AFEM: Axisymmetric Project

Derek Lontine, Stuart Childs, Alex Bailey

April 27, 2016

Chapter 1

Abstract

One of the main loading scenarios we're looking at is the loading of incompressible materials and seeing the Simplify the entire paper...

Chapter 2

Introduction

What are we doing?

Why is it this important?

Does it have benefits?

In this project, we develop a 4-node axisymmetric element able to capture behavior of solid materials

Axisymmetric elements are for solids or revolution which have loadings which are also axisymmetric. These are ideal ...

As Fellipa says "The finite element discretization can be therefore confined to the {r,z} plane. The circumferential (?) dimension conceptually disappears from the FEM discretization."

Check to make sure the problem being modeled lies in the domain of axisymmetry. For a problem to be axisymmetric, , the geometry, material properties, boundary conditions and loading all must be axisymmetric. Should any of those things be out of place, the problem should be modeled using three dimensional elements.

One of the advantages of axisymmetric elements is the fact that a three dimensional system can be modeled using augmented 2 dimensional elements. The computational time is comparable to 2D bar systems, yet the behavior is that of a disk, cup, or (other solid of revolution)...

$$thisismath$$

We had a problem proposed (the 9 tests) and we used 3 different element types. They are standard axisymmetricQuad, AxiSym4Reduced and AxiSym4 SelectiveReduced. So that's 27 tests to run. we check those against the analytical solutions. which are =¿ shown. The error as a function of E and ν is plotted and patterns / phenomena are noted.

Chapter 3

Analytical Formulation

Displacement

$$\boldsymbol{u}(r,z) = \left[\begin{array}{c} u_r(r,z) \\ u_z(r,z) \end{array}\right]$$

Strain

$$[\boldsymbol{e}] = \left[\begin{array}{ccc} e_{rr} & e_{rz} & 0 \\ e_{rz} & e_{zz} & 0 \\ 0 & 0 & e_{\theta\theta} \end{array}\right]$$

where

$$e_{rr} = \frac{\partial u_r}{\partial r}, \quad e_{zz} = \frac{\partial u_z}{\partial z}, \quad e_{\theta\theta} = \frac{\partial u_\theta}{\partial \theta}, \quad \gamma_{rz} = \frac{\partial u_r}{\partial z} + \frac{\partial u_z}{\partial r} = e_{rz} + e_{zr} = 2e_{rz}$$

then

$$\boldsymbol{e} = \left[\begin{array}{c} e_{rr} \\ e_{zz} \\ e_{\theta\theta} \\ \gamma_{rz} \end{array}\right] = \left[\begin{array}{c} e_{rr} \\ e_{zz} \\ e_{\theta\theta} \\ 2e_{rz} \end{array}\right] = \left[\begin{array}{cc} \frac{\partial}{\partial r} & 0 \\ 0 & \frac{\partial}{\partial z} \\ \frac{1}{r} & 0 \\ \frac{\partial}{\partial z} & \frac{\partial}{\partial r} \end{array}\right] \left[\begin{array}{c} u_r(r,z) \\ u_z(r,z) \end{array}\right] = \boldsymbol{D}\boldsymbol{u}$$

The displacement interpolation is what we're used to:

$$\left[\begin{array}{c} u_r \\ u_z \end{array}\right] = \left[\begin{array}{ccccccc} N_1^e & 0 & N_2^e & 0 & \dots & N_n^e & 0 \\ 0 & N_1^e & 0 & N_2^e & \dots & 0 & N_n^e \end{array}\right] \boldsymbol{u}^e = \boldsymbol{N}\boldsymbol{u}^e$$

The Strain-Displacement Matrix, $\boldsymbol{B}$, is therefore:

$$\boldsymbol{e} = \left[\begin{array}{c} e_{rr} \\ e_{zz} \\ e_{\theta\theta} \\ \gamma_{rz} \end{array}\right] = \left[\begin{array}{ccccccc} \frac{\partial N_1^e}{\partial r} & 0 & \frac{\partial N_2^e}{\partial r} & 0 & \dots & \frac{\partial N_n^e}{\partial r} & 0 \\ 0 & \frac{\partial N_1^e}{\partial z} & 0 & \frac{\partial N_2^e}{\partial z} & \dots & 0 & \frac{\partial N_n^e}{\partial z} \\ \frac{N_1^e}{r} & 0 & \frac{N_2^e}{r} & 0 & \dots & \frac{N_n^e}{r} & 0 \\ \frac{\partial N_1^e}{\partial z} & \frac{\partial N_1^e}{\partial r} & \frac{\partial N_2^e}{\partial z} & \frac{\partial N_2^e}{\partial r} & \dots & \frac{\partial N_n^e}{\partial z} & \frac{\partial N_n^e}{\partial r} \end{array}\right] \boldsymbol{u}^e = \boldsymbol{B}\boldsymbol{u}^e$$

