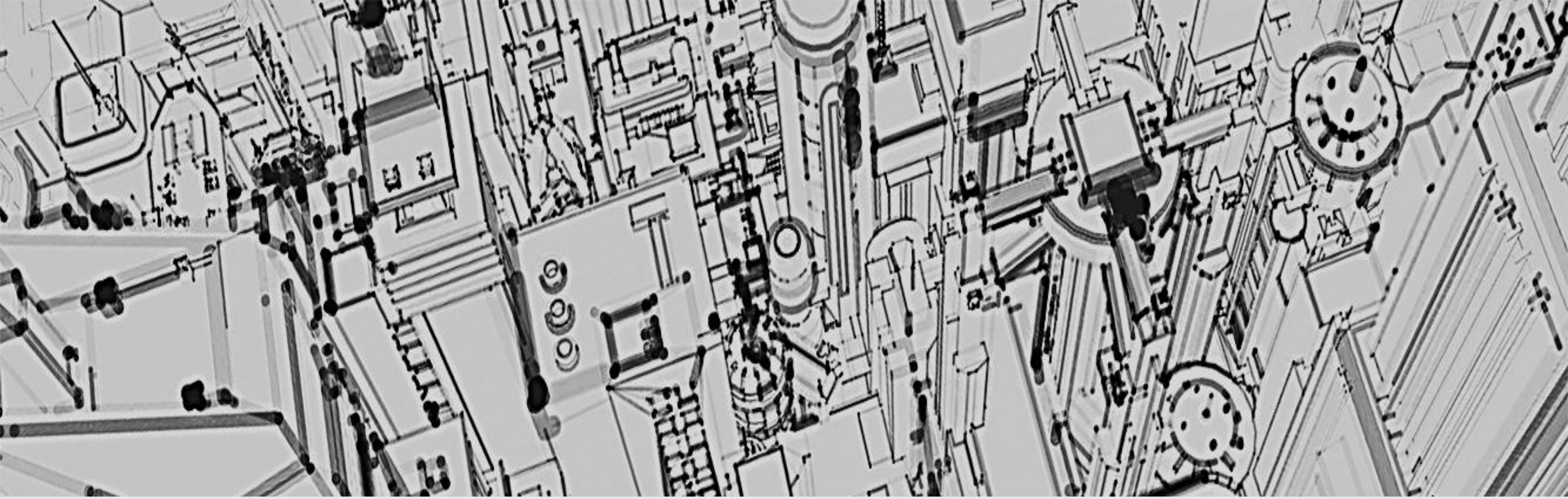
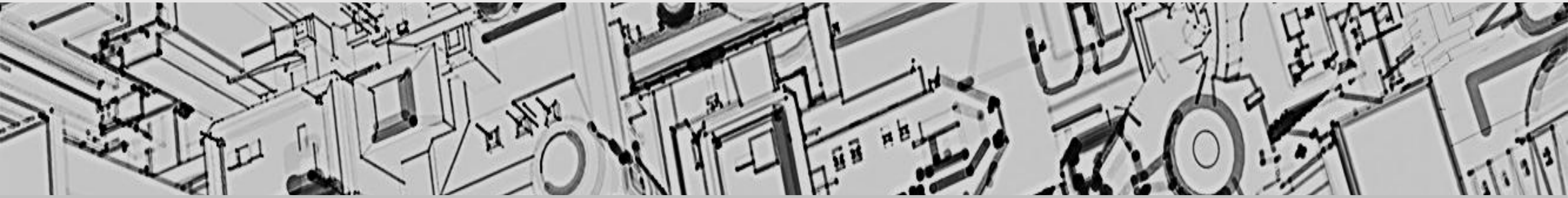


Modeling Concrete Frames

OpenSees Days 2014

Silvia Mazzoni, PhD
Earthquake and Structural Engineering Consultant



Key Modeling Parameters

- Stiffness
- Strength
- Stiffness Loss
- Strength Loss
- Deformability – ductility capacity
- Monotonic vs. Cyclic Response
- Rate of Loading
- Robustness of model

Considerations in Modeling RC Frames

- Materials
  - Confined Concrete
  - Unconfined Concrete
  - Reinforcing Steel
- Sections
  - Elastic Section
  - Uniaxial Section – uncoupled axial & flexure
  - Fiber Section – coupled P-M-M Interaction
- System
  - 2D/3D
  - Rigid/Flexible Diaphragm

- Elements
  - Structural Element
    - Beams – no axial load
    - Columns (P-M Interaction)
  - Plastic-Hinge behavior
    - Confinement
    - Hinge length & growth
    - Yield Penetration
    - Bond Stress/Strength
    - Bar Pull-out, Anchorage loss
    - Bar elongation and buckling
  - Element Type
    - Continuum model
    - Distributed plasticity
    - Lumped plasticity
    - Displacement-based
  - Beam-Column Connections
  - Shear
    - Moment-Shear Interaction
    - Shear-Critical Elements

Modeling Limitations

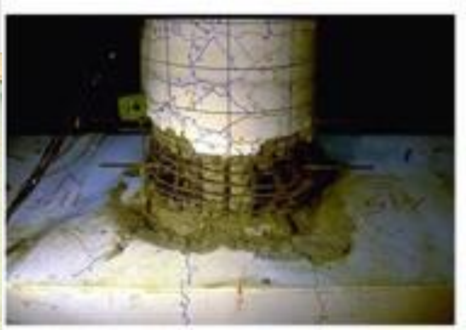
- We are modeling 3D elements with 1-D line elements with a 2D cross section
- Basic assumption: PLANE SECTIONS REMAIN PLANE
- We don't have a full understanding, hence no dependable predictive model on flexure & shear interaction
- For design, we can just design around shear, we have to face it for evaluation of older non-ductile existing RC structures
- For many existing structures we don't know exactly what is within the cover
- Quality-control in construction is good enough
- Material properties over time
  - Concrete strength with curing
  - Creep and Fatigue
  - Corrosion in reinforcement

Understanding Behavior

- Make a personal prediction of expected behavior – use the numerical model to validate your prediction. Iterate until you have agreement.





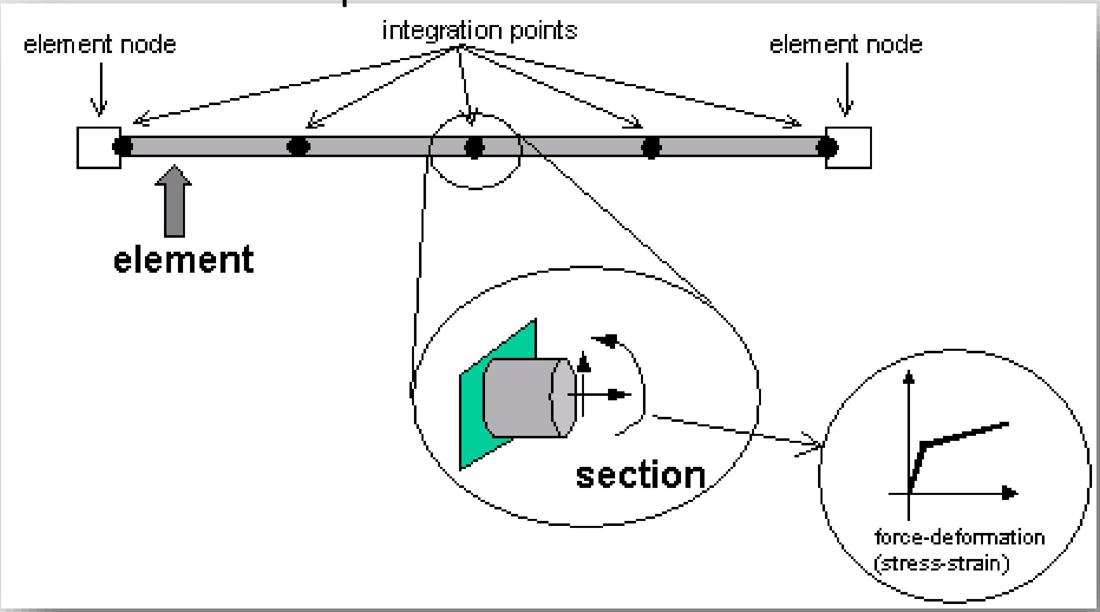





RC Frames: Beam-Column Elements

Element Types

- Distributed plasticity
- Lumped plasticity
- Displacement-based



Sections

- Elastic Section
- Uniaxial Section – uncoupled axial & flexure
- Fiber Section – coupled P-M-M Interaction

Materials

- Confined Concrete
- Unconfined Concrete
- Reinforcing Steel

All these materials define a force-deformation response

Uniaxial Materials

Steel & Reinforcing-Steel Materials

- Steel01 Material
- Steel02 Material
- Hysteretic Material
- Reinforcing Steel Material
- Dodd Restrepo
- RambergOsgoodSteel Material

Concrete Materials

- Concrete01 Material
- Concrete02 Material
- Concrete04 Material
- Concrete06 Material
- Concrete07
- Concrete01 Material With Stuff in the Cracks
- ConfinedConcrete01 Material

Standard Uniaxial Materials

- Elastic Uniaxial Material
- Elastic-Perfectly Plastic Material
- Elastic-Perfectly Plastic Gap Material
- Elastic-No Tension Material
- Parallel Material
- Series Material

Other Uniaxial Materials

- Modified Ibarra-Medina-Krawinkler Deterioration Model with Bilinear Hysteretic Response (Bilin Material)
- Modified Ibarra-Medina-Krawinkler Deterioration Model with Peak-Oriented Hysteretic Response (ModIMKPeakOriented Material)
- Modified Ibarra-Medina-Krawinkler Deterioration Model with Pinched Hysteretic Response (ModIMKPinching Material)
- BARSLIP Material
- Bond_SP01
- Fatigue Material
- Hardening Material
- Hyperbolic Gap Material
- Limit State Material
- MinMax Material
- ElasticBilin Material
- ElasticMultiLinear Material
- MultiLinear Material
- Pinching4 Material
- Engineered Cementitious Composites Material
- SelfCentering Material
- Viscous Material
- BoucWen Material
- BWBN Material (Pinching Hysteretic Bouc-Wen Material)
- Pinching Limit State Material

Uniaxial Materials

Uniaxial materials define a uniaxial force-deformation relationship.

