[[EleNum,2]]*2]]+=Ke[[2,2]];

       Kg[[NodalConnect[[EleNum,3]]*2-1,NodalConnect[[EleNum,3]]*2-1]]+=Ke[[3,3]];
       Kg[[NodalConnect[[EleNum,3]]*2-1,NodalConnect[[EleNum,3]]*2]]+=Ke[[3,4]];
       Kg[[NodalConnect[[EleNum,3]]*2,NodalConnect[[EleNum,3]]*2-1]]+=Ke[[4,3]];
       Kg[[NodalConnect[[EleNum,3]]*2,NodalConnect[[EleNum,3]]*2]]+=Ke[[4,4]];

       Kg[[NodalConnect[[EleNum,4]]*2-1,NodalConnect[[EleNum,4]]*2-1]]+=Ke[[5,5]];
       Kg[[NodalConnect[[EleNum,4]]*2-1,NodalConnect[[EleNum,4]]*2]]+=Ke[[5,6]];
       Kg[[NodalConnect[[EleNum,4]]*2,NodalConnect[[EleNum,4]]*2-1]]+=Ke[[6,5]];
       Kg[[NodalConnect[[EleNum,4]]*2,NodalConnect[[EleNum,4]]*2]]+=Ke[[6,6]];


       Kg[[NodalConnect[[EleNum,2]]*2-1,NodalConnect[[EleNum,3]]*2-1]]+=Ke[[1,3]];
       Kg[[NodalConnect[[EleNum,2]]*2-1,NodalConnect[[EleNum,3]]*2]]+=Ke[[1,4]];
       Kg[[NodalConnect[[EleNum,2]]*2,NodalConnect[[EleNum,3]]*2-1]]+=Ke[[2,3]];
       Kg[[NodalConnect[[EleNum,2]]*2,NodalConnect[[EleNum,3]]*2]]+=Ke[[2,4]];

       Kg[[NodalConnect[[EleNum,2]]*2-1,NodalConnect[[EleNum,4]]*2-1]]+=Ke[[1,5]];
       Kg[[NodalConnect[[EleNum,2]]*2-1,NodalConnect[[EleNum,4]]*2]]+=Ke[[1,6]];
       Kg[[NodalConnect[[EleNum,2]]*2,NodalConnect[[EleNum,4]]*2-1]]+=Ke[[2,5]];
       Kg[[NodalConnect[[EleNum,2]]*2,NodalConnect[[EleNum,4]]*2]]+=Ke[[2,6]];

       Kg[[NodalConnect[[EleNum,3]]*2-1,NodalConnect[[EleNum,2]]*2-1]]+=Ke[[3,1]];
       Kg[[NodalConnect[[EleNum,3]]*2-1,NodalConnect[[EleNum,2]]*2]]+=Ke[[3,2]];
       Kg[[NodalConnect[[EleNum,3]]*2,NodalConnect[[EleNum,2]]*2-1]]+=Ke[[4,1]];
       Kg[[NodalConnect[[EleNum,3]]*2,NodalConnect[[EleNum,2]]*2]]+=Ke[[4,2]];

       Kg[[NodalConnect[[EleNum,3]]*2-1,NodalConnect[[EleNum,4]]*2-1]]+=Ke[[3,5]];
       Kg[[NodalConnect[[EleNum,3]]*2-1,NodalConnect[[EleNum,4]]*2]]+=Ke[[3,6]];
       Kg[[NodalConnect[[EleNum,3]]*2,NodalConnect[[EleNum,4]]*2-1]]+=Ke[[4,5]];
       Kg[[NodalConnect[[EleNum,3]]*2,NodalConnect[[EleNum,4]]*2]]+=Ke[[4,6]];

       Kg[[NodalConnect[[EleNum,4]]*2-1,NodalConnect[[EleNum,2]]*2-1]]+=Ke[[5,1]];
       Kg[[NodalConnect[[EleNum,4]]*2-1,NodalConnect[[EleNum,2]]*2]]+=Ke[[5,2]];
       Kg[[NodalConnect[[EleNum,4]]*2,NodalConnect[[EleNum,2]]*2-1]]+=Ke[[6,1]];
       Kg[[NodalConnect[[EleNum,4]]*2,NodalConnect[[EleNum,2]]*2]]+=Ke[[6,2]];