The finite element formulation
Fellipa

$$\boldsymbol{K}^e = \sum_{k=1}^{p}\sum_{l=1}^{p} w_k w_i \boldsymbol{B}^T \boldsymbol{E}\boldsymbol{B} r J_\Omega \tag{3.1}$$

$$\boldsymbol{f} = \int_{\Omega^e} r\boldsymbol{N}^T\boldsymbol{b}\,d\Omega + \int_{\Gamma^e} r\boldsymbol{N}^T\hat{\boldsymbol{t}}\,d\Gamma \tag{3.2}$$

Santos:

$$[\boldsymbol{K}^e] = \underbrace{\int_\Omega [\boldsymbol{B}]^T[\boldsymbol{D}][\boldsymbol{B}]dA}_{PlaneStrain} - \int_r \frac{1}{r}(\sigma_{33}\delta_{1i} - \sigma_{i1})w_i dr \tag{3.3}$$

$$\{\boldsymbol{f}\} = \int_{A^e} \rho\{\boldsymbol{b}\}[\boldsymbol{P}]dA + \int_S \{\boldsymbol{t}\}[\boldsymbol{P}]ds \tag{3.4}$$

Chapter 4

What we actually "did"

Fuller made CSDAX4F.py

Derek copied that, with the addition of the reduced stuff from the quad reduced to create CSDAX4R.py

Derek copied the selective reduced to pruduce the CSDAX4S.py

These are the 3 elements "we've" built.

Chapter 5

Computational Implementation

The axisymmetric elements produced are fundamentally quite simple. They modify the basic quad element found in pyfem2. They were developed and tested using the latest version of pyfem2 as given in update bea5af7 (4/19/2016).

Each of these elements has several things in common. Each of these elements modifies the $[\boldsymbol{B}]$ matrix so that the third row (B[2,0::2]) includes the shape functions divided by the radius from the axis of symmetry.

Additionally, each of these elements has a group of "formulation setting" functions to access the different modifications to the stiffness matrix as defined in chapter 2 in the Galerkin and Petrov-Galerkin formulations of axisymmetry.

5.1 Full integration

The full integration element was not developed by the authors, however, it is used heavily in the verification problems. This element is the base element and other elements such as the selective reduced and reduced integration elements are only slight modifications of this the base element.

The full integration element applies gauss points at $(x, y) = (\pm\sqrt{\frac{1}{3}}, \pm\sqrt{\frac{1}{3}})$ inside the element. This is applying 2 gauss points for each dimension. The gauss weight for each of the gauss points is equal to 1.

As will be discussed in great detail, this element can exhibit problems such as shear and volume locking in different scenarios.

```
from numpy import *
from .isop2_4 import CSDIsoParametricQuad4 as BaseElement
# --------------------------------------------------------------------------- #
# ------------------------ Axisymmetric Quad Element ------------------------ #
# --------------------------------------------------------------------------- #
class AxiSymmetricQuad4(BaseElement):
    ndir = 3
    nshr = 1
    integration = 4
    elefab = {'formulation': 1}
    gaussw = ones(4)
    gaussp = array([[-1., -1.], [ 1., -1.], [-1.,  1.], [ 1.,  1.]]) / sqrt(3.)
    @property
    def formulation(self):
        return self.axisymmetric
    @formulation.setter
    def formulation(self, arg):
        assert arg in (0, 1, 2)
        self.axisymmetric = arg
    def bmatrix(self, dN, N, xi, *args):
        rp = dot(N, self.xc[:,0])
        B = zeros((4, 8))
        B[0, 0::2] = B[3, 1::2] = dN[0, :]
        B[1, 1::2] = B[3, 0::2] = dN[1, :]
        B[2, 0::2] = N / rp
```

```python
        return B
```

