Geotechnical Examples using **OpenSees**

Pedro Arduino

University of Washington, Seattle

OpenSees Days 2010,
OpenSees User Workshop, Thursday Sept 2, 2010

Outline

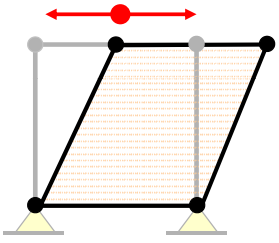
- **Take Advantage of tcl scripting!!**
  - Simple example
  - Push over pile analysis
  - Layout of more advance scripts for:
    - Push-over pile analysis
    - free-field analysis in liquefiable soils
  - Useful links
- **Take advantage of pre- and post- processors**
  - GID, just one possible alternative ☺
  - Dynamic analysis of piles
  - Analysis of complete bridge system
  - 3D analysis of piles

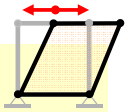
Take advantage of **tcl** scripting!!

- **tcl** can be used to develop scripts for geotechnical applications.
- Possible applications:
  - Pushover analysis (p-y) including nonlinear pile response – (similar to **lpile)**
  - Free-field analysis (including liquefiable soil conditions)- (similar to **shake**)
  - Pile load tests
  - Consolidation
  - Dynamic analysis of piles
  - Other ….

Basic Example

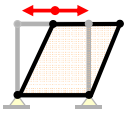
- Response of saturated soil element to harmonic excitation





Define basic variables using good names!!!

```
#Created by Zhaohui Yang (zhyang@ucsd.edu)
#plastic pressure dependent material
#plane strain,  single element,  dynamic analysis (input motion: sinusoidal acceleration at base)
#SI units (m, s, KN, ton)
#
#       4    3
#       -------
#       |    |
#       |    |
#       |    |
#      1-------2   (nodes 1 and 2 fixed)
#       ^    ^
#        <--> input motion: sinusoidal acceleration at base
wipe
#

#some user defined variables
#
set accMul   2       ;# acceleration multiplier
set massDen  2.0     ;# solid mass density
set fluidDen 1.0     ;# fluid mass density
set massProportionalDamping   0.0 ;
set stiffnessProportionalDamping 0.001 ;
set fangle 31.40       ;#friction angle
set ptangle 26.50      ;#phase transformation angle
set E     90000.0     ;#shear modulus
set poisson 0.40 ;
set G [expr $E/(2*(1+$poisson))] ;
set B [expr $E/(3*(1-2*$poisson))] ;
set press   0.0 ;# isotropic consolidation pressure on quad element(s)
set deltaT   0.010  ;# time step for analysis
set numSteps 2000   ;# Number of analysis steps
set gamma    0.600   ;# Newmark integration parameter
set period   1     ;# Period of applied sinusoidal load
set pi 3.1415926535    ;
set inclination 0      ;
set unitWeightX [expr  ($massDen-$fluidDen)*9.81*sin($inclination/180.0*$pi)] ;# unit weight in X direction
set unitWeightY [expr -($massDen-$fluidDen)*9.81*cos($inclination/180.0*$pi)] ;# unit weight in Y direction
```

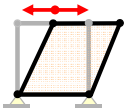


Define model geometry, materials & fixities

```
##################################################################

##################################################################

#create the ModelBuilder
model basic -ndm 2 -ndf 2

# define material and properties
nDMaterial PressureDependMultiYield 2 2 $massDen $G $B  $fangle .1 80 0.5 \
                        $ptangle 0.17 0.4 10  10 0.015 1.0
nDMaterial FluidSolidPorous 1 2 2 2.2D+6

updateMaterialStage -material 1 -stage 0
updateMaterialStage -material 2 -stage 0

# define the nodes
node 1   0.0D0 0.0D0
node 2   1.0D0 0.0D0
node 3   1.0D0 1.0D0
node 4   0.0D0 1.0D0

# define the element     thick  material     maTag   press    mDensity   gravity
element quad  1  1 2 3 4  1.0   "PlaneStrain"    1   $press     0.      $unitWeightX  $unitWeightY

# fix the base
fix 1 1 1
fix 2 1 1

#tie nodes 3 and 4
equalDOF 3 4   1 2
```

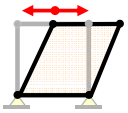
Define gravity step

VERY IMPORTANT!!!



```
##################################################################
# GRAVITY APPLICATION (elastic behavior)
# create the SOE, ConstraintHandler, Integrator, Algorithm and Numberer
system ProfileSPD
test NormDispIncr 1.D-12 25 0
constraints Transformation
integrator LoadControl 1 1 1 1
algorithm Newton
numberer RCM

# create the Analysis
analysis Static

#analyze
analyze 2

# switch the material to plastic
updateMaterialStage -material 1 -stage 1
updateMaterialStage -material 2 -stage 1
updateParameter -material 2 -refB [expr $G*2/3.];

#analyze
analyze 1
```

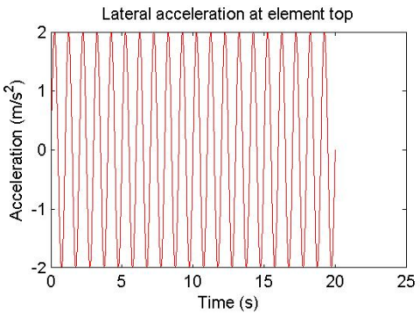
Define dynamic step



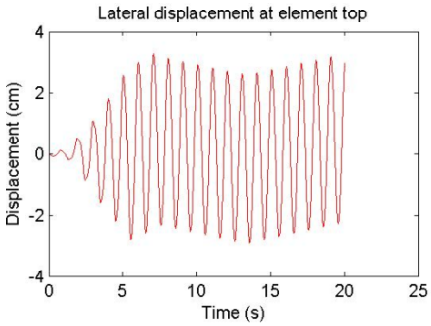
```
##################################################################
# NOW APPLY LOADING SEQUENCE AND ANALYZE (plastic)

# rezero time
setTime 0.0
wipeAnalysis

#create loading pattern
pattern UniformExcitation 1 1 -accel "Sine 0 1000 $period -factor $accMul"

# create the Analysis
constraints Transformation;
test NormDispIncr 1.D-12 25 0
algorithm Newton
numberer RCM
system ProfileSPD
integrator Newmark $gamma  [expr pow($gamma+0.5, 2)/4]  \
            $massProportionalDamping 0.0 $stiffnessProportionalDamping 0.0
analysis VariableTransient

#create Recorders
recorder Node -file disp.out   -time  -node 1 2 3 4 -dof 1 2 -dT  0.01 disp
recorder Node -file acce.out  -time  -node 1 2 3 4 -dof 1 2 -dT 0.01 accel
recorder Element -ele 1 -time -file stress1.out material 1 stress -dT 0.01
recorder Element -ele 1 -time -file strain1.out material 1 strain -dT 0.01
recorder Element -ele 1 -time -file stress3.out material 3 stress -dT 0.01
recorder Element -ele 1 -time -file strain3.out material 3 strain -dT 0.01
recorder Element -ele 1 -time -file press1.out material 1 pressure -dT 0.01
recorder Element -ele 1 -time -file press3.out material 3 pressure -dT 0.01

#analyze
set startT [clock seconds]
analyze $numSteps $deltaT [expr $deltaT/100] $deltaT 10
set endT [clock seconds]
puts "Execution time: [expr $endT-$startT] seconds."

wipe  #flush ouput stream
```

Plot Results

Input Accel time history

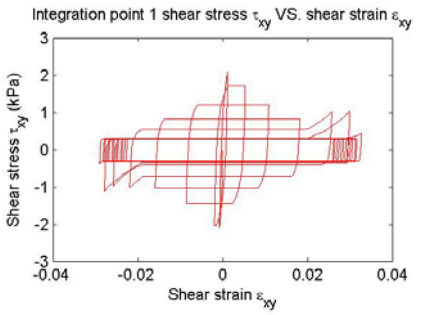


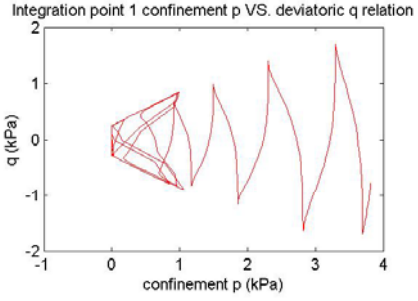
Output displ. time history



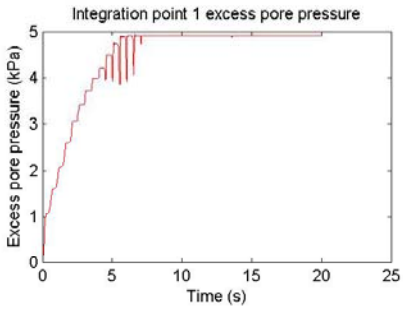
Plot results