       Kg[[NodalConnect[[EleNum,4]]*2-1,NodalConnect[[EleNum,3]]*2-1]]+=Ke[[5,3]];
       Kg[[NodalConnect[[EleNum,4]]*2-1,NodalConnect[[EleNum,3]]*2]]+=Ke[[5,4]];
       Kg[[NodalConnect[[EleNum,4]]*2,NodalConnect[[EleNum,3]]*2-1]]+=Ke[[6,3]];
       Kg[[NodalConnect[[EleNum,4]]*2,NodalConnect[[EleNum,3]]*2]]+=Ke[[6,4]]

]
KgFinal=0.5*h*Kg;
```

قدم بعدی نوشتن ماتریس مربوط به نیرو است. طبق صورت مسئله نیرو ی گسترده به لبه ی سمت راست قطعه وارد میشود. این نیرو را روی نود ها تقسیم میکنیم. همچنین نیرو های مجهول تکیه گاهی را نیز با مقادیر نشان داده شده به صورت مجهول وارد میکنیم. طبق کد زیر :

```
Ftot=1000*0.5*0.05*10^(-3);
(*Distribiution of total force on 7 nodes (rounded up!) *)
FNode=Ftot/7;
FMat=ConstantArray[0,{240,1}];
FMat[[117*2-1]]={FNode};
FMat[[118*2-1]]={FNode};
FMat[[119*2-1]]={FNode};
FMat[[120*2-1]]={FNode};

(*Support Forces*)
FMat[[1]]={f1x};
FMat[[2]]={f1y};

FMat[[3]]={f2x};
FMat[[4]]={f2y};

FMat[[5]]={f3x};
FMat[[6]]={f3y};

FMat[[7]]={f4x};
FMat[[8]]={f4y};

FMat[[9]]={f5x};
FMat[[10]]={f5y};

FMat[[11]]={f6x};
FMat[[12]]={f6y};
```

حال تمام مقادیر لازم را بدست آورده ایم. ماتریس سختی و ماتریس مربوط به نیروها به دست آمده اند. حال باید معادله ی f=ku را حل نمود و مقادیر جابجایی را بدست آورد. مقادیر مربوط به جابجایی نودهای تکیه گاه معلوم و برابر صفر است اما نیروهای آن مجهول میباشد. یعنی نود های ۱ تا ۶ دارای جابجایی صفر و هر کدام دارای دو نیروی عمود بر هم مجهول هستند. با نوشتن ۱۲ سطر از معادله‌ی حاصل از ماتریس ها که معادل همین جابجایی ها و نیروها است میتوان مقادیر نیرو را بدست آورد. این مراحل طبق کد زیر انجام میپذیرد :

```
UMat=Inverse[KgFinal].FMat;
ForceResult=Solve[{UMat[[1]]==0,UMat[[2]]==0,UMat[[3]]==0,UMat[[4]]==0,UMat[[5]]==0,UMat[[6]]==0,UMat[[
DispResult=((UMat/.{ForceResult})[[1,1]])
```

این مقادیر را ثبت کرده و مجددا در ماتریس نیرو جایگذاری میکنیم تا ماتریس نیرو دیگر مجهولی نداشته باشد. سپس تمام معادلات ناشی از ماتریس موجود را مینویسیم که برابر بردار جابجایی است. بنابراین بردار جابجایی را نیز بدست خواهیم آورد.کد این قسمت به کد بالایی قابل مشاهده است.که مقادیر DispResult مقادیر جابجایی را میدهد.

حال با استفاده از مقادیر جابجایی بدست آمده, میتوان کرنش و تنش را برای هر المان بدست آورد. در قسمت زیر کرنش های طولی , عرضی و برشی (که دو برابر کرنش برشی است) بدست می آید. برای این کار ماتریس با ۱۸۶ سطر و ۳ ستون تعریف

میشود که ستون اول کرنش طولی, ستون دوم کرنش عرضی و ستون سوم دو برابر کرنش برشی است. ابتدا باید ماتریس B استخراج شود که ژاکوبین ماتریس Shape Function است. سپس باید در بردار جابجایی هر المان ضرب شود. در کد زیر ماتریس جابجایی نود ها ی هز المان را از بردار جابجایی کلی استجراج میکنیم. سپس مقادیر را در ماتریس صفری که برای کرنش ساختیم جای گذاری خواهیم نمود. همچنین در این بخش ماتریس تنش را نیز ساخته و پر میکنیم که تنها باید در ماتریس مدول الاستیسیته ضرب گردد. برای این قسمت نیز مانند قبل متاریس صفری مسازیم سپس مقادیر بدست آمده را در آن جایگذاری میکنیم که ستون اول تنش طولی, ستون دوم تنش عرضی و ستون سوم تنش برشی است. سپس ماتریس جدیدی تعریف میکنیم که با استفاده از سه تنش بدست آمده , تنش فون میسز را محاسبه میکند و در برای هر المان ذخیره مینماید :