5.2 Reduced Integration with Hourglass Control

```python
from numpy import *
from .isop2_4 import CSDIsoParametricQuad4 as BaseElement
# --------------------------------------------------------------------------- #
# --------- Axisymmetric Reduced Integration With Hourglass Element --------- #
# --------------------------------------------------------------------------- #
class AxiSymmetricQuad4Reduced(BaseElement):
    ndir = 3
    nshr = 1
    integration = 1
    elefab = {'formulation': 1}
    gaussw = array([4.])
    gaussp = array([[0.,0.]])
    hourglass_control = True
    #HOURGLASS CONTROL PARAMETERS
    hglassp = array([[0.,0.,]])
    hglassv = array([[1.,-1.,1.,-1.]])
    #REST
    @property
    def formulation(self):
        return self.axisymmetric
    @formulation.setter
    def formulation(self, arg):
        assert arg in (0, 1, 2)
        self.axisymmetric = arg
    def bmatrix(self, dN, N, xi, *args):
        rp = dot(N, self.xc[:,0])
        B = zeros((4, 8))
        B[0, 0::2] = B[3, 1::2] = dN[0, :]
        B[1, 1::2] = B[3, 0::2] = dN[1, :]
        B[2, 0::2] = N / rp
        return B
```

5.3 Implementation

The following listing is required for __init__.py in order for the elements above to be implemented into pyfem2. Note that the primary changes to the previously generated pyfem2 in the commit mentioned above is found in lines 4,5,22,23. The AxisymmetricQuad4 element was implemented in pyfem2 without any contribution of the authors.

```python
1  __all__ = ['PlaneStrainTria3', 'PlaneStressTria3',
2             'PlaneStrainQuad4BBar', 'PlaneStrainQuad4', 'PlaneStrainQuad4Reduced',
3             'PlaneStrainQuad4SelectiveReduced', 'PlaneStressQuad4',
4             'AxiSymmetricQuad4', 'AxiSymmetricQuad4SelectiveReduced',
5             'AxiSymmetricQuad4Reduced',
6             'PlaneStressQuad4Incompat', 'PlaneStrainQuad8BBar',
7             'PlaneStrainQuad8', 'PlaneStrainQuad8Reduced', 'PlaneStressQuad8',
8             'CSDIsoParametricElement', 'IsoPElement']
9
10 from .isoplib import CSDIsoParametricElement
11 IsoPElement = CSDIsoParametricElement
12
13 from .CSDT3EF import PlaneStrainTria3
14 from .CSDT3SF import PlaneStressTria3
15
16 from .CSDQ4EB import PlaneStrainQuad4BBar
17 from .CSDQ4EF import PlaneStrainQuad4
18 from .CSDQ4ER import PlaneStrainQuad4Reduced
19 from .CSDQ4ES import PlaneStrainQuad4SelectiveReduced
20
21 from .CSDAX4F import AxiSymmetricQuad4
```

```python
from .CSDAX4S import AxiSymmetricQuad4SelectiveReduced
from .CSDAX4R import AxiSymmetricQuad4Reduced

from .CSDQ4SF import PlaneStressQuad4
from .CSDQ4SI import PlaneStressQuad4Incompat

from .CSDQ8EB import PlaneStrainQuad8BBar
from .CSDQ8EF import PlaneStrainQuad8
from .CSDQ8ER import PlaneStrainQuad8Reduced

from .CSDQ8SF import PlaneStressQuad8
```

Chapter 6

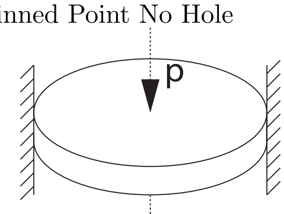
Verification Problems

For solid plates and washers, the constant D is defined as:

$$D = \frac{Et^3}{12(1-\nu^2)}$$

6.1 Solid Plates

Pinned Point No Hole



$$y_{max} = \frac{-pR^2}{16\pi D}\frac{3+\nu}{1+\nu}$$

Fixed Point No Hole

$$y_{max} = \frac{-pR^2}{16\pi D}$$

Pinned Distributed No Hole

$$y_{max} = \frac{-qR^4(5+\nu)}{64D(1+\nu)}$$

Fixed Distributed No Hole

$$y_{max} = \frac{-qrR^4}{64D}$$

6.2 Plate With a Hole (Washers)

Pinned Point Hole

$$y_{max} = \frac{-pR^3}{D}\left(\frac{C_1 L_9}{C_7} - L_3\right)$$

$$C_1 = \frac{1+\nu}{2}\frac{r}{R}\ln\frac{R}{r} + \frac{1-\nu}{4}\left(\frac{R}{r} - \frac{r}{R}\right)$$

$$C_7 = \frac{1}{2}(1-\nu^2)\left(\frac{R}{r} - \frac{r}{R}\right)$$

$$L_3 = \frac{r}{4R}\left(\left(\left(\frac{r}{R}\right)^2 + 1\right)\ln\frac{R}{r} + \left(\frac{r}{R}\right)^2 - 1\right)$$

$$L_9 = \frac{r}{R}\left(\frac{1+\nu}{2}\ln\frac{R}{r} + \frac{1-\nu}{4}\left(1-\left(\frac{r}{R}\right)^2\right)\right)$$

Fixed Point Hole

$$y_{max} = \frac{-pR^3}{D}\left(\frac{C_1 L_6}{C_4} - L_3\right)$$

$$C_1 = \frac{1+\nu}{2}\frac{r}{R}\ln\frac{R}{r} + \frac{1-\nu}{4}\left(\frac{R}{r} - \frac{r}{R}\right)$$

$$C_4 = \frac{1}{2}\left((1+\nu)\frac{r}{R} + (1-\nu)\frac{R}{r}\right)$$

$$L_3 = \frac{r}{4R}\left(\left(\left(\frac{r}{R}\right)^2 + 1\right)\ln\frac{R}{r} + \left(\frac{r}{R}\right)^2 - 1\right)$$

$$L_6 = \frac{r}{4R}\left(\left(\frac{r}{R}\right)^2 - 1 + 2\ln\frac{R}{r}\right)$$

Pinned Distributed Hole

$$y_{max} = \frac{-qR^4}{D}\left(\frac{C_1 L_{17}}{C_7} - L_{11}\right)$$

$$C_1 = \frac{1+\nu}{2}\frac{r}{R}\ln\frac{R}{r} + \frac{1-\nu}{4}\left(\frac{R}{r} - \frac{r}{R}\right)$$

$$C_7 = \frac{1}{2}(1-\nu^2)\left(\frac{R}{r} - \frac{r}{R}\right)$$

$$L_{11} = \frac{1}{64}\left(1 + 4\left(\frac{r}{R}\right)^2 - 5\left(\frac{r}{R}\right)^4 - 4\left(\frac{r}{R}\right)^2\left[2 + \left(\frac{r}{R}\right)^2\right]\ln\frac{r}{R}\right)$$

$$L_{17} = \frac{1}{4}\left(1 - \frac{1-\nu}{4}\left(1-\left(\frac{r}{R}\right)^4\right) - \left(\frac{r}{R}\right)^2\left(1 + (1+\nu)\ln\frac{R}{r}\right)\right)$$

Fixed Distributed Hole

$$y_{max} = \frac{-qR^4}{D}\left(\frac{C_1 L_{14}}{C_4} - L_{11}\right)$$

$$C_1 = \frac{1+\nu}{2}\frac{r}{R}\ln\frac{R}{r} + \frac{1-\nu}{4}\left(\frac{R}{r} - \frac{r}{R}\right)$$

$$C_4 = \frac{1}{2}\left((1+\nu)\frac{r}{R} + (1-\nu)\frac{R}{r}\right)$$

$$L_{11} = \frac{1}{64}\left(1 + 4\left(\frac{r}{R}\right)^2 - 5\left(\frac{r}{R}\right)^4 - 4\left(\frac{r}{R}\right)^2\left[2 + \left(\frac{r}{R}\right)^2\right]\ln\frac{r}{R}\right)$$

$$L_{14} = \frac{1}{16}\left(1 - \left(\frac{r}{R}\right)^4 - 4\left(\frac{r}{R}\right)^2\ln\frac{R}{r}\right)$$

Chapter 7

Performance Evaluation

Chapter 8

Conclusions