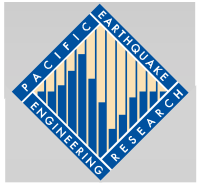
OpenSees

How to Build a Model of a Structure Faster with a Script than with a GUI

Frank McKenna
UC Berkeley

svn co svn://opensees.berkeley.edu:/usr/local/svn/OpenSees/trunk/Workshops/OpenSeesDays/Steel2dModels Steel2dModels

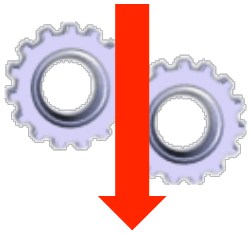


Outline of Seminar

- Introduction
- Moment Frame Example 1
- Moment Frame Example 2
- Braced Frame Examples
- Conclusions

How Do You Interact With OpenSees?

main.tcl



output

Each OpenSees Script You Write IS A PROGRAM

To Build Models in OpenSees faster (and with less errors) than you can in a GUI you simply have to START PROGRAMMING!

An Issue With How Users Program in OpenSee: When should you Use Variables In a Program?

- **NOT to document the commands**

(USE comments instead: i.e. more '#' and less 'set')

- When you have a variable that you might change at some point later in time or in the script, or for use in a mathematical expression.

Truss example using variables:

```
model Basic -ndm 2 -ndf 2

set xCrd1 0.0
set xCrd2 144.0
set xCrd3 168.0
set xCrd4  62.0
set yCrd1 0
set yCrd2 96
set matTag1 1
set timeSeriesTag1
set patternTag1
set E1 3000
set nodeLoadTag 4
set nodeLoadx 100.
set nodeLoadY  -50;
set A1 10
set A2 5.0
```

```
node 1 $xCrd1 $yCrd1
node 2 $xCrd2 $yCrd1
node 3 $xCrd3 $yCrd1
node 4 $xCrd4 $yCrd2

fix 1 1 1
fix 2 1 1
fix 3 1 1

uniaxialMaterial Elastic $matTag1 $E1

element truss 1 1 4  $A1 $matTag1
element truss 2 2 4  $A2 $matTag1
element truss 3 3 4  $A3 $matTag1

 timeSeries Linear $tsTag1

 pattern Plain $patternTag1 $tsTag1  {
    load 4 $nodeLoadX $nodeLoadY
 }
```

DON'T DO THIS or STOP DOING THIS

USE COMMENTS INSTEAD OF VARIABLES

```
model Basic -ndm 2 -ndf 2

#node $tag $xCrd $yCrd
node 1     0.0    0.0
node 2 144.0    0.0
node 3 168.0    0.0
node 4   72.0  96.0

#fix $nodeTag $xFix $yFix
fix 1 1 1
fix 2 1 1
fix 3 1 1

#uniaxialMaterial Elastic $tag $E
uniaxialMaterial Elastic 1 3000.0
```

```
#element truss $tag $iNode $jNode $A $matTag
element truss 1 1 4 10.0 1
element truss 2 2 4   5.0 1
element truss 3 3 4   5.0 1

timeSeries Linear 1

#pattern Plain $tag $tsTag
pattern Plain 1 1 {
   #load $nodeTag $xForce $yForce
   load 4 100.0 -50.0
}
```

You Want This

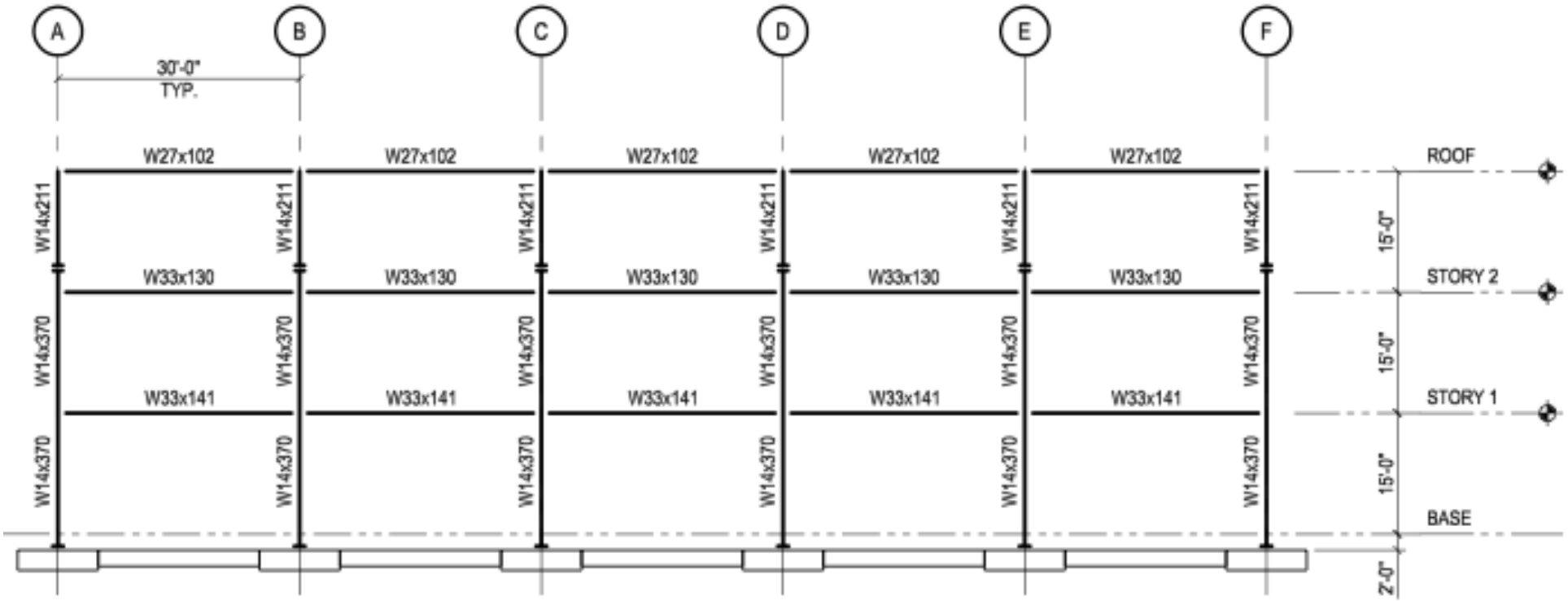
Because if you Don't
Your Moment Frame Script
Will look like ->

3 Story 5 Bay Moment Frame Original Code >350 lines

If I had to do this I would want a GUI too!

Outline of Seminar

- Introduction
- Moment Frame Example 1
- Moment Frame Example 2
- Braced Frame Examples
- Conclusions

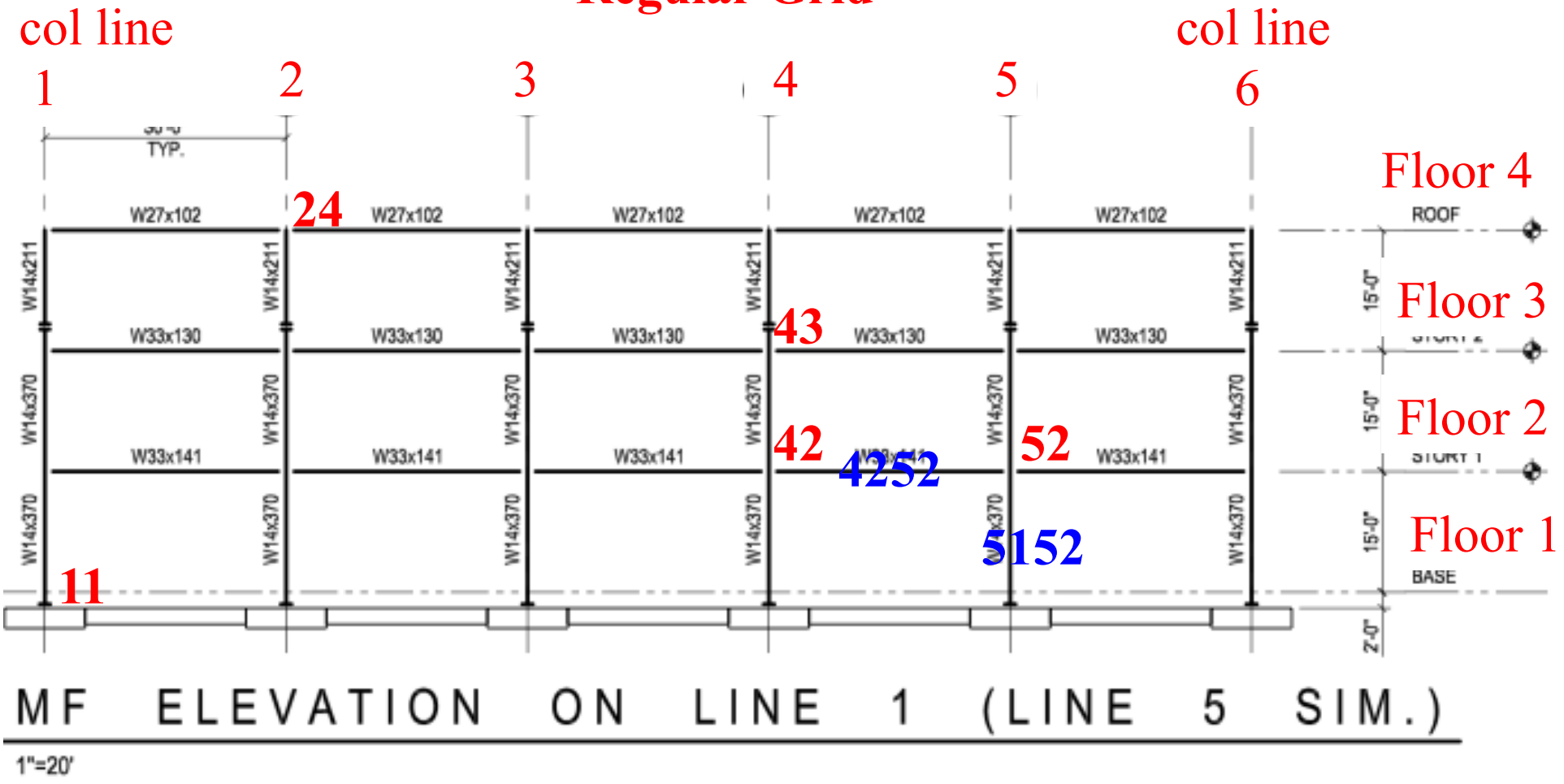


MF ELEVATION ON LINE 1 (LINE 5 SIM.)

1"=20'

THINK Before You Type

Steel W Sections & Regular Grid



Nodes # colfloor

Elements # $iNode$jNode

(if more than 10 col lines or floors, start numbering at 10, if > 100, at 100, ….)

DEMO

MRF1.tcl

```
model Basic –ndm 2 –ndf 3
source Steel2d.tcl

# set some lists containing floor and column line locations and nodal masses
set floorLocs {0. 204. 384. 564.};                  # floor locations in inches
set colLocs {0. 360. 720. 1080. 1440. 1800.};   #column line locations in inches
set massesX {0. 0.419 0.419 0.430};                  # mass at nodes on each floor in x dirn
set massesY{0. 0.105 0.105 0.096} ;                  #    “     “     “     “      “      “    in y dirn

# add nodes at each floor at each column line location & fix nodes if at floor 1
foreach floor {1 2 3 4} floorLoc $floorLocs massX $massesX massY $massesY {
    foreach colLine {1 2 3 4 5 6} colLoc $colLocs {
       node $colLine$floor $colLoc $floorLoc -mass $massX $massY 0.
        if {$floor == 1} {fix $colLine$floor 1 1 1}
    }
}

#uniaxialMaterial Steel02 $tag $Fy $E $b $R0 $cr1 $cr2
uniaxialMaterial Steel02 1 50.0 29000. 0.003  20 0.925 0.15; ; # material to be used for steel elements

# set some list for col and beam sizes
set colSizes  {W14X370 W14X370 W14X211};    #col sizes stories 1, 2 and 3
set beamSizes {W33X141 W33X130 W27X102};  #beams sizes floor 1, 2, and 3

# add columns  at each column line between floors
geomTransf PDelta 1
foreach colLine {1 2 3 4 5 6} {
    foreach floor1 {1 2 3} floor2 { 2 3 4} {
       set theSection [lindex $colSizes [expr $floor1 -1]]; # obtain section size for column
       ForceBeamWSection2d $colLine$floor1$colLine$floor2 $colLine$floor1 $colLine$floor2 $theSection 1 1 –nip 5
    }
 }
#add beams between column lines at each floor
geomTransf Linear 2
foreach colLine1 {1 2 3 4 5} colLine2 {2 3 4 5 6} {
    foreach floor {2 3 4} {
       set theSection [lindex $beamSizes [expr $floor -2]]; # obtain section size for floor
       ForceBeamWSection2d $colLine1$floor$colLine2$floor $colLine1$floor $colLine2$floor $theSection 1 2
    }
}
```

Same model in 35 lines!

Steel2.tcl – contains a library of procedures

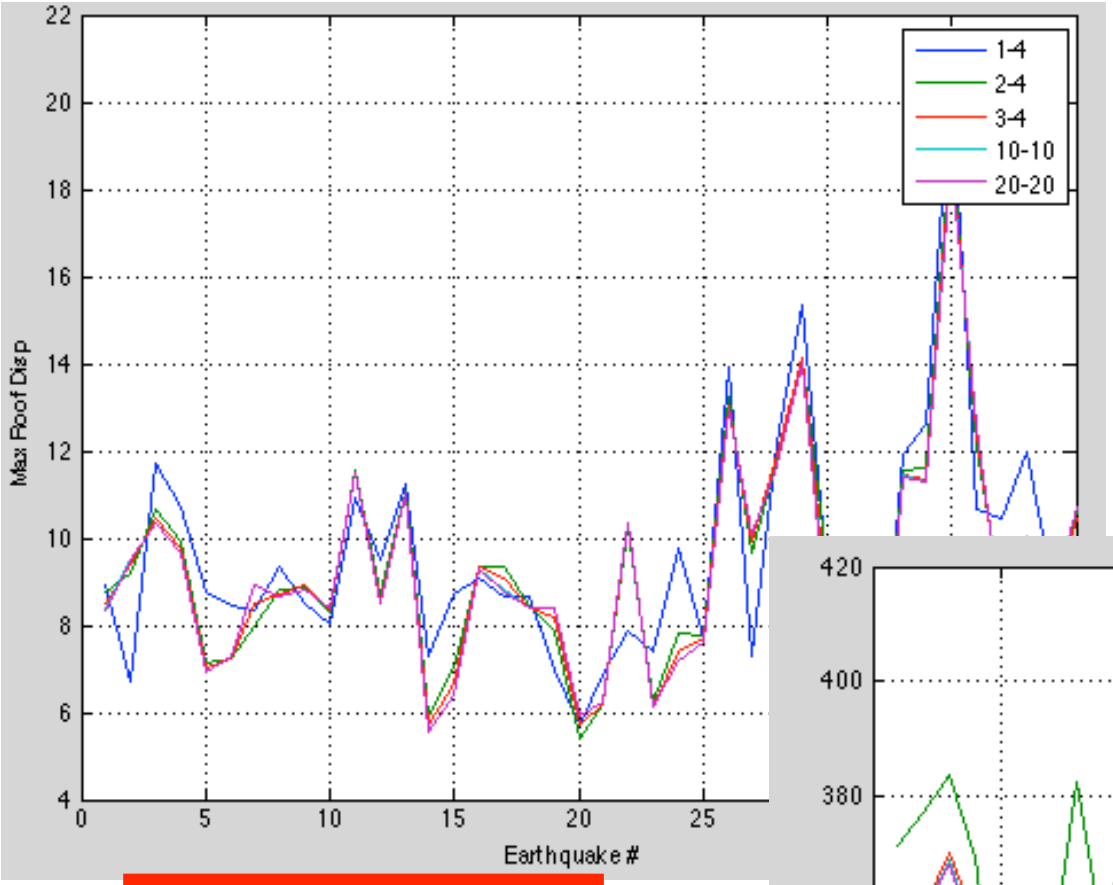
```
proc ElasticBeamWSection2d {eleTag iNode jNode sectType E transfTag {Orient XX}} {
    global WSection
    global in
    set found 0
    foreach {section prop} [array get WSection $sectType] {
        set propList [split $prop]
        set A [expr [lindex $propList 0]*$in*$in]
        set Ixx [expr [lindex $propList 5]*$in*$in*$in*$in]
        set Iyy [expr [lindex $propList 6]*$in*$in*$in*$in]
        if {$Orient == "YY" } {
          element elasticBeamColumn $eleTag $iNode $jNode $A $E $Iyy $transfTag
        } else {
          element elasticBeamColumn $eleTag $iNode $jNode $A $E $Ixx $transfTag
      }
    }
}

#Winxlb/f "Area(in2) d(in) bf(in) tw(in) tf(in) Ixx(in4) Iyy(in4)"
array set WSection {
    W44X335     "98.5 44.0 15.9 1.03 1.77 31100 1200 74.7"
    W44X290     "85.4 43.6 15.8 0.865 1.58 27000 1040 50.9"
    W44X262     "76.9 43.3 15.8 0.785 1.42 24100 923 37.3"
    W44X230     "67.7 42.9 15.8 0.710 1.22 20800 796 24.9"
    W40X593     "174 43.0 16.7 1.79 3.23 50400 2520 445"
    W40X503     "148 42.1 16.4 1.54 2.76 41600 2040 277"
```

```
proc ForceBeamWSection2d {eleTag iNode jNode sectType matTag transfTag args} {

   global FiberSteelWSection2d
   global ElasticSteelWSection2d

   set Orient "XX"
   if {[lsearch $args "YY"] != -1} {
      set Orient "YY"
   }

   set nFlange 8
   if {[lsearch $args "-nFlange"] != -1} {
      set loc [lsearch $args "-nFlange"]
      set nFlange [lindex $args [expr $loc+1]]
   }

   set nWeb  4
   if {[lsearch $args "-nWeb"] != -1} {
      set loc [lsearch $args "-nWeb"]
      set nWeb [lindex $args [expr $loc+1]]
   }

   set nip 4
   if {[lsearch $args "-nip"] != -1} {
      set loc [lsearch $args "-nip"]
                  set nip [lindex $args [expr $loc+1]]
   }

   if {[lsearch $args "-elasticSection"] != -1} {
      set loc [lsearch $args "-elasticSection"]
                  set E [lindex $args [expr $loc+1]]
                  ElasticSteelWSection2d $eleTag $sectType $E  $Orient
   } else {
      FiberSteelWSection2d $eleTag $sectType $matTag $nFlange $nWeb $Orient
   }

   element forceBeamColumn $eleTag $iNode $jNode $nip $eleTag $transfTag
}
```

Now We Can Do Fun Stuff
How Many Fibers?

MRF2.tcl

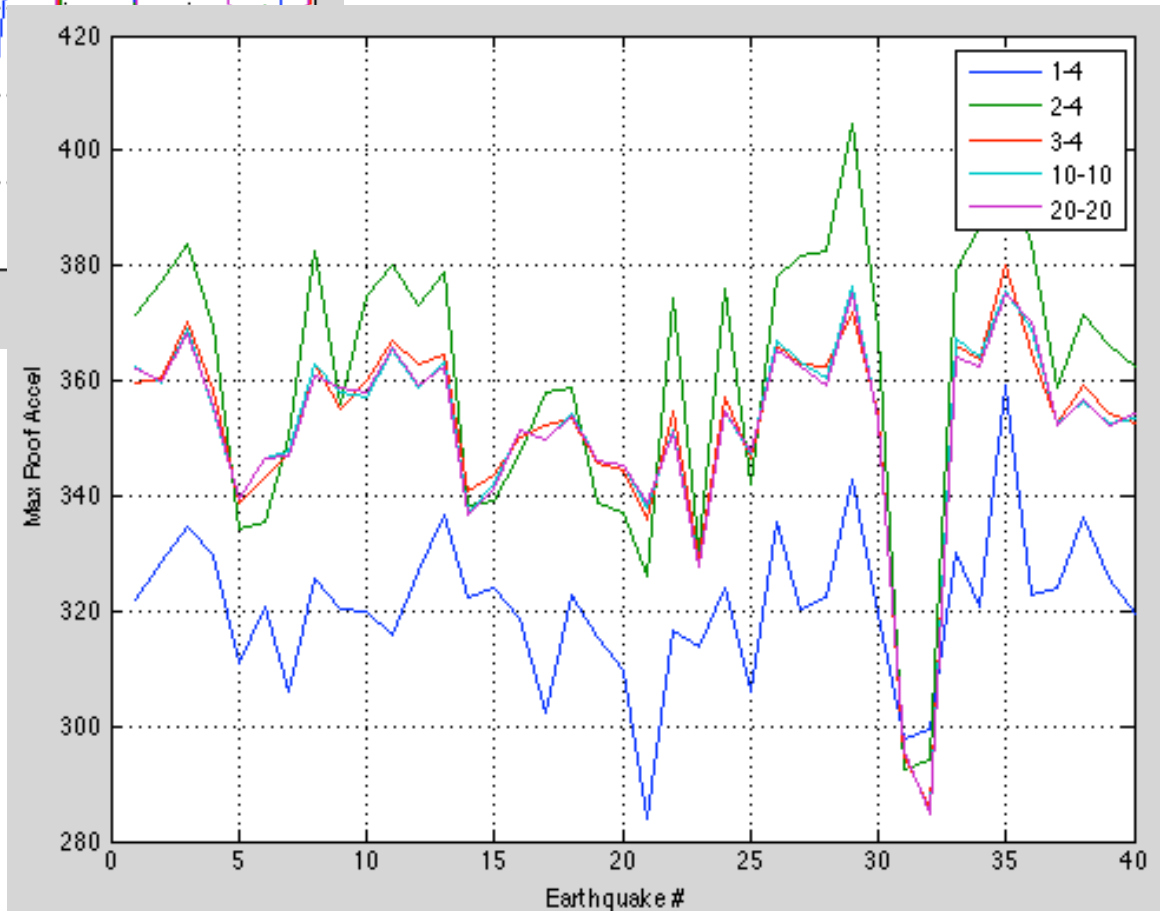
```
foreach nFlange {1 2 3 10 10 20 35} nWeb {4 4 4 5 10 20 30} {
    puts ”nFlange: $nFlange nWeb: $nWeb"

    wipe;
    model BasicBuilder -ndm 2 -ndf 3;

       ….

       ….

 # add columns  at each column line between floors
  geomTransf PDelta 1
  foreach colLine {1 2 3 4 5 6} {
    foreach floor1 {1 2 3} floor2 { 2 3 4} {
      set theSection [lindex $colSizes [expr $floor1 -1]]; # obtain section size for column
      ForceBeamWSection2d $colLine$floor1$colLine$floor2 $colLine$floor1 $colLine$floor2 $theSection 1 1 –nFlange $nFlange –nWeb $nW
    }
  }

   #add beams between column lines at each floor
   geomTransf Linear 2
   foreach colLine1 {1 2 3 4 5} colLine2 {2 3 4 5 6} {
    foreach floor {2 3 4} {
      set theSection [lindex $beamSizes [expr $floor -2]]; # obtain section size for floor
      ForceBeamWSection2d $colLine1$floor$colLine2$floor $colLine1$floor $colLine2$floor $theSection 1 2 –nFlange $nFlange –nWeb $nW
    }
   }
}
```

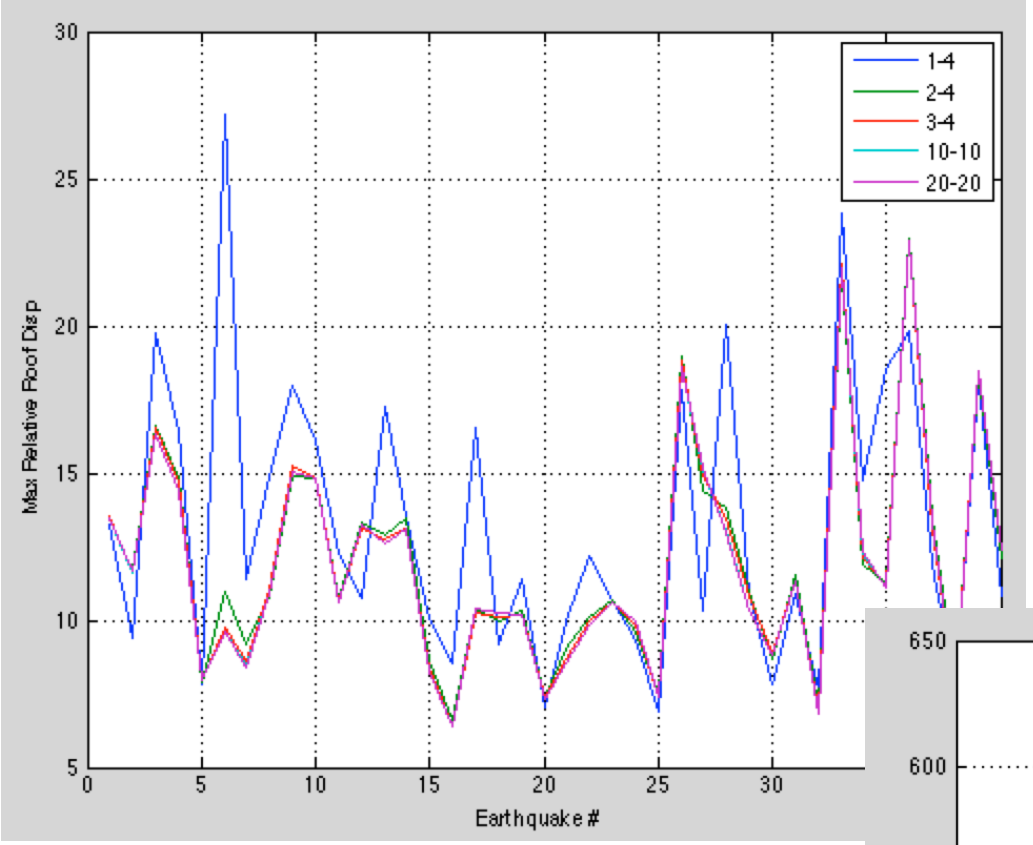


fibers flange- # fibers web

Results 10 % in 50year

	Time
1-4	292sec
2-4	302sec
3-4	309sec
10-10	578sec
20-20	1001sec
35-30	1305sec

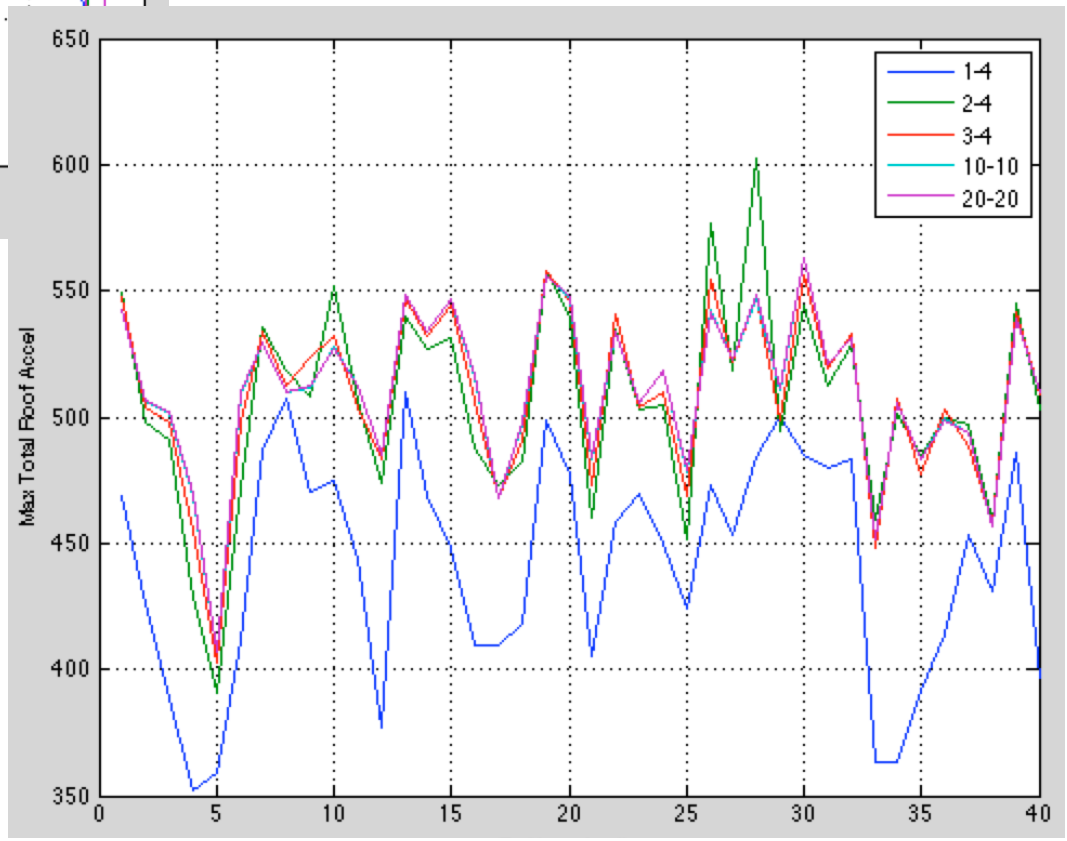




fibers flange- # fibers web

Results 2% in 50year

	Time
1-4	254sec
2-4	265sec
3-4	272sec
10-10	506sec
20-20	879sec
35-30	1539sec



Little More Useful
(BUT COMPILCATED APPROACH)

Replace foreach with for construct

WHY?

So I Can generate a new model in SECONDS.

MRF3.tcl

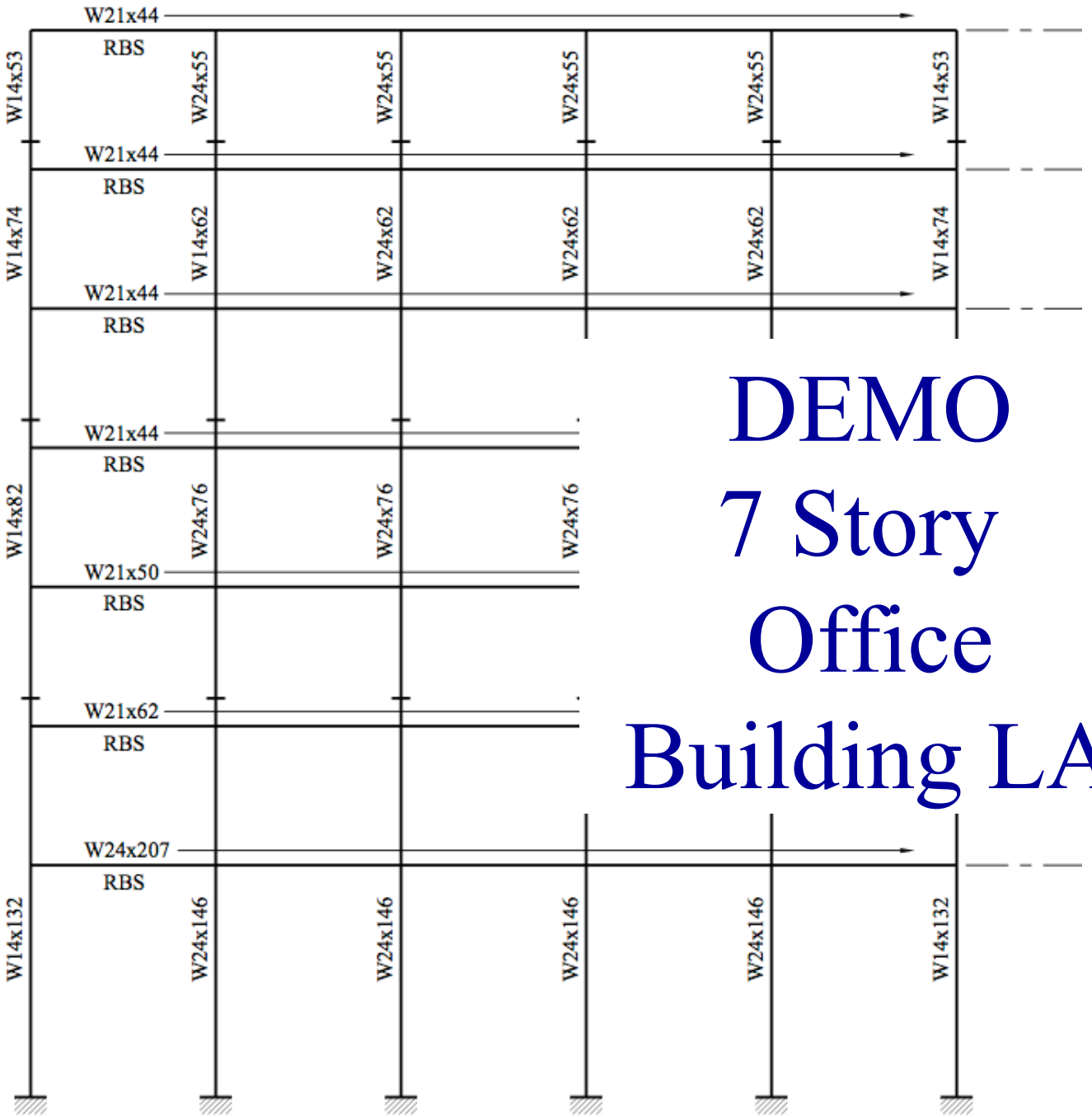
```
model Basic –ndm 2 –ndf 3
source Steel2d.tcl

# set up my structure
set floorOffsets {204. 180. 180.}
set colOffsets   {360. 360. 360. 360. 360.}
set massesX      {0. 0.419 0.419 0.400}
set massesY      {0. 0.105 0.105 0.096}
set colSizes     {W14X370 W14X370 W14X211};
set beamSizes    {W33X141 W33X130 W27X102};

set floorLoad -0.11238
set roofLoad -0.1026

# build colLocations and floorLocations & set some variables
set numFloor [expr [llength $floorOffsets]+1]
set numCline [expr [llength $colOffsets]+1]

set floorLocations 0; set floorLoc 0;
set colLocations 0; set colLoc 0;

for {set i 1} {$i < $numFloor} {incr i 1} {
   set floorLoc [expr $floorLoc + [lindex $floorOffsets [expr $i-1]]]
  lappend floorLocations $floorLoc;
}
for {set i 1} {$i < $numCline} {incr i 1} {
   set colLoc [expr $colLoc + [lindex $colOffsets [expr $i-1]]]
   lappend colLocations $colLoc;
}

# following in case num floors or cols exceed 10
set floorStart 1;
if {$numFloor > 10}  set floorStart 10;
set clineStart 1;
if {$numCline > 10} set clineStart 10;

# check of list dimensions for errors
if {[llength $massesX] != $numFloor} {puts "ERROR: massX"; quit}
if {[llength $massesY] != $numFloor} {puts "ERROR: massY"; quit}
if {[llength $colSizes] != [expr $numFloor-1]} {puts "ERROR: colSizes"; quit}
```

MRF3.tcl

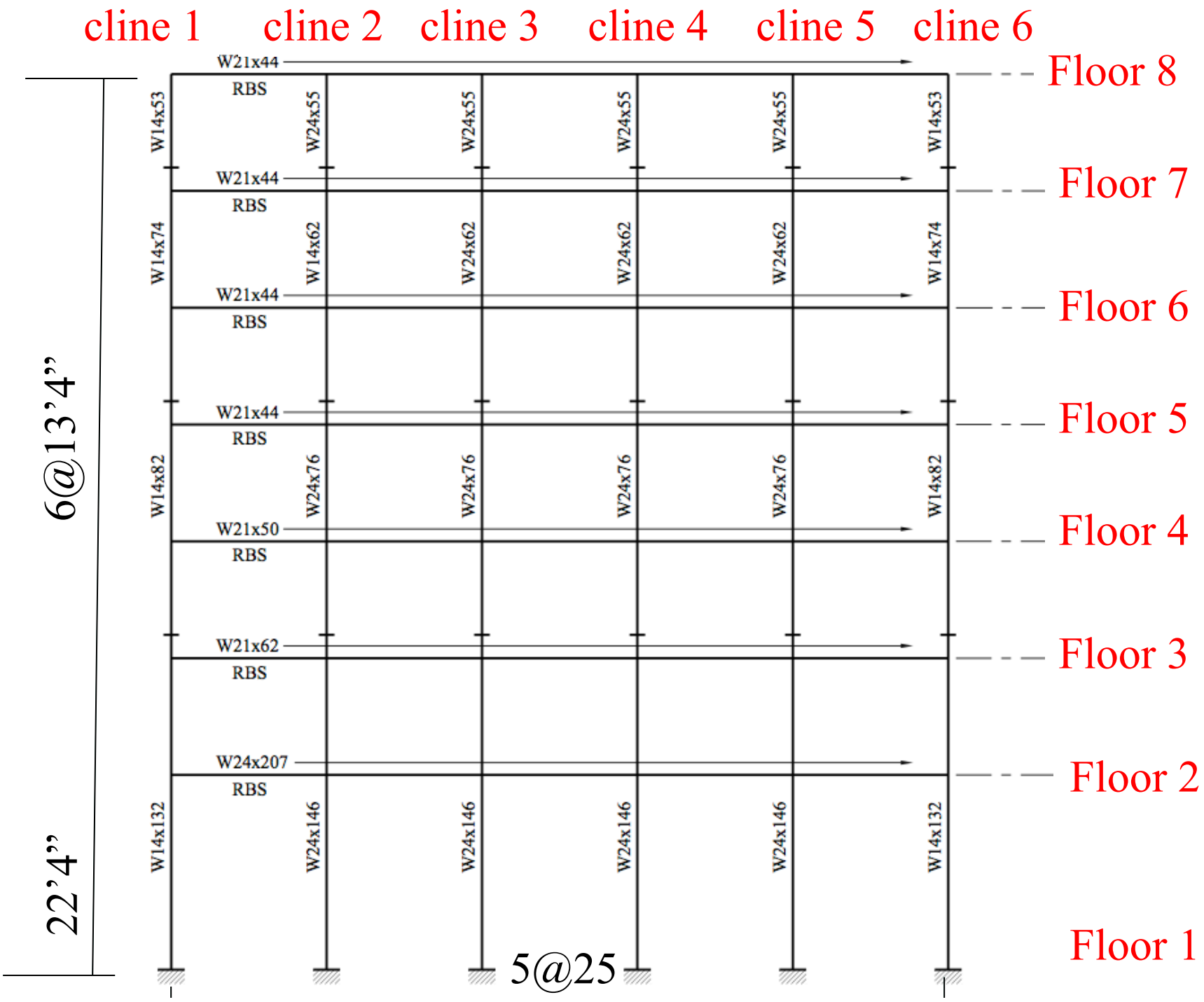
```
# Build the Nodes
for {set floor 1} {$floor <= $numFloor} {incr floor 1} {
   set floorN [expr $floorStart + $floor-1]
    set floorLoc [lindex $floorLocations [expr $floor-1]]
    set massX [lindex $massesX [expr $floor-1]]
    set massY [lindex $massesY [expr $floor-1]]
    for {set colLine 1} {$colLine <= $numCline} {incr colLine 1} {
       set colLineN [expr $clineStart + $colLine -1]
       set colLoc [lindex $colLocations [expr $colLine-1]]
       node $colLineN$floorN $colLoc $floorLoc -mass $massX $massY 0.
       if {$floor == 1} {
          fix $colLineN$floorN 1 1 1
       }
    }
}
# build the columns
geomTransf PDelta 1
for {set colLine 1} {$colLine <= $numCline} {incr colLine 1} {
    set colLineN [expr $clineStart + $colLine -1]
    for {set floor1 1} {$floor1 < $numFloor} {incr floor1 1} {
       set floor2 [expr $floor1+1]
       set floor1N [expr $floorStart + $floor1-1]
       set floor2N [expr $floorStart + $floor1]
       set theSection [lindex $colSizes [expr $floor1 -1]]
       ForceBeamWSection2d $colLineN$floor1N$colLineN$floor2N $colLineN$floor1N $colLineN$floor2N $theSection 1 1 –nip 5
    }
}
## build the beams
geomTransf Linear 2
for {set colLine1  1} {$colLine1 < $numCline} {incr colLine1 1} {
    set colLine2 [expr $colLine1 + 1]
    set colLine1N [expr $clineStart + $colLine1 -1]
    set colLine2N [expr $clineStart + $colLine1]
    for {set floor 2} {$floor <= $numFloor} {incr floor 1} {
        set floorN [expr $floorStart + $floor-1]
       set theSection [lindex $beamSizes [expr $floor -2]]
       ForceBeamWSection2d $colLine1N$floorN$colLine2N$floorN $colLine1N$floorN $colLine2N$floorN $theSection 1 2
    }
`
```

Outline of Seminar

- Introduction
- Moment Frame Example 1
- Moment Frame Example 2
- Braced Frame Examples
- Conclusions



DEMO 7 Story Office Building LA


cline 1
cline 2
cline 3
cline 4
cline 5
cline 6
W21x44
RBS
W14x53
W24x55
W24x55
W24x55
W24x55
W14x53
Floor 8
W21x44
Floor 7
W14x74
W14x62
W24x62
W24x62
W24x62
W14x74
W21x44
Floor 6
RBS
W21x44
Floor 5
W14x82
W24x76
W24x76
W24x76
W24x76
W14x82
W21x50
Floor 4
RBS
W21x62
Floor 3
RBS
W24x207
Floor 2
RBS
W14x132
W24x146
W24x146
W24x146
W24x146
W14x132
Floor 1
6@13'4"
22'4"
5@25

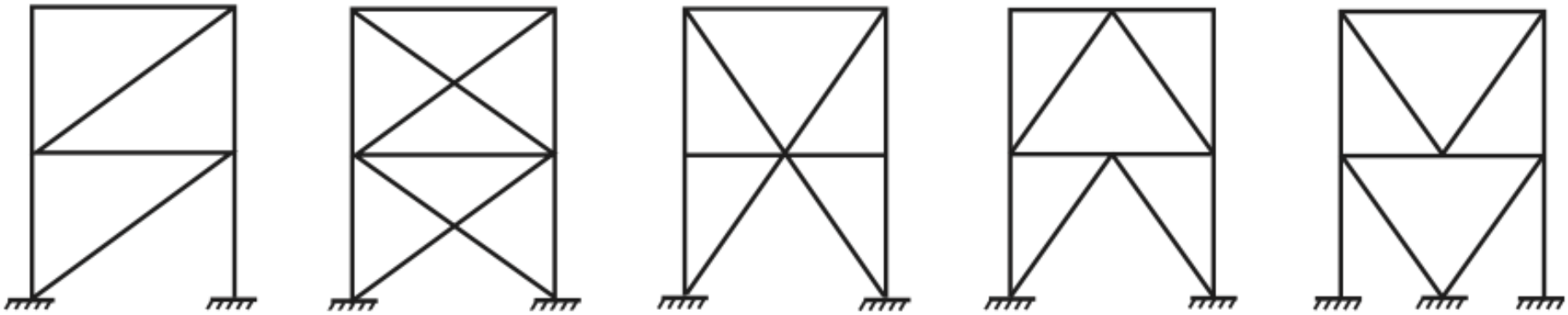
MRF4.tcl

```
model Basic –ndm 2 –ndf 3
source Steel2d.tcl

# set up my structure
set floorOffsets {268. 160. 160. 160. 160. 160. 160.}
set colOffsets    {300. 300. 300. 300. 300.}
set colSizes      {W24X146 W24X146 W24X76 W24X76 W14X61 W14X61 W24X55};
set beamSizes   {W24X207 W21X62 W21X50 W21X44 W21X44 W21X44 W21X44};

set colSizesE    {W14X132 W14X132 W14X82 W14X82 W14X74 W14X74 W14X53};
….
….
set roofWeight 1537.0;
set floorWeight 1920.0;
set floorLoad [expr (-$floorWeight*12.5)/(2.*127.*177.)];
set roofLoad  [expr (-$roofWeight*12.5)/(2.*127.*177.)];
set roofMass [expr $roofWeight/($g*2.*$numCline)]; # kips 2 frames per dirn 6 col line
set floorMass [expr $floorWeight/($g*2.*$numCline)]
set massesCMD  "set masses {0. $floorMass $floorMass $floorMass $floorMass $floorMass $floorMa
eval $massesCMD
….
….
# build the columns
geomTransf PDelta 1
for {set colLine 1} {$colLine <= $numCline} {incr colLine 1} {
   set colLineN [expr $clineStart + $colLine -1]
   for {set floor1 1} {$floor1 < $numFloor} {incr floor1 1} {
      set floor2 [expr $floor1+1]
      set floor1N [expr $floorStart + $floor1-1]
            set floor2N [expr $floorStart + $floor1]
      set theSection [lindex $colSizes [expr $floor1 -1]]
      if {$colLine == 1 || $colLine == $numCline} {
         set theSection [lindex $colSizesE [expr $floor1 -1]]
      } else {
      set theSection [lindex $colSizes [expr $floor1 -1]]
      }
      ForceBeamWSection2d $colLineN$floor1N$colLineN$floor2N $colLineN$floor1N $colLineN$floor
p 5
```

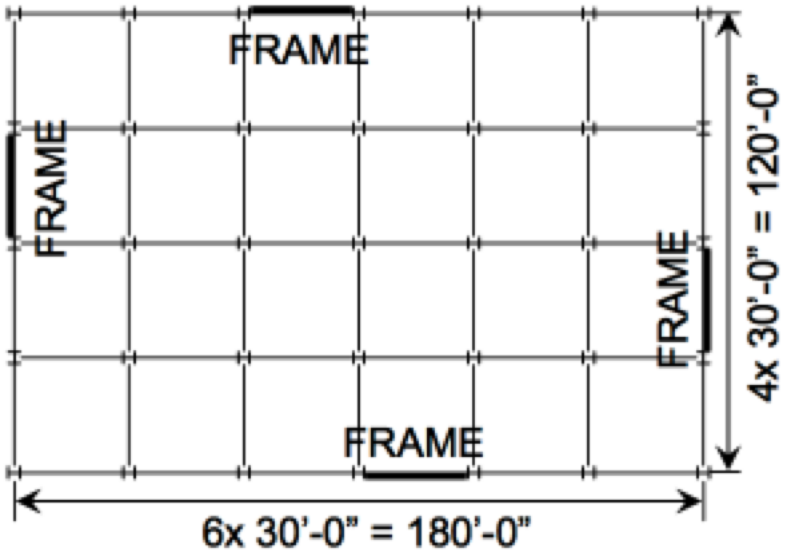
Outline of Seminar

- Introduction
- Moment Frame Example 1
- Moment Frame Example 2
- Braced Frame Examples
- Conclusions

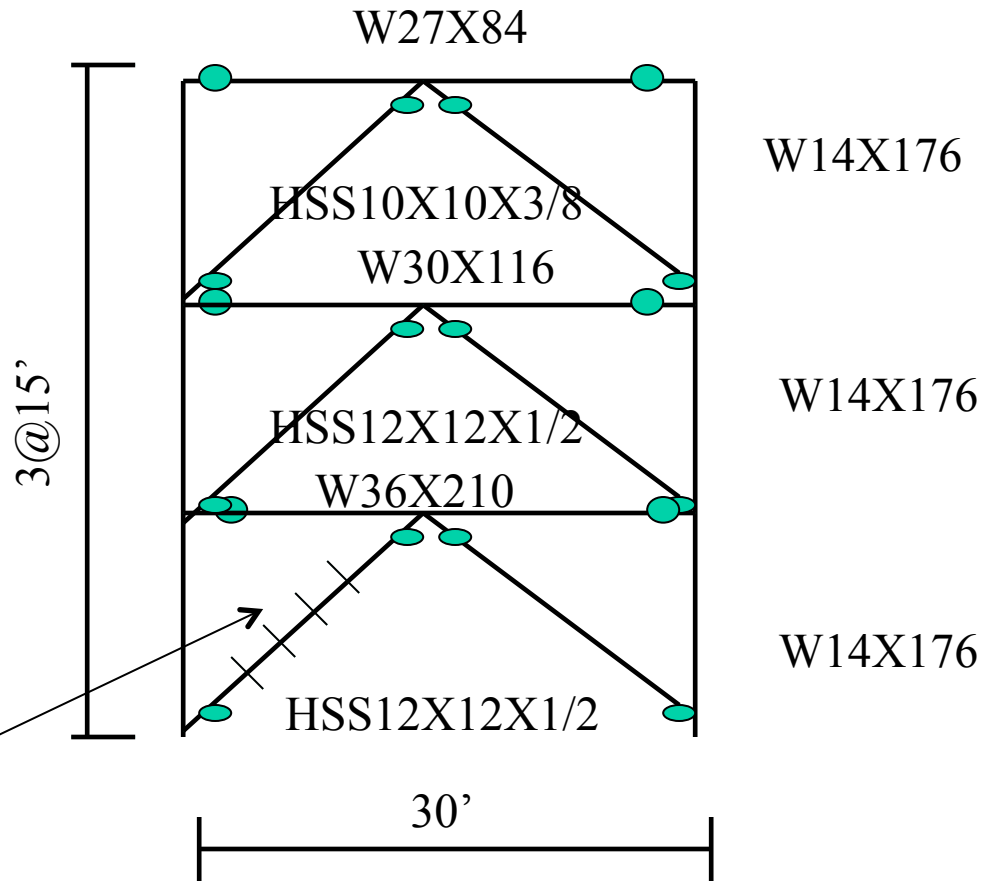
Typical Configurations



Diagonal bracing | X-bracing | Multistory X-bracing | Inverted V-bracing (Chevron) | V-bracing



- Beams pinned at column



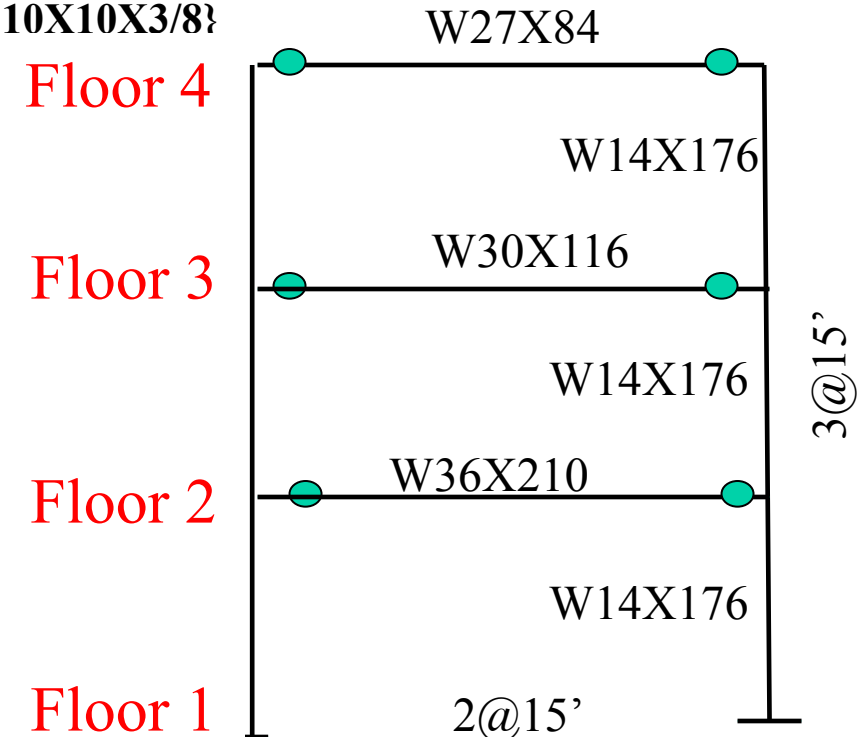
#number of beam elements
pinned at either end
Imperfection so will buckle

proc HSSbrace $eleTag $iNode $jNode $secType $matTag $numSeg $imperfection $transfTag arg

D Lignos (McGill)

CBFbase.tcl

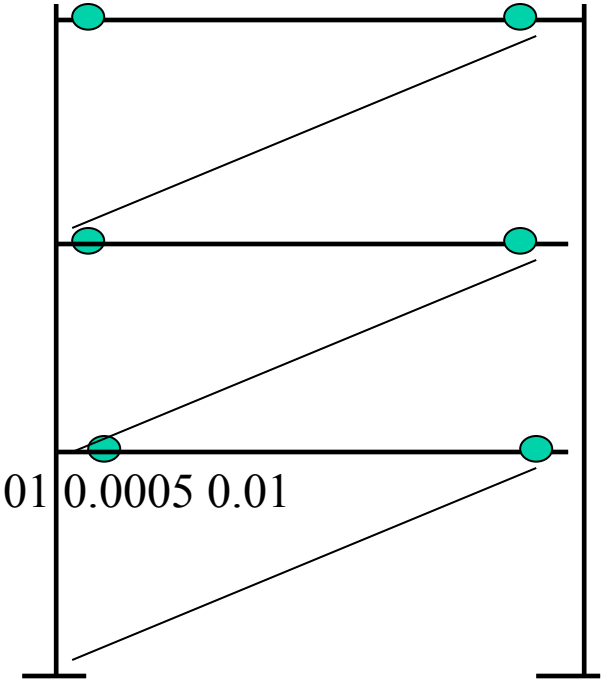
```
source Steel2d.tcl
# set up my structure
set colLocations {0. 180. 360.}
set floorLocations {0. 180. 360. 540.}
set masses      {0. 0.419 0.419 0.400}
set colSizes      {W14X176 W14X176 W14X176};
set beamSizes    {W36X210  W30X116 W27X84};
set braceSizes    {HSS12X12X1/2 HSS12X12X1/2 HSS10X10X3/8}

set numFloor [llength $floorLocations]
set numStory [expr $numFloor-1]
set numCline [llength $colLocations]

model BasicBuilder -ndm 2 -ndf 3;
#add the Nodes
for {set floor 1} {$floor <= $numFloor} {incr floor 1} {
   set floorLoc [lindex $floorLocations [expr $floor-1]]
   set mass [lindex $masses [expr $floor-1]]
   for {set colLine 1} {$colLine <= $numCline} {incr colLine 1}
      set colLoc [lindex $colLocations [expr $colLine-1]]
      node $colLine$floor $colLoc $floorLoc -mass $mass $mass 0.
      if {$floor == 1} {
         fix $colLine$floor 1 1 1
      }
   }
}
# define material for col and beams
uniaxialMaterial Steel02 1 50.0 29000 0.03 20 0.925 0.15 0.0005 0.01 0.0005 0.01

# add the columns
geomTransf PDelta 1
for {set colLine 1} {$colLine <= $numCline} {incr colLine 2} {
   for {set floor1 1} {$floor1 < $numFloor} {incr floor1 1} {
      set floor2 [expr $floor1+1]
      set theSection [lindex $colSizes [expr $floor1 -1]]
      ForceBeamWSection2d $colLine$floor1$colLine$floor2 $colLine$floor1 $colLine$floor2 $theSectic
   }
}

# add the beams, pinned connection at column end
geomTransf Linear 2
for {set floor 2} {$floor <= $numFloor} {incr floor 1} {
   set colLine1 1; set colLine2 2; set colLine3 3;
   set theSection [lindex $beamSizes [expr $floor -2]]
   DispBeamWSection2d $colLine1$floor$colLine2$floor $colLine1$floor $colLine2$floor $theSection 1 2 -release1
   DispBeamWSection2d $colLine2$floor$colLine3$floor $colLine2$floor $colLine3$floor $theSection 1 2 -release2
}
```



* While diagonal and X braced models do not need me to break up beams, I am doing so as to keep all examples the same.

CBF1.tcl
(Diagonal bracing)

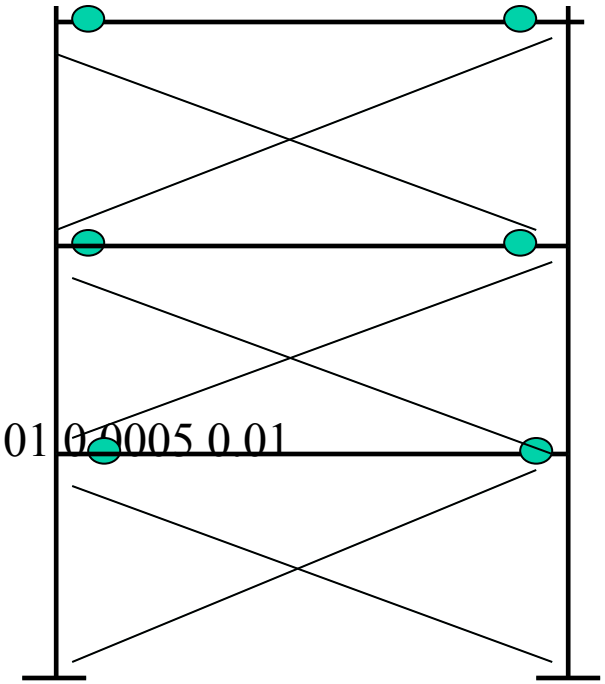
```
source Steel2d.tcl
# set up my structure
set colLocations {0. 180. 360.}
set floorLocations {0. 180. 360. 540.}
set masses      {0. 0.419 0.419 0.400}
set colSizes      {W14X176 W14X176 W14X176};
set beamSizes    {W36X210  W30X116 W27X84};
set braceSizes    {HSS12X12X1/2 HSS12X12X1/2 HSS10X10X3/8}

set numFloor [llength $floorLocations]
set numStory [expr $numFloor-1]
set numCline [llength $colLocations]

model BasicBuilder -ndm 2 -ndf 3;
#add the Nodes
for {set floor 1} {$floor <= $numFloor} {incr floor 1} {
   set floorLoc [lindex $floorLocations [expr $floor-1]]
   set mass [lindex $masses [expr $floor-1]]
   for {set colLine 1} {$colLine <= $numCline} {incr colLine 1}
```



```
# define material for braces
set Fy_b 46.0;

set E0 0.095
set m -0.3
uniaxialMaterial Steel02 2 $Fy_b $Es $b 20 0.925 0.15 0.0005 0.01 0.0005 0.01
uniaxialMaterial Fatigue 3 2 -E0 $E0 -m $m -min -1.0 -max 0.04

set imperfection 0.001

# add the X braces
geomTransf Corotational 3
for {set story 1} {$story <= $numStory} {incr story 1} {
    set colLine1 1; set colLine2 3;
    set floor1 $story; set floor2 [expr $story +1];

    set theSection [lindex $braceSizes [expr $story -1]
    HSSbrace $colLine1$floor1$colLine2$floor2 $colLine1$floor1 $colLine2$floor2 $theSection$
}
```

CBF2.tcl
(X bracing)

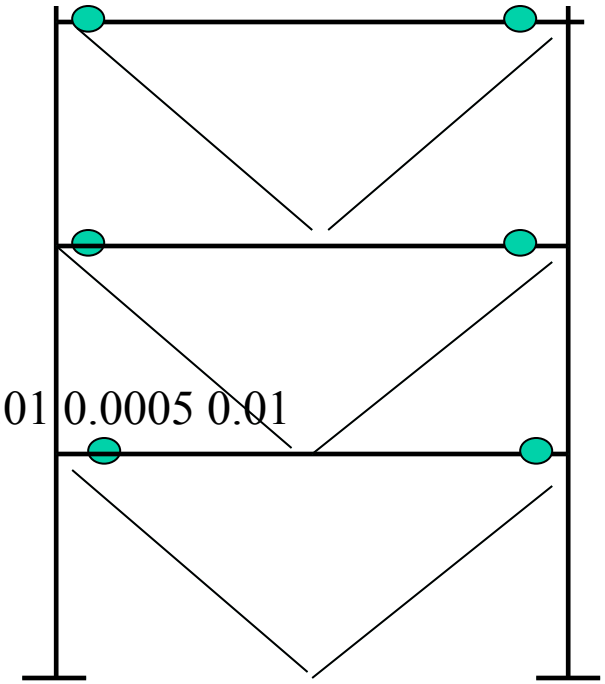
```
source Steel2d.tcl
# set up my structure
set colLocations {0. 180. 360.}
set floorLocations {0. 180. 360. 540.}
set masses      {0. 0.419 0.419 0.400}
set colSizes      {W14X176 W14X176 W14X176};
set beamSizes    {W36X210  W30X116 W27X84};
set braceSizes    {HSS12X12X1/2 HSS12X12X1/2 HSS10X10X3/8}

set numFloor [llength $floorLocations]
set numStory [expr $numFloor-1]
set numCline [llength $colLocations]

model BasicBuilder -ndm 2 -ndf 3;
#add the Nodes
for {set floor 1} {$floor <= $numFloor} {incr floor 1} {
   set floorLoc [lindex $floorLocations [expr $floor-1]]
   set mass [lindex $masses [expr $floor-1]]
   for {set colLine 1} {$colLine <= $numCline} {incr colLine 1}
```



```
# define material for braces
set Fy_b 46.0;
set E0 0.095
set m -0.3
uniaxialMaterial Steel02 2 $Fy_b $Es $b 20 0.925 0.15 0.0005 0.01 0.0005 0.01
uniaxialMaterial Fatigue 3 2 -E0 $E0 -m $m -min -1.0 -max 0.04

set imperfection 0.001

# add the X braces
geomTransf Corotational 3
for {set story 1} {$story <= $numStory} {incr story 1} {
    set colLine1 1; set colLine2 2; set colLine3 3;
    set floor1 $story; set floor2 [expr $story +1];
    set theSection [lindex $braceSizes [expr $story -1]]
    # proc HSSbrace {eleTag iNode jNode secType matTag numSeg Im transfTag args}
    HSSbrace $colLine1$floor1$colLine3$floor2 $colLine1$floor1 $colLine3$floor2 $theSection
    HSSbrace $colLine3$floor1$colLine1$floor2 $colLine3$floor1 $colLine1$floor2 $theSection
}
```

CBF3.tcl (V bracing)

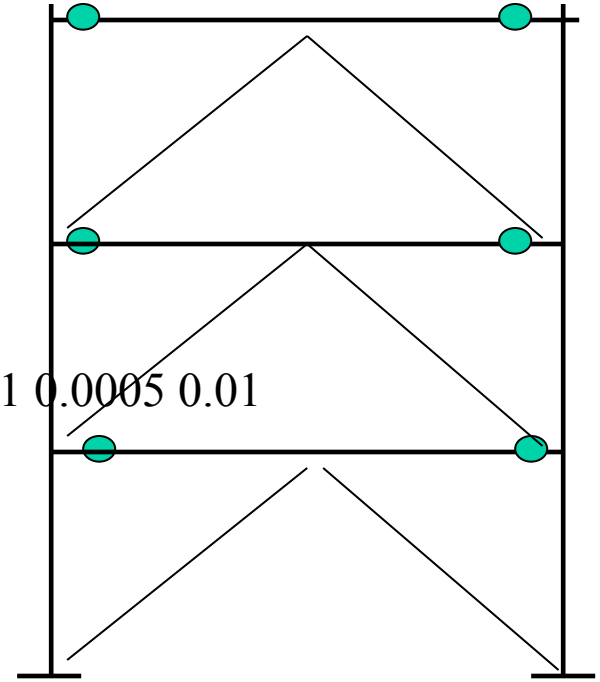
```
source Steel2d.tcl
# set up my structure
set colLocations {0. 180. 360.}
set floorLocations {0. 180. 360. 540.}
set masses      {0. 0.419 0.419 0.400}
set colSizes      {W14X176 W14X176 W14X176};
set beamSizes    {W36X210  W30X116 W27X84};
set braceSizes    {HSS12X12X1/2 HSS12X12X1/2 HSS10X10X3/8}

set numFloor [llength $floorLocations]
set numStory [expr $numFloor-1]
set numCline [llength $colLocations]

model BasicBuilder -ndm 2 -ndf 3;
#add the Nodes
for {set floor 1} {$floor <= $numFloor} {incr floor 1} {
   set floorLoc [lindex $floorLocations [expr $floor-1]]
```



```
# define material for braces
set Fy_b 46.0;
set E0 0.095
set m -0.3
uniaxialMaterial Steel02 2 $Fy_b $Es $b 20 0.925 0.15 0.0005 0.01 0.0005 0.01
uniaxialMaterial Fatigue 3 2 -E0 $E0 -m $m -min -1.0 -max 0.04

set imperfection 0.001

# add the V braces
geomTransf Corotational 3
for {set story 1} {$story <= $numStory} {incr story 1} {
    set colLine1 1; set colLine2 2; set colLine3 3;
    set floor1 $story; set floor2 [expr $story +1];
    set theSection [lindex $braceSizes [expr $story -1]]
    HSSbrace $colLine2$floor1$colLine1$floor2 $colLine2$floor1 $colLine1$floor2 $theSection
    HSSbrace $colLine2$floor1$colLine3$floor2 $colLine2$floor1 $colLine3$floor2 $theSection
}
```

CBF4.tcl
(inverted V bracing)

```
source Steel2d.tcl
# set up my structure
set colLocations {0. 180. 360.}
set floorLocations {0. 180. 360. 540.}
set masses      {0. 0.419 0.419 0.400}
set colSizes      {W14X176 W14X176 W14X176};
set beamSizes   {W36X210  W30X116 W27X84};
set braceSizes   {HSS12X12X1/2 HSS12X12X1/2 HSS10X10X3/8}

set numFloor [llength $floorLocations]
set numStory [expr $numFloor-1]
set numCline [llength $colLocations]

model BasicBuilder -ndm 2 -ndf 3;
#add the Nodes
for {set floor 1} {$floor <= $numFloor} {incr floor 1} {
```



```
# define material for braces
set Fy_b 46.0;
set E0 0.095
set m -0.3
uniaxialMaterial Steel02 2 $Fy_b $Es $b 20 0.925 0.15 0.0005 0.01 0.0005 0.01
uniaxialMaterial Fatigue 3 2 -E0 $E0 -m $m -min -1.0 -max 0.04

set imperfection 0.001

# add the inverted V braces
geomTransf Corotational 3
for {set story 1} {$story <= $numStory} {incr story 1} {
    set colLine1 1; set colLine2 2; set colLine3 3;
    set floor1 $story; set floor2 [expr $story +1];
    set theSection [lindex $braceSizes [expr $story -1]]
    HSSbrace $colLine1$floor1$colLine2$floor2 $colLine1$floor1 $colLine2$floor2 $theSection 3
    HSSbrace $colLine3$floor1$colLine2$floor2 $colLine3$floor1 $colLine2$floor2 $theSection 3
}
```

Outline of Seminar

- Introduction
- Moment Frame Example 1
- Moment Frame Example 2
- Braced Frame Examples
- Conclusions

- THINK Before You Type
- Develop a Library of Procedures
- If Done Correctly, someone who develops their models using a scripting language will crush someone who has to use a GUI.

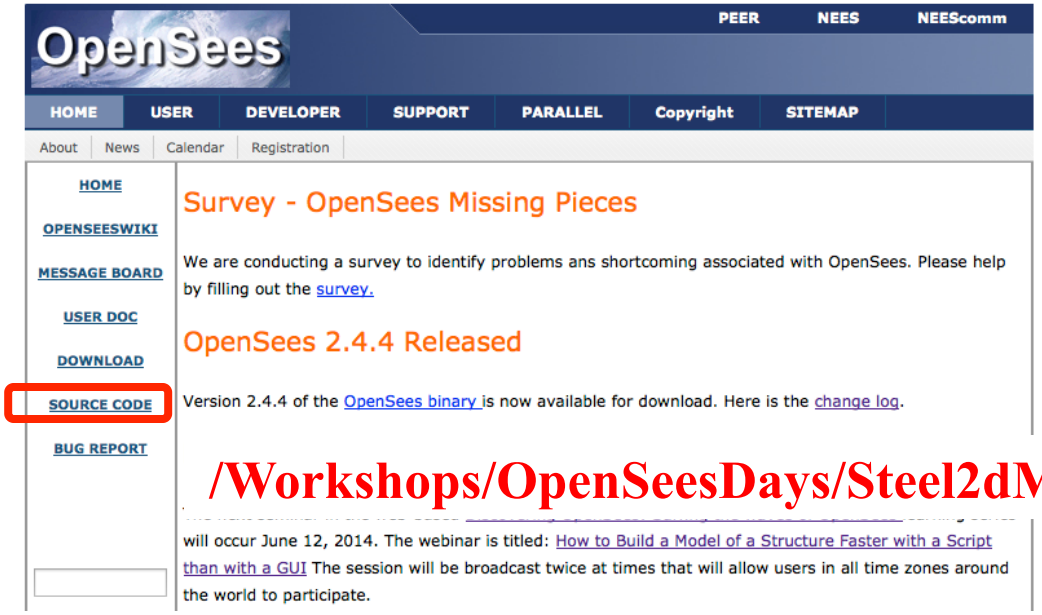
* AND Spend the time you save with scripting to explore the effects of modeling choices (i.e. element types, discretizations, ..) on the response.

Obtaining This Source Code

svn option:

svn co svn://opensees.berkeley.edu:/usr/local/svn/OpenSees/trunk/Workshops/OpenSeesDays/Steel2dModels Steel2dModels

browser option (source on main page):



Any Questions?