```
EpsMat=ConstantArray[0,{186,3}];
StressMat=ConstantArray[0,{186,3}];
vonMises=ConstantArray[0,{186,1}];

For[EleNum=1,EleNum<=186,EleNum++,
    B={{D[SF[[EleNum,5]],r],0,D[SF[[EleNum,6]],r],0,D[SF[[EleNum,7]],r],0},
       {0,D[SF[[EleNum,5]],s],0,D[SF[[EleNum,6]],s],0,D[SF[[EleNum,7]],s]},
       {D[SF[[EleNum,5]],s],D[SF[[EleNum,5]],r],D[SF[[EleNum,6]],s],D[SF[[EleNum,6]],r],D[SF[[EleNum,7]

           Ue={
           DispResult[[NodalConnect[[EleNum,2]]*2-1]],
           DispResult[[NodalConnect[[EleNum,2]]*2]],
           DispResult[[NodalConnect[[EleNum,3]]*2-1]],
           DispResult[[NodalConnect[[EleNum,3]]*2]],
           DispResult[[NodalConnect[[EleNum,4]]*2-1]],
           DispResult[[NodalConnect[[EleNum,4]]*2]]};

           epsE=B.Ue;
(*Global Epsilon Matrix= 186*3: exx, eyy, 2exy*)
           EpsMat[[EleNum,1]]=epsE[[1,1]];
           EpsMat[[EleNum,2]]=epsE[[2,1]];
           EpsMat[[EleNum,3]]=epsE[[3,1]];

           strssE=EMat.B.Ue;
           StressMat[[EleNum,1]]=strssE[[1,1]];
           StressMat[[EleNum,2]]=strssE[[2,1]];
           StressMat[[EleNum,3]]=strssE[[3,1]];

           vonMises[[EleNum,1]]=Sqrt[StressMat[[EleNum,1]]^2+StressMat[[EleNum,2]]^2-StressMat[[EleNum,
]
```

در قسمت بعد تلاش برای ترسیم این تنش صورت گرفته است. برای این کار یک نقطه به عنوان نماینده المان انتخاب شده است که برابر میانگین مختصات است. سپس تنش هر المان را برابر تنش فون میسز در نظر گرفته ایم :

```
StressCoord=ConstantArray[0,{186,3}];
For[EleNum=1,EleNum<=186,EleNum++,
    StressCoord[[EleNum,3]]=vonMises[[EleNum,1]];
    StressCoord[[EleNum,1]]=(NodeCoord[[NodalConnect[[EleNum,2]],2]]+NodeCoord[[NodalConnect[[EleNum,3]]
    StressCoord[[EleNum,2]]=(NodeCoord[[NodalConnect[[EleNum,2]],3]]+NodeCoord[[NodalConnect[[EleNum,3]]
]
```

در این ماتریس، ستون اول مقدار طول المان (میانگین طولهای نودهای المان) و ستون دوم مقادیر عرض المان هاست. ستون سوم نیز تنش فون میسز آن میباشد.

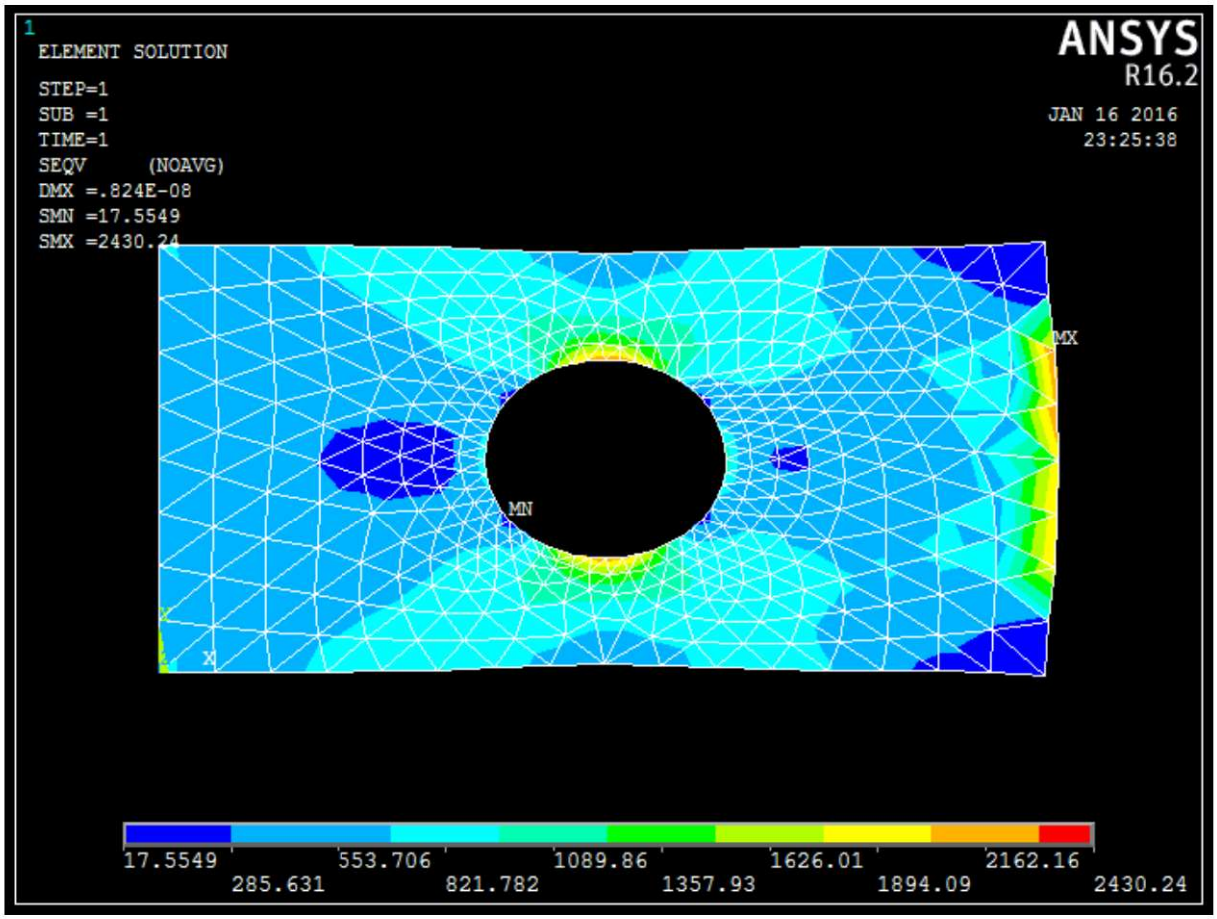
حال با دستور زیر این ماتریس رسم شده است:

```
ListDensityPlot[StressCoord]
```

شبیه سازی مسئله در APDL

در این بخش، مسئله توسط نرم افزار انسیس و بخش APDL شبیه سازی و حل شده است. در این تحلیل، المان ها به همان صورت قبل که برای کد نویسی استفاده گردیده، زده شده است.

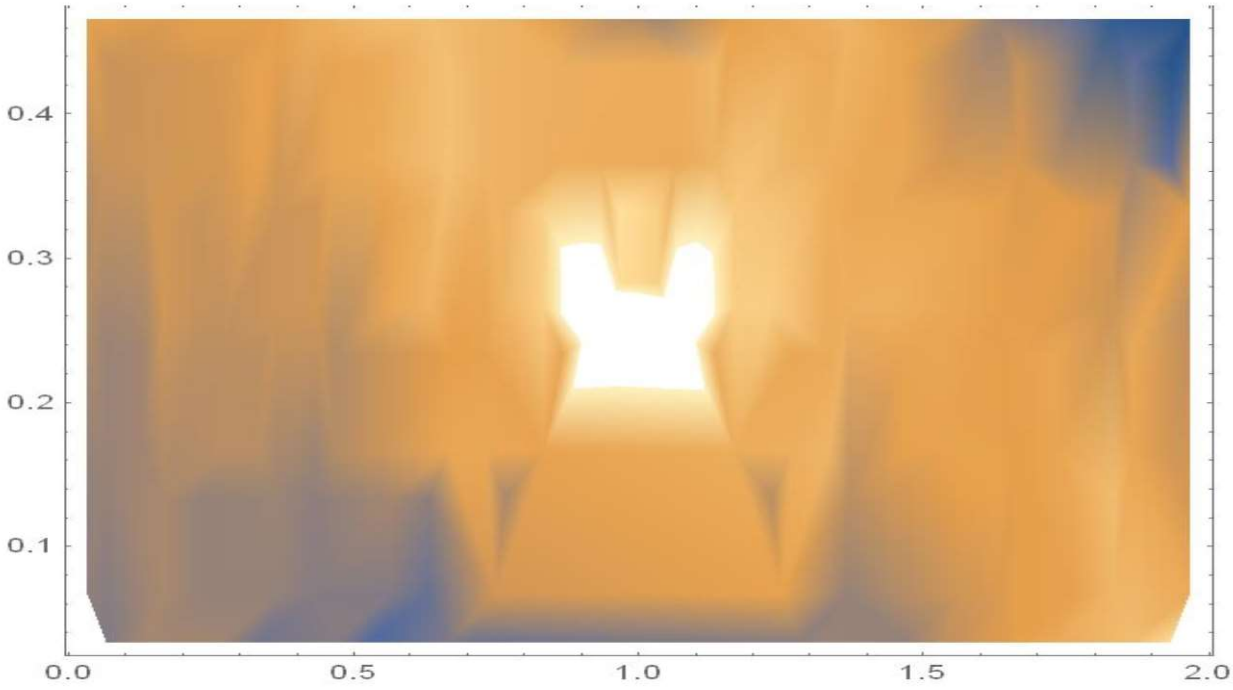
در شکل زیر نتایج این تحلیل را میبینیم.



تصویر ۴ نتایج شبیه‌سازی در APDL

نتایج کد :

شکل زیر نیز تصویر نمودار تنش برای المان ها میباشد :



تصویر ۵ کانتور تنش فون میسز روی المان‌ها

در ادامه تمامی مقادیر موجود در این برنامه مشاهده میشود.

توجه فرمایید

ماتریس NodaleConnectنشان دهنده ی نحوه ی اتصال و ترتیب نود ها در المان هاست.

ماتریس NodeCoord نشان دهنده ی مختصات نود های هر المان میباشد.

ماتریس SF شامل شماره ی نود در ستون سمت چپ, شماره ی نود ها به ترتیب, Shape Functionها به ترتیب و در نهایت ستون آخر که مربوط به مساحت هر المان است.

ماتریس KgFinal نشان دهنده ی ماتریس سختی کلی است .

بردار FMat بردار نیروهاست که مقادیر مجهول نیروهای تکیه گاهی را شامل میشود.

بردار Dispresult جابجایی هر نود را میدهد که از حل روابط بدست آمده حاصل شده است.

ماتریس EpsMat ماتریسی شامل کرنش ها است.هر ستون مربوط به یک المان میباشد که ستون اول کرنش طولی, ستون دوم کرنش عرضی, و ستون سوم دو برابر کرنش برشی است.

ماتریس StressMat ماتریسی است که هر سطر تنش مربوطبه المان را میدهد. ستون اول تنش طولی,ستون دوم تنش عرضی و ستون سوم تنش برشی است.

بردار vonMises تنش فون میسز است که برای هر المان طبق رابطه ی موجود به دست آمده است.هر سطر مربوط به المان خود است.