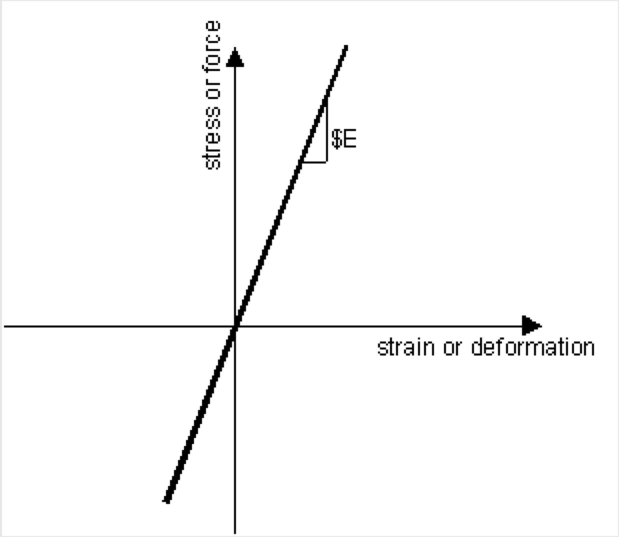
The type of force and deformation depends on where these materials are placed.

- In section fibers, they represent
  - material stress vs. strain
- In uniaxial section, they can represent:
  - force vs. deformation (axial, shear)
  - Bending moment vs. curvature
  - Torsion vs. twist
- In spring elements:
  - Force vs. Deformation
  - Moment vs. Rotation

Elastic Material

uniaxialMaterial Elastic $matTag $E <$eta>

$matTag	unique material object integer tag
$E	tangent
$eta	damping tangent (optional, default=0.0)

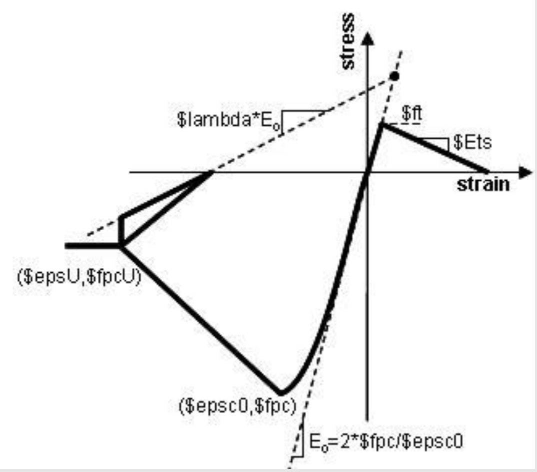


Concrete

- Concrete01 Material -- Zero Tensile Strength
- Concrete02 Material -- Linear Tension Softening
- Concrete04 Material -- Popovics Concrete Material
- Concrete06 Material
- Concrete07 – Chang & Mander’s 1994 Concrete Model
- Concrete01 Material With Stuff in the Cracks
- ConfinedConcrete01 Material

uniaxialMaterial Concrete02 $matTag $fpc $epsc0 $fpcu $epsU $lambda $ft $Ets

$matTag	integer tag identifying material
$fpc	concrete compressive strength at 28 days (compression is negative)*
$epsc0	concrete strain at maximum strength*
$fpcu	concrete crushing strength *
$epsU	concrete strain at crushing strength*
$lambda	ratio between unloading slope at $epscu and initial slope
$ft	tensile strength
$Ets	tension softening stiffness (absolute value) (slope of the linear tension softening branch)

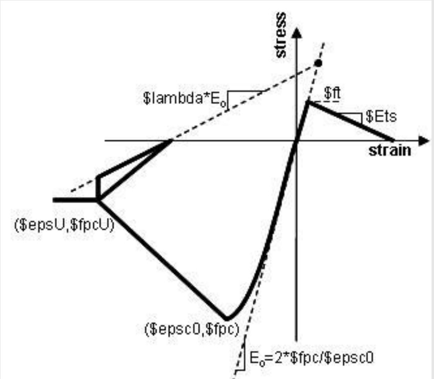


For a large structure with deformations well beyond the elastic limit, Concrete01 may be enough

Confined & Unconfined Concrete

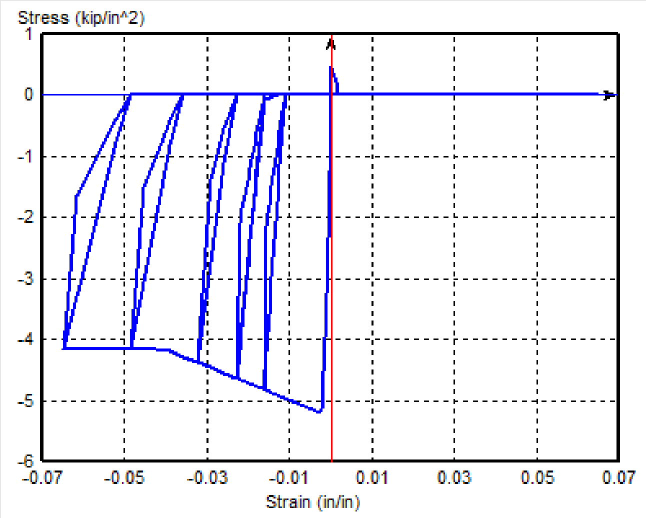
uniaxialMaterial Concrete02 $matTag $fpc $epsc0 $fpcu $epsU $lambda $ft $Ets

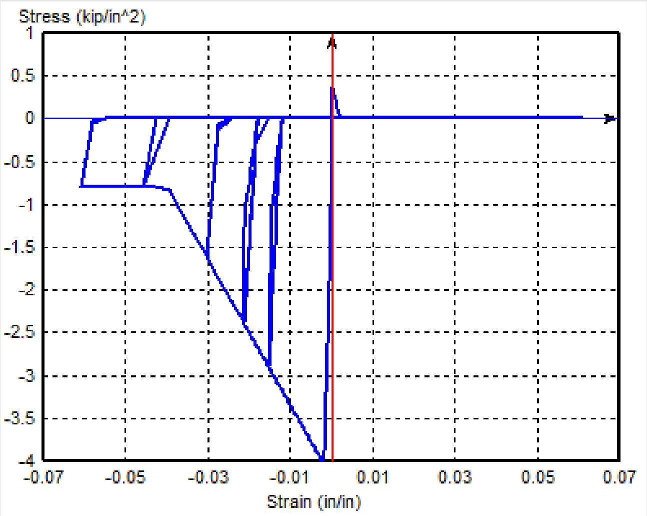
$matTag	integer tag identifying material
$fpc	concrete compressive strength at 28 days (compression is negative)*
$epsc0	concrete strain at maximum strength*
$fpcu	concrete crushing strength *
$epsU	concrete strain at crushing strength*
$lambda	ratio between unloading slope at $epscu and initial slope
$ft	tensile strength
$Ets	tension softening stiffness (absolute value) (slope of the linear tension softening branch)



uniaxialMaterial Concrete02 1 **-5.2** -0.0025 **-4.16** -0.04 0.1 0.52 260.0

uniaxialMaterial Concrete02 2 **-4.0** -0.0022 **-0.8** -0.04 0.1 0.4 200.0

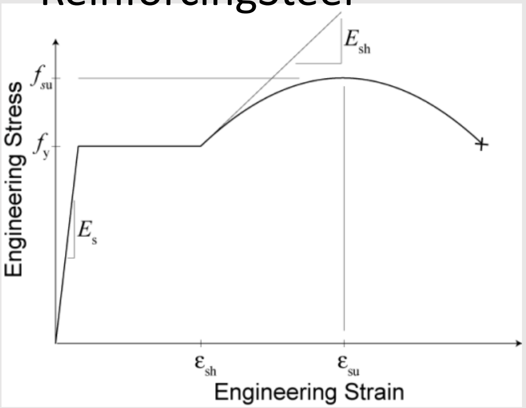




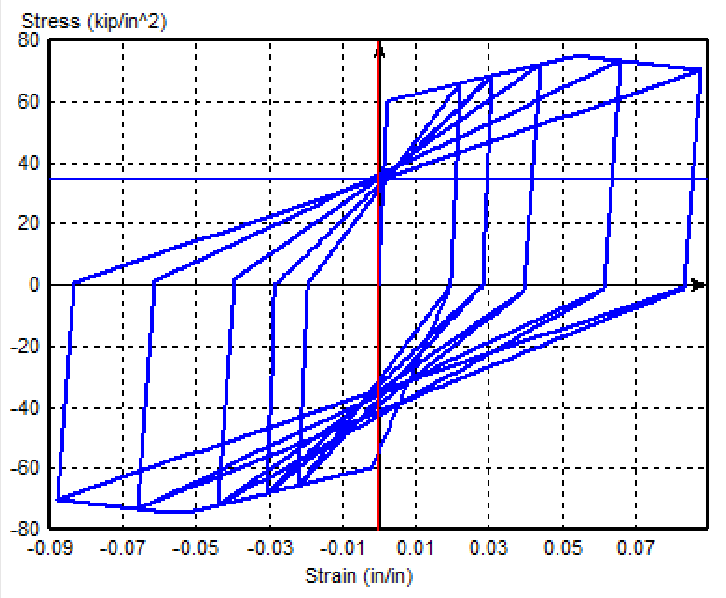
Reinforcing Steel Materials

- Steel01 Material
- Steel02 Material
- Hysteretic Material
- Reinforcing Steel Material
- Dodd Restrepo
- RambergOsgoodSteel Material

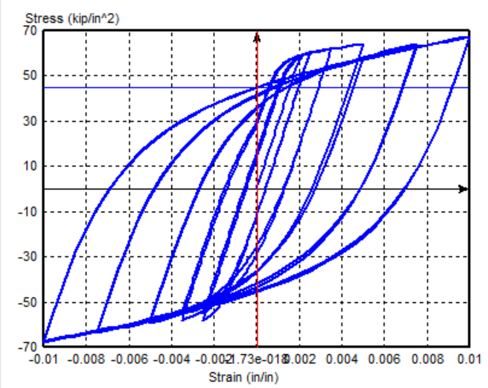
ReinforcingSteel



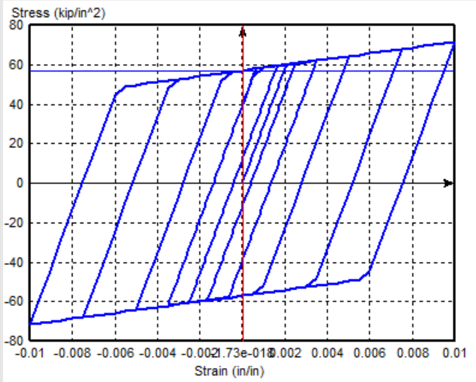
Hysteretic Material



Steel02 Material

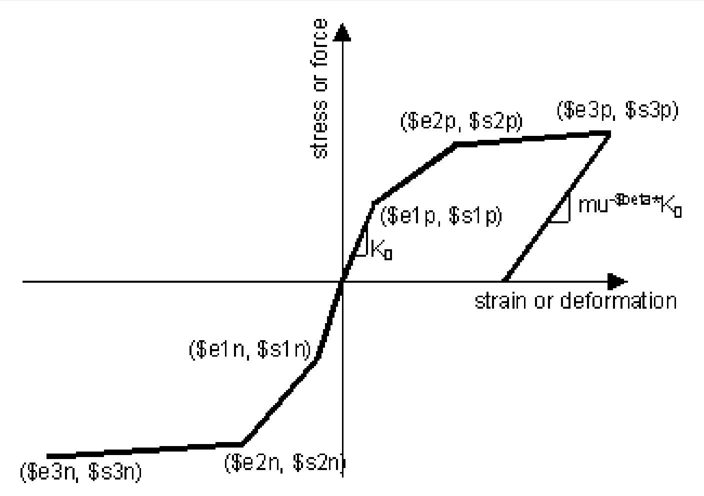


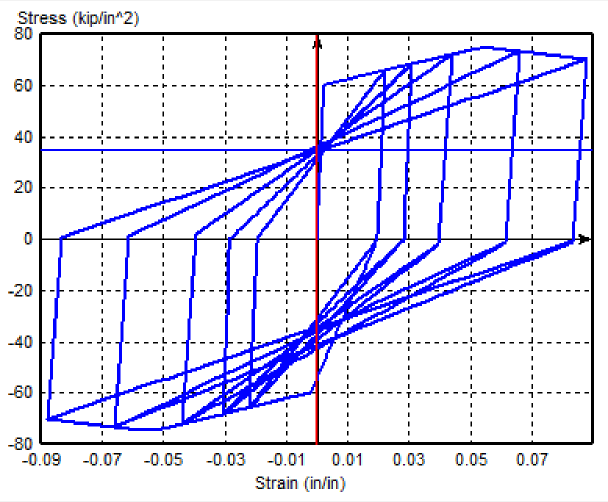
Steel01



Hysteretic Material

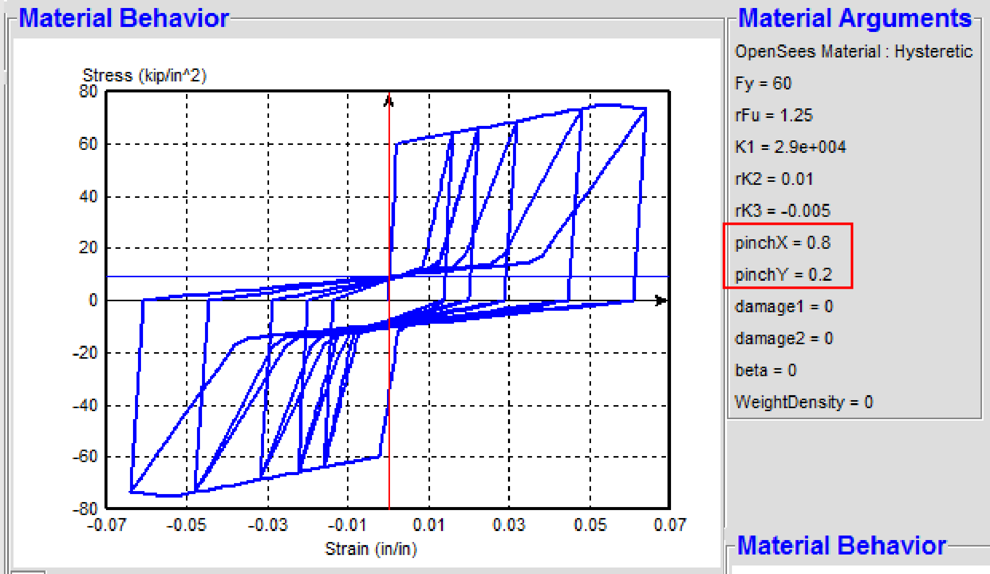
uniaxialMaterial Hysteretic $matTag $s1p $e1p $s2p $e2p <$s3p $e3p> $s1n $e1n $s2n $e2n <$s3n $e3n> $pinchX $pinchY $damage1 $damage2 <$beta>

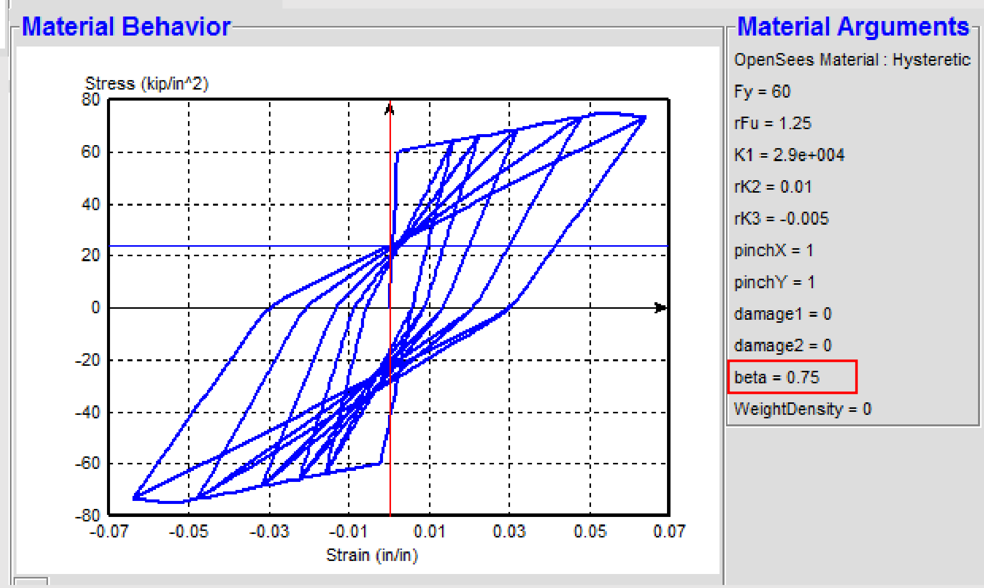




- Being able to have a negative tangent stiffness at large deformations is useful
- Do not have to have symmetry in tension and compression
- This material can also be used in defining section moment-curvature behavior

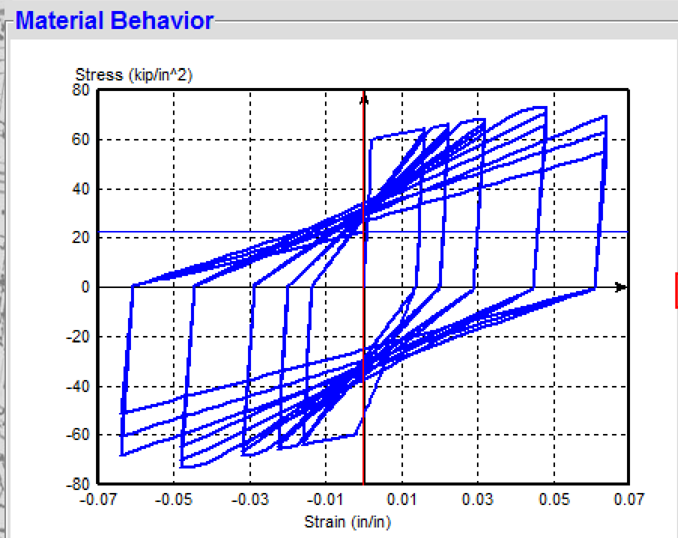
Hysteretic-Material Options





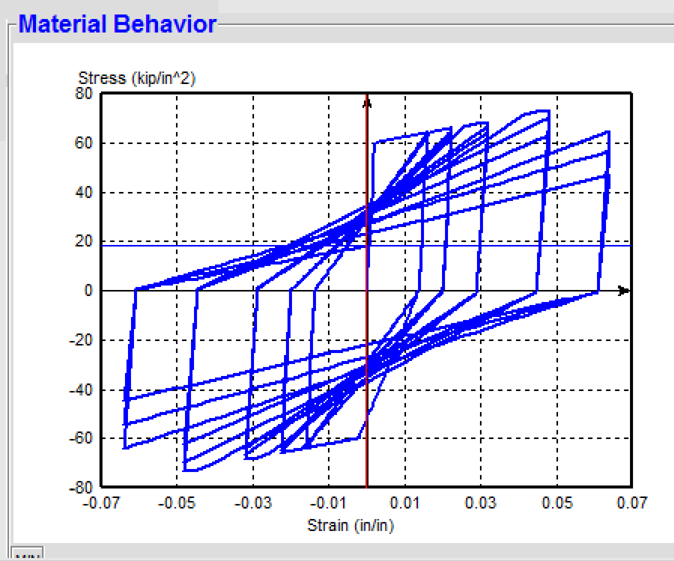
Hysteretic Material Damage Parameters

$damage1	damage due to ductility: D1(mu-1)
$damage2	damage due to energy: D2(Eii/Eult)



Material Arguments

OpenSees Material : Hysteretic
Fy = 60
rFu = 1.25
K1 = 2.9e+004
rK2 = 0.01
rK3 = -0.005
pinchX = 1
pinchY = 1
damage1 = 0.005
damage2 = 0
beta = 0
WeightDensity = 0



Material Arguments

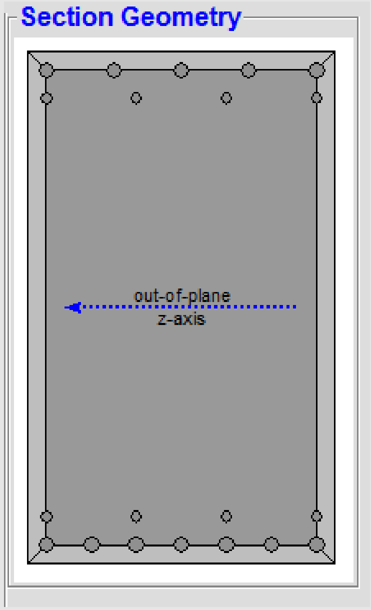
OpenSees Material : Hysteretic
Fy = 60
rFu = 1.25
K1 = 2.9e+004
rK2 = 0.01
rK3 = -0.005
pinchX = 1
pinchY = 1
damage1 = 0
damage2 = 0.2
beta = 0
WeightDensity = 0

Beam-Column Sections

- Elastic Section
  - linear-elastic moment-curvature relationship
- Uniaxial Section
  - user-defined moment-curvature relationship (use unaxialMaterial)
  - uncoupled P-M and anything else
- Fiber Section
  - user-defined section geometry/materials via fibers
  - coupled P-M interaction
  - coupled bi-directional response
- Section Aggregator
  - combine all uncoupled responses (e.g., Uniaxial flexure + Uniaxial Axial, Fiber flexure/axial + shear)

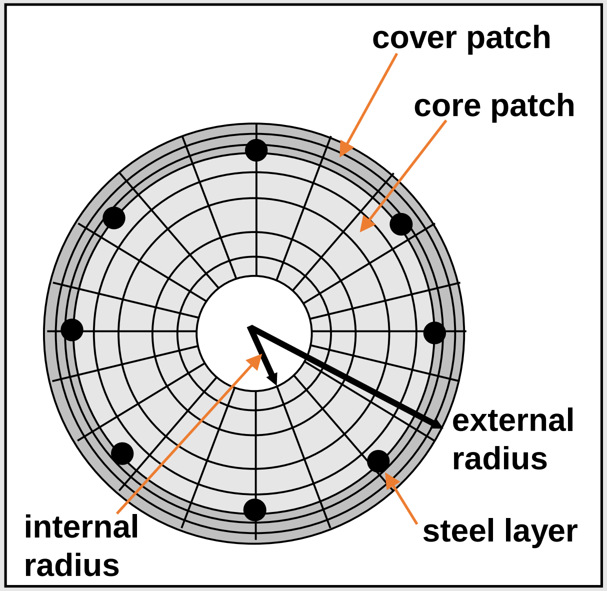
Fiber Section

```
section Fiber $secTag {
        fiber <fiber arguments>
        patch <patch arguments>
        layer <layer arguments>
}
```



- User-defined section geometry/materials via fibers
- Coupled P-M interaction
- Coupled bi-directional response
- No shear
- Do not have to compute moment-curvature *a-priori*

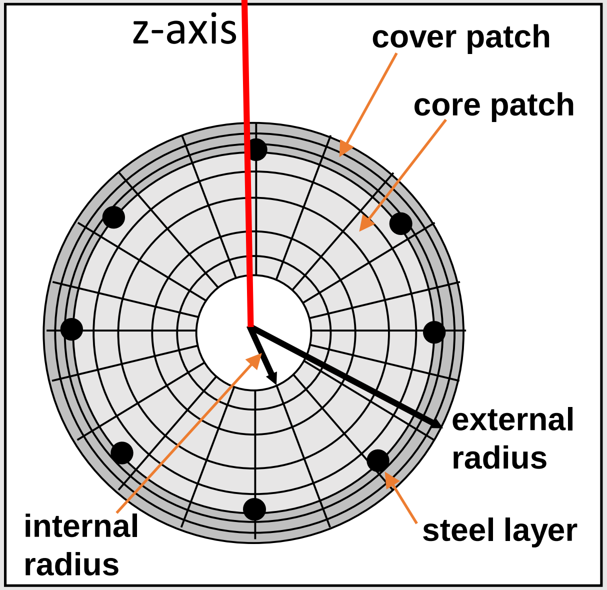
Note: You cannot assign this type of section directly to a beam-column element, you need to define behavior for all modes of deformation (axial, bending & shear, for all ndm).



Fiber Section -- fiber

```
section Fiber $secTag {
        fiber <fiber arguments>
        patch <patch arguments>
        layer <layer arguments>
}
```

fiber $yLoc $zLoc $A $matTag

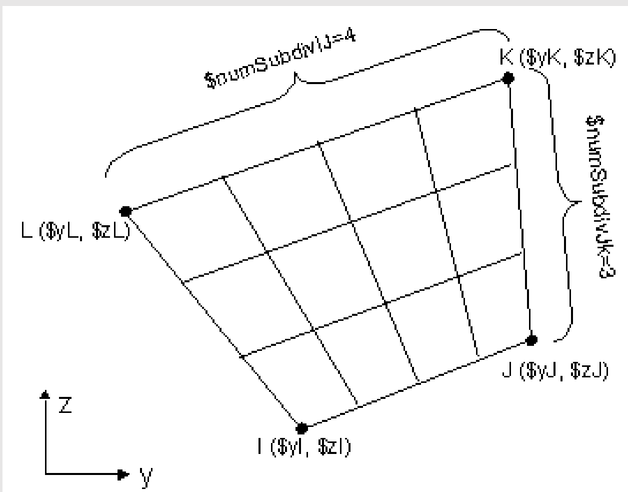


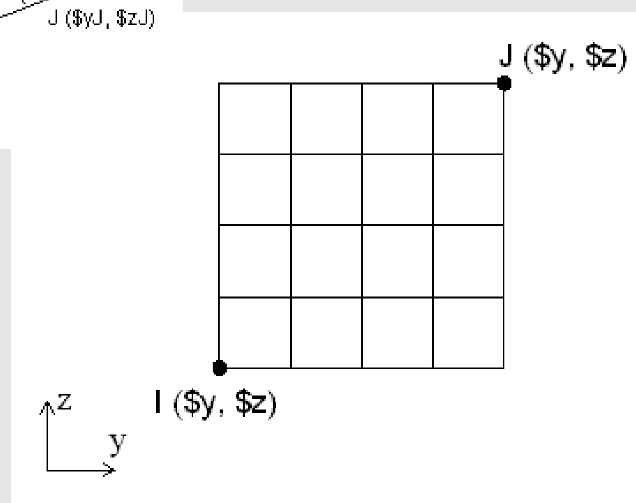
$yLoc	y coordinate of the fiber in the section (local coordinate system)
$zLoc	z coordinate of the fiber in the section (local coordinate system)
$A	area of fiber
$matTag	material tag of the pre-defined uniaxialMaterial object used to represent the stress-strain for the area of the fiber

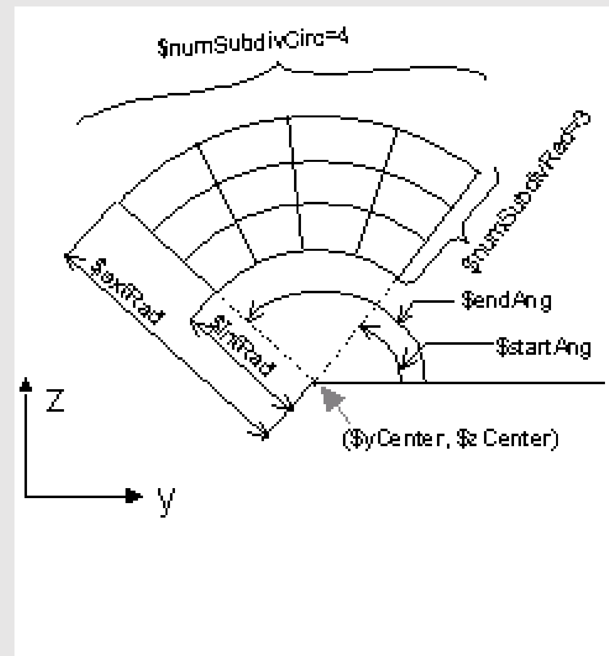
Fiber Section -- patch

```
section Fiber $secTag {
        fiber <fiber arguments>
        patch <patch arguments>
        layer <layer arguments>
}
```

patch rect $matTag $numSubdivY $numSubdivZ $yI $zI $yJ $zJ
patch quad $matTag $numSubdivIJ $numSubdivJK $yI $zI $yJ $zJ $yK $zK $yL $zL
patch circ $matTag $numSubdivCirc $numSubdivRad $yCenter $zCenter $intRad $extRad <$startAng $endAng>



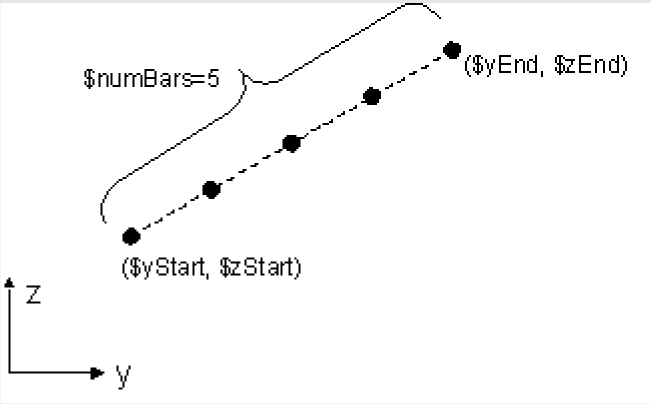


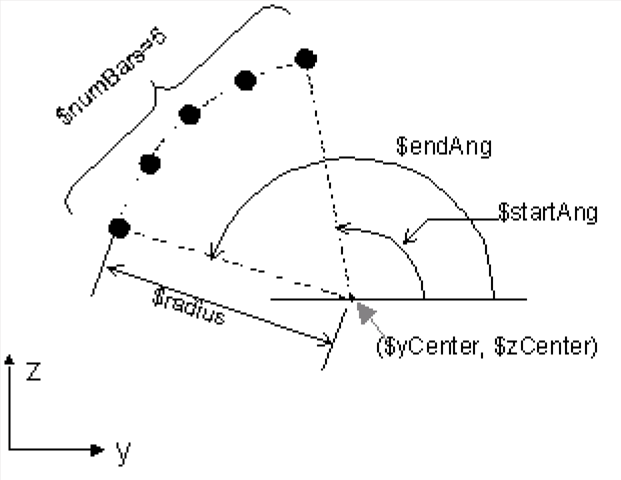


Fiber Section -- layer

```
section Fiber $secTag {
        fiber <fiber arguments>
        patch <patch arguments>
        layer <layer arguments>
}
```

layer straight $matTag $numFiber $areaFiber $yStart $zStart $yEnd $zEnd
layer circ $matTag $numFiber $areaFiber $yCenter $zCenter $radius <$startAng $endAng>





Tcl procedure

A procedure is a function in tcl.

Define procs for a repeated series of commands on a fixed set of input variables

Application: create a procedure to define a fiber section

User: create a library of procedures to use on different projects

Define:

```
proc procName {input variables} {
        … series of commands
}
```

Execute:

procName (input variables)

Example:

```
proc multiply {a b} {
    set c [expr $a*$b]
    return $c
}
set a 3; set b 5
set result [multiply $a $b]
```

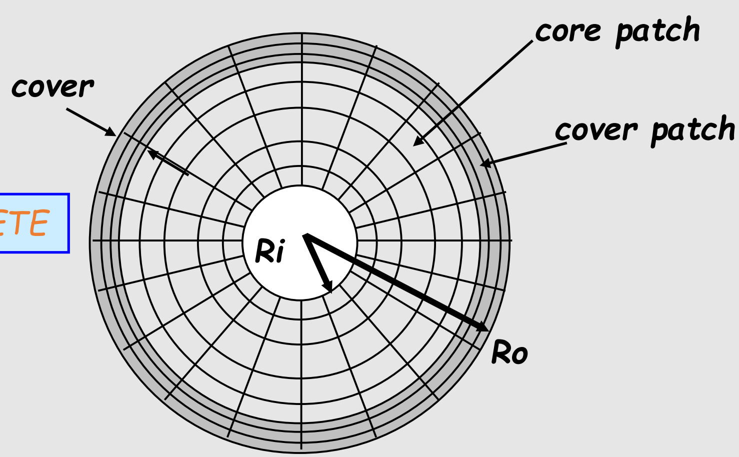
Tcl proc: define rc section

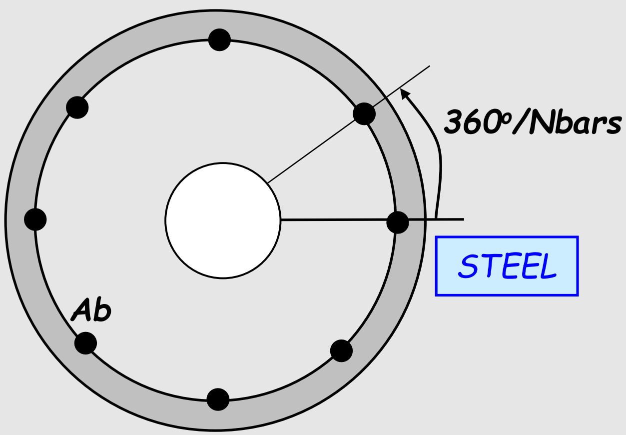
```
proc RCcircSection {id  Ri Ro cover coreID coverID steelID  Nbars Ab  nfCoreR nfCoreT nfCoverR nfCoverT} {
        section fiberSec $id {
                set Rc [expr $Ro-$cover];                                      # Core radius
                patch circ $coreID $nfCoreT $nfCoreR 0 0 $Ri $Rc 0 360;        # Define the core patch
                patch circ $coverID $nfCoverT $nfCoverR 0 0 $Rc $Ro 0 360;     # Define the cover patch

                if {$Nbars<= 0} { return }
                set theta [expr 360.0/$Nbars];                        # angle increment between bars
                layer circ $steelID $Nbars $Ab 0 0 $Rc $theta 360;    # Define the reinforcing layer
        }
}
```

CONCRETE

STEEL

CONCRETE





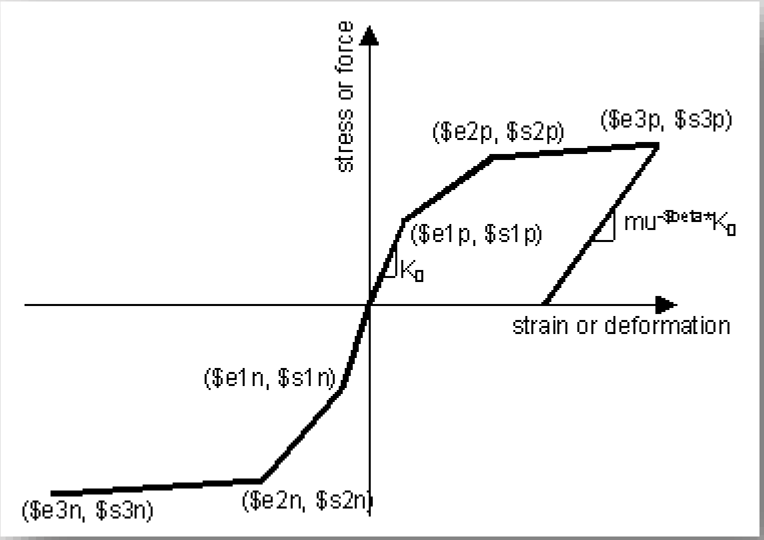
STEEL

Call:

RCcircSection $id $Ri $Ro $cover $coreID $coverID $steelID $Nbars $Ab $nfCoreR $nfCoreT $nfCoverR $nfCoverT

Uniaxial Section

- Use uniaxialMaterial to define section **moment-curvature** relationship
  - E.g. Hysteretic Material



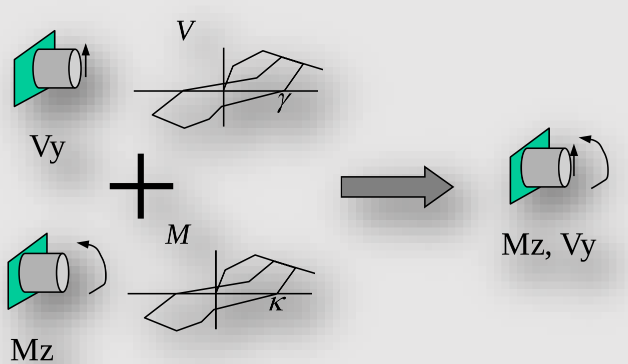
Note: You cannot assign this type of section directly to a beam-column element, you need to define behavior for all modes of deformation (axial, bending & shear, for all ndm)

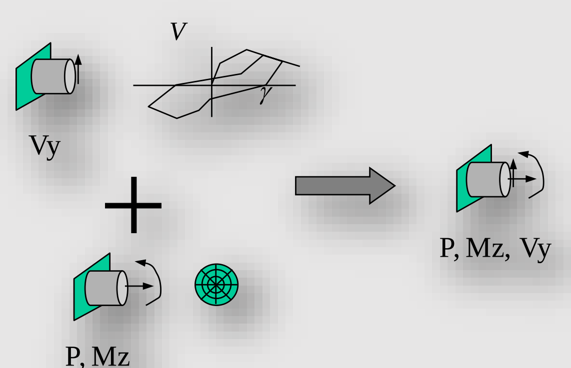
Section Aggregator

- groups previously-defined section objects (or uniaxialMaterials) into a single section force-deformation model which defines all modes of deformation
- This is the section that gets assigned to the beam-column element

```
section Aggregator $secTag $matTag1 $string1 $matTag2 $string2 ....... <-section $sectionTag>
```

$secTag		unique section object integer tag
$matTag1, $matTag2 ...		previously-defined uniaxialMaterial objects
$string1, $string2		the force-deformation quantities corresponding to each section object. One of the following strings is used:
	P	Axial force-deformation
	Mz	Moment-curvature about section local z-axis
	Vy	Shear force-deformation along section local y-axis
	My	Moment-curvature about section local y-axis
	Vz	Shear force-deformation along section local z-axis
	T	Torsion Force-Deformation
<-section $sectionTag>		specifies a previously-defined Section object (identified by the argument $sectionTag) to which these uniaxialMaterial objects may be added to recursively define a new Section object





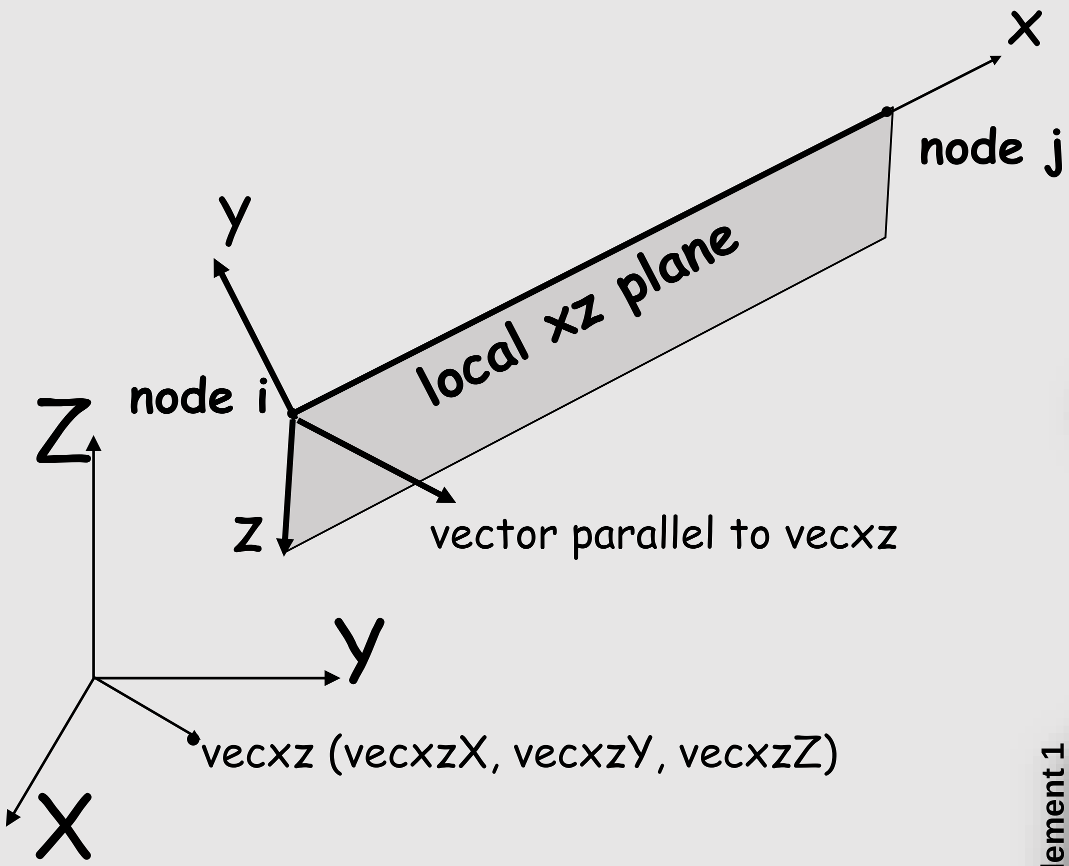
Geometric transformation

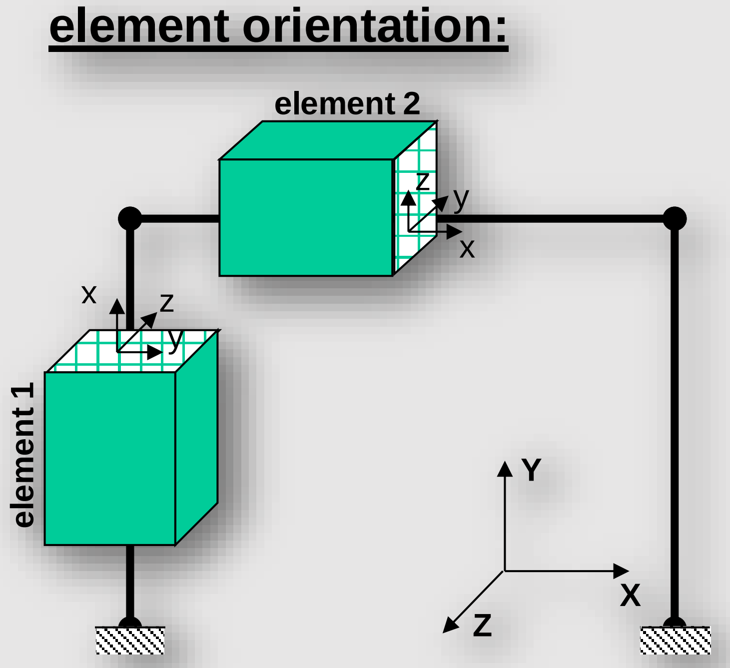
- Performs a linear geometric transformation of beam stiffness and resisting force from the basic system to the global-coordinate system
- Defines rigid offsets
- Defines Geometric nonlinearity

```
geomTransf $Type $transfTag $vecxzX $vecxzY $vecxzZ
                                        <-jntOffset $dXi $dYi $dZi $dXj $dYj $dZj>
```

$Type	Transformation Type: Linear, PDelta or Corotational
$transfTag	unique identifier for CrdTransf object
$vecxzX $vecxzY $vecxzZ	ONLY IN 3D. X, Y, and Z components of vecxz, the vector used to define the local x-z plane of the local-coordinate system. The local y-axis is defined by taking the cross product of the x-axis and the vecxz vector. These components are specified in the global-coordinate system X,Y,Z and define a vector that is in a plane parallel to the x-z plane of the local-coordinate system. These items need to be specified for the three-dimensional problem.
$dXi $dYi $dZi	joint offset values -- absolute offsets specified with respect to the global coordinate system for element-end node i (the number of arguments depends on the dimensions of the current model) (optional)
$dXj $dYj $dZj	joint offset values -- absolute offsets specified with respect to the global coordinate system for element-end node j (the number of arguments depends on the dimensions of the current model) (optional)

Local & Global Coordinate Systems





Beam-Column Elements

- Elastic Beam Column Element
- Elastic Beam Column Element with Stiffness Modifiers
- Beam With Hinges Element
- Displacement-Based Beam-Column Element
- Force-Based Beam-Column Element
- Flexure-Shear Interaction Displacement-Based Beam-Column Element

Elastic Beam-Column Element

2D

```
element elasticBeamColumn $eleTag $iNode $jNode $A $E $Iz $transfTag
```

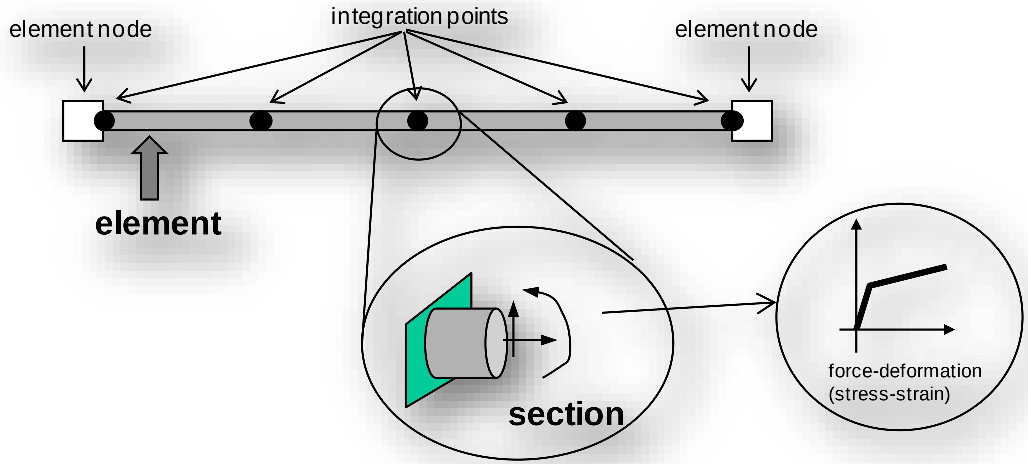
3D

```
element elasticBeamColumn $eleTag $iNode $jNode $A $E $G $J $Iy $Iz $transfTag
```

Force-Based Beam-Column Element

element forceBeamColumn $eleTag $iNode $jNode $transfTag "IntegrationType arg1 arg2 ..."
<-mass $massDens> <-iter $maxIters $tol>

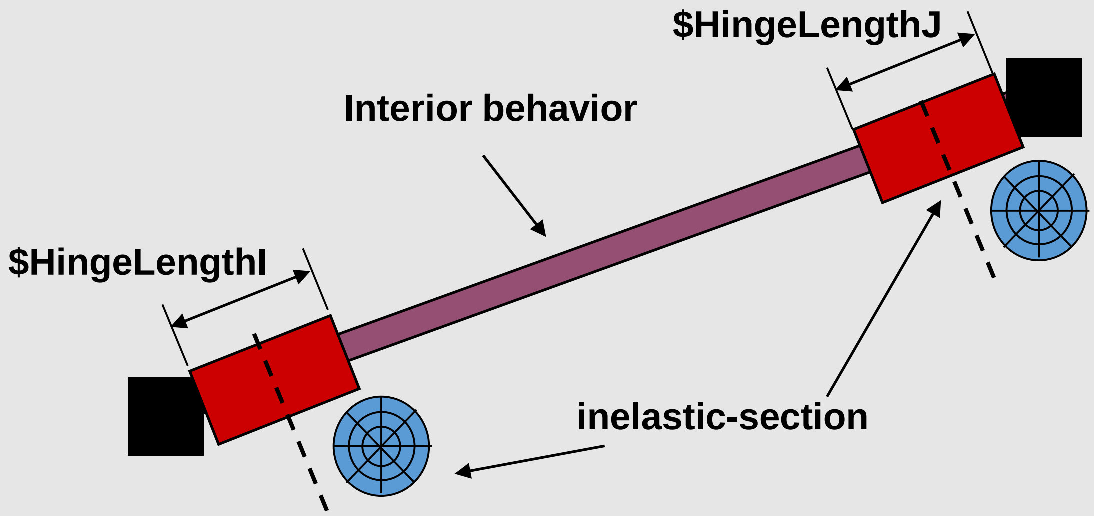
$eleTag	unique element object tag
$iNode $jNode	end nodes
$transfTag	identifier for previously-defined coordinate-transformation (CrdTransf) object
IntegrationType arg1 arg2 ...	specifies locations and weights of integration points and their associated section force-deformation models (see [File:IntegrationTypes.pdf])
$massDens	element mass density (per unit length), from which a lumped-mass matrix is formed (optional, default=0.0)
$maxIters	maximum number of iterations to undertake to satisfy element compatibility (optional, default=10)
$tol	tolerance for satisfaction of element compatibility (optional, default=10-12)



BeamWithHinges Element

•The original implementation constrained the interior behavior to be elastic.
•Plasticity can now spread beyond the plastic hinge regions
•Hinges can form on the element interior, e.g., due to distributed member loads

element forceBeamColumn $eleTag $iNode $jNode $transfTag "HingeRadau $secTagI $LpI $secTagJ $LpJ $secTagInterior" <-mass $massDens> <-iter $maxIters $tol>



Example

```
# ------------------------------------------------------------------------------------------
# Example4. 2D Portal Frame--  Build Model
# nonlinearBeamColumn element, inelastic fiber section
#     Silvia Mazzoni & Frank McKenna, 2006
#
#    ^Y
#    |
#    3__________(3)_________4      __
#    |                      |       |
#    |                      |       |
#    |                      |       |
#  (1)                     (2)     LCol
#    |                      |       |
#    |                      |       |
#    |                      |       |
#  =1=                     =2=    _|_  -------->X
#    |----------LBeam------------|
#

# SET UP ----------------------------------------------------------------------------
wipe;                                # clear memory of all past model definitions
model BasicBuilder -ndm 2 -ndf 3;    # Define the model builder, ndm=#dimension, ndf=#dofs
set dataDir Data;                    # set up name of data directory
file mkdir $dataDir;                 # create data directory
set GMdir "GMfiles";                 # ground-motion file directory
source LibUnits.tcl;                 # define basic and system units
```

Define units

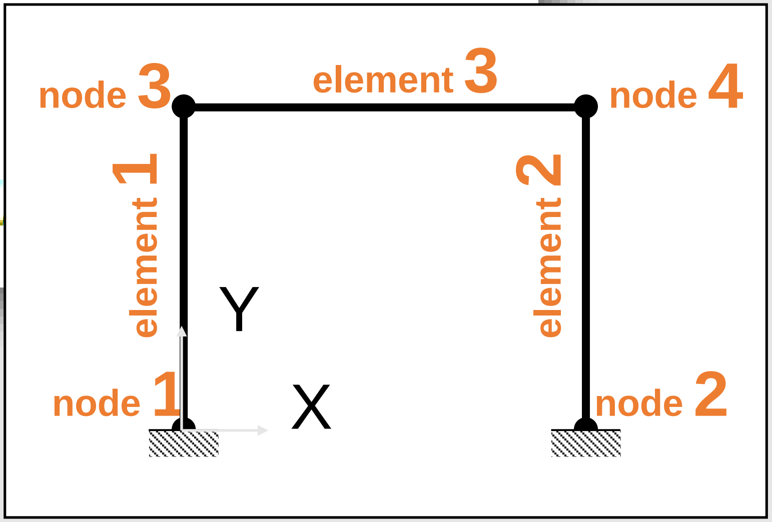
```
# define UNITS ----------------------------------------------------------------
set in 1.;                                  # define basic units -- output units
set kip 1.;                                 # define basic units -- output units
set sec 1.;                                 # define basic units -- output units
set LunitTXT "inch";                        # define basic-unit text for output
set FunitTXT "kip";                         # define basic-unit text for output
set TunitTXT "sec";                         # define basic-unit text for output
set ft [expr 12.*$in];                      # define engineering units
set ksi [expr $kip/pow($in,2)];
set psi [expr $ksi/1000.];
set lbf [expr $psi*$in*$in];                # pounds force
set pcf [expr $lbf/pow($ft,3)];             # pounds per cubic foot
set psf [expr $lbf/pow($ft,3)];             # pounds per square foot
set in2 [expr $in*$in];                     # inch^2
set in4 [expr $in*$in*$in*$in];             # inch^4
set cm [expr $in/2.54];                     # centimeter
set PI [expr 2*asin(1.0)];                  # define constants
set g [expr 32.2*$ft/pow($sec,2)];          # gravitational acceleration
set Ubig 1.e10;                             # a really large number
set Usmall [expr 1/$Ubig];                  # a really small number
```

Example – Geometry, Weight

```
# define GEOMETRY -------------------------------------------------------------
set LCol [expr 36*$ft];                 # column length
set LBeam [expr 42*$ft];                # beam length
# define section geometry
set HCol [expr 5.*$ft];                 # Column Depth
set BCol [expr 5.*$ft];                 # Column Width
set HBeam [expr 8.*$ft];                # Beam Depth
set BBeam [expr 5.*$ft];                # Beam Width
# superstructure weight
set Weight [expr 2000.*$kip];

# calculated parameters
set PCol [expr $Weight/2];                                   # nodal dead-load weigh
set Mass [expr $PCol/$g];                                    # nodal mass
set MCol [expr 1./12.*($Weight/$LBeam)*pow($LBeam,2)];       # beam-end moment due
# calculated geometry parameters
set ACol [expr $BCol*$HCol];                                 # cross-sectional area
set ABeam [expr $BBeam*$HBeam];
set IzCol [expr 1./12.*$BCol*pow($HCol,3)];                  # Column moment of iner
set IzBeam [expr 1./12.*$BBeam*pow($HBeam,3)];               # Beam moment of inert
```

Example: Nodes, BC, Mass

```
# nodal coordinates:
node 1 0 0;            # node#, X, Y
node 2 $LBeam 0
node 3 0 $LCol
node 4 $LBeam $LCol
# Single point constraints -- Boundary Conditions
fix 1 1 1 0;           # node DX DY RZ
fix 2 1 1 0;           # node DX DY RZ
fix 3 0 0 0
fix 4 0 0 0
# nodal masses:
mass 3 $Mass 0. 0.;       # node#, Mx M
mass 4 $Mass 0. 0.
```

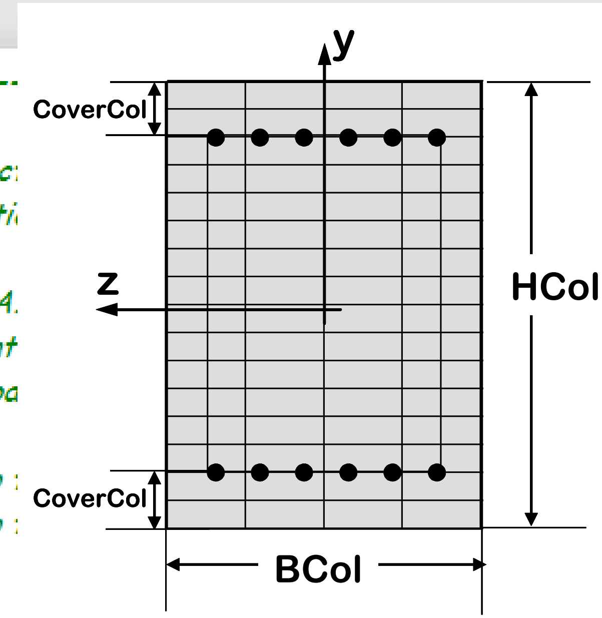


Example: Materials

```
# MATERIAL parameters ---------------------------------------------------------------
set IDconcU 1;                          # material ID tag -- unconfined cover concrete
set IDreinf 2;                          # material ID tag -- reinforcement
# nominal concrete compressive strength
set fc [expr -4.0*$ksi];                # CONCRETE Compressive Strength, ksi   (+Tension, -Compression)
set Ec [expr 57*$ksi*sqrt(-$fc/$psi)];  # Concrete Elastic Modulus
# unconfined concrete
set fc1U $fc;                           # UNCONFINED concrete (todeschini parabolic model), maximum stress
set eps1U -0.003;                       # strain at maximum strength of unconfined concrete
set fc2U [expr 0.2*$fc1U];              # ultimate stress
set eps2U  -0.05;                       # strain at ultimate stress
set lambda 0.1;                         # ratio between unloading slope at $eps2 and initial slope $Ec
# tensile-strength properties
set ftU [expr -0.14*$fc1U];             # tensile strength +tension
set Ets [expr $ftU/0.002];              # tension softening stiffness
# -----------
set Fy [expr 66.8*$ksi];                # STEEL yield stress
set Es [expr 29000.*$ksi];              # modulus of steel
set Bs 0.01;                            # strain-hardening ratio
set R0 18;                              # control the transition from elastic to plastic branches
set cR1 0.925;                          # control the transition from elastic to plastic branches
set cR2 0.15;                           # control the transition from elastic to plastic branches

uniaxialMaterial Concrete02 $IDconcU $fc1U $eps1U $fc2U $eps2U $lambda $ftU $Ets;  # build cover concrete
uniaxialMaterial Steel02 $IDreinf $Fy $Es $Bs $R0 $cR1 $cR2;        # build reinforcement material
```

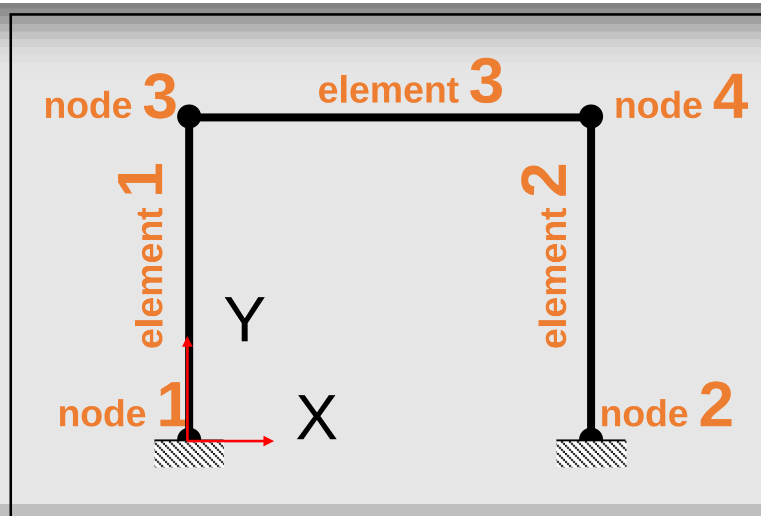
Example: Column & Beam Sections

```
# Define ELEMENTS & SECTIONS -----------------------------------------
# symmetric section
set ColSecTag 1;                    # assign a tag number to the column sec
set BeamSecTag 2;                   # assign a tag number to the beam secti
# define section geometry
set coverCol [expr 6.*$in];         # Column cover to reinforcing steel NA.
set numBarsCol 10;                  # number of longitudinal-reinforcement
set barAreaCol [expr 2.25*$in2];    # area of longitudinal-reinforcement ba
# RC section:
   set coverY [expr $HCol/2.0];  # The distance from the section z-axis to
   set coverZ [expr $BCol/2.0];  # The distance from the section y-axis to
   set coreY [expr $coverY-$coverCol]
   set coreZ [expr $coverZ-$coverCol]
   set nfY 16;           # number of fibers for concrete in y-direction
   set nfZ 4;         # number of fibers for concrete in z-direction
   section fiberSec $ColSecTag   {;     # Define the fiber section
    patch quadr $IDconcU $nfZ $nfY -$coverY $coverZ -$coverY -$coverZ $coverY -$coverZ $coverY $coverZ;
    layer straight $IDreinf $numBarsCol $barAreaCol -$coreY $coreZ -$coreY -$coreZ;   # top layer reinfocement
    layer straight $IDreinf $numBarsCol $barAreaCol  $coreY $coreZ  $coreY -$coreZ; # bottom layer reinforcement
    };  # end of fibersection definition

# BEAM section:
section Elastic $BeamSecTag   $Ec $ABeam $IzBeam;     # elastic beam section
```



Example: Transformation & Elements

```
# define geometric transformation: performs a linear geometric transformation of
#      beam stiffness and resisting force from the basic system to the global-coordinate system
set ColTransfTag 1;           # associate a tag to column transformation
set BeamTransfTag 2;          # associate a tag to beam transformation (good practice to keep col and beam separate)
set ColTransfType Linear ;            # options, Linear PDelta Corotational
geomTransf $ColTransfType $ColTransfTag ;     # only columns can have PDelta effects (gravity effects)
geomTransf Linear $BeamTransfTag  ;

# element connectivity:
set numIntgrPts 5;                              # number of integration points for force-based element
element nonlinearBeamColumn 1 1 3 $numIntgrPts $ColSecTag $ColTransfTag;  # self-explanatory when using variables
element nonlinearBeamColumn 2 2 4 $numIntgrPts $ColSecTag $ColTransfTag;
element nonlinearBeamColumn 3 3 4 $numIntgrPts $BeamSecTag $BeamTransfTag;
```



Example: Recorders

```
# Define RECORDERS -------------------------------------------------------------
recorder Node -file $dataDir/DFree.out -time -node 3 4 -dof 1 2 3 disp;            # displacements of free node
recorder Node -file $dataDir/DBase.out -time -node 1 2 -dof 1 2 3 disp;            # displacements of support nod
recorder Node -file $dataDir/RBase.out -time -node 1 2 -dof 1 2 3 reaction;         # support reaction
recorder Drift -file $dataDir/Drift.out -time -iNode 1 2 -jNode 3 4 -dof 1  -perpDirn 2 ;       # lateral drift
recorder Element -file $dataDir/FCol.out -time -ele 1 2 globalForce;               # element forces -- column
recorder Element -file $dataDir/FBeam.out -time -ele 3 globalForce;                # element forces -- beam
recorder Element -file $dataDir/ForceColSec1.out -time -ele 1 2 section 1 force;          # Column section forces, c
recorder Element -file $dataDir/DefoColSec1.out -time -ele 1 2 section 1 deformation;          # section deforma
recorder Element -file $dataDir/ForceColSec$numIntgrPts.out -time -ele 1 2 section $numIntgrPts force;      # se
recorder Element -file $dataDir/DefoColSec$numIntgrPts.out -time -ele 1 2 section $numIntgrPts deformation;
recorder Element -file $dataDir/ForceBeamSec1.out -time -ele 3 section 1 force;          # Beam section forces,
recorder Element -file $dataDir/DefoBeamSec1.out -time -ele 3 section 1 deformation;          # section deforma
recorder Element -file $dataDir/ForceBeamSec$numIntgrPts.out -time -ele 3 section $numIntgrPts force;      # se
recorder Element -file $dataDir/DefoBeamSec$numIntgrPts.out -time -ele 3 section $numIntgrPts deformation;    #
```

Considerations in Modeling RC Frames

- Materials
  - Confined Concrete
  - Unconfined Concrete
  - Reinforcing Steel
- Sections
  - Elastic Section
  - Uniaxial Section – uncoupled axial & flexure
  - Fiber Section – coupled P-M-M Interaction
- System
  - 2D/3D
  - Rigid/Flexible Diaphragm
- Elements
  - Structural Element
    - Beams – no axial load
    - Columns (P-M Interaction)
  - Plastic-Hinge behavior
    - Confinement
    - Hinge length & growth
    - Yield Penetration
    - Bond Stress/Strength
    - Bar Pull-out, Anchorage loss
    - Bar elongation and buckling
  - Element Type
    - Continuum model
    - Distributed plasticity
    - Lumped plasticity
    - Displacement-based
  - Beam-Column Connections
  - Shear
    - Moment-Shear Interaction
    - Shear-Critical Elements

Questions?