ماتریس StressCoord شامل سه ستون است.ستون اول مختصات طولی المان (میانگین طول نودها) ,ستون دوم مربوط به عرض المان (میانگین عرض نودها) و ستون آخر نیز مربوط به تنش فون میسز مربوط به هر المان است که برای رسم نمودار از آن استفاده شده است.

این مقادیر در ضمیمه آمده است.

مقایسه ی نتایج و محاسبه ی ضریب تمرکز تنش

ابتدا ضریب تمرکز تنش تقریبی را با توجه به مقادیر بدست آمده در تحلیل APDL محاسبه میکنیم.

$$K = \frac{\sigma_{max}}{\sigma_{nom}}$$

$$K \approx \frac{1600}{550} \approx 2.9$$

حال با توجه به نتایج حاصل از کد, مقدار ضریب تمرکز تنش را محاسبه خواهیم کرد.

$$K \approx \frac{1900}{550} \approx 3.4$$

که مشاهده میشود تا حد قابل قبولی نتیجه نزدیک است.

تحلیل خطا

در محاسبات انجام شده، خطاهای زیادی میتوانند دخیل باشند. از جمله خطا در وارد کردن اطلاعات و خطاهای محاسباتی و عملیاتی توسط خود نویسنده‌ی کد.

اما مهمترین خطایی که با آن سر و کار بسیار داشتیم، خطاهای عددی بودند که در فرایند محاسبات بسیار به وجود می آمد. حتی نرم افزار هشدار این گونه خطاها را میداد. به عنوان مثال اعداد موجود در ماتریس ها دارای اختلاف بالایی بودند، که موجب ایجاد چنین خطاهایی شده است و در نتایج میتواند تاثیر قابل توجه‌ای داشته باشد.

خطای ایجاد شده در اثر استفاده از روش عددی المان محدود نیز مطابق معمول وجود دارد که ناشی از نوع المان، اندازه‌ی آن، روش اختصاص هر مشخصه به کل المان (Reduced Integration یا Full Integration)، تعداد نودهای هر المان و سایر فاکتورهای رایج در روش المان‌های محدود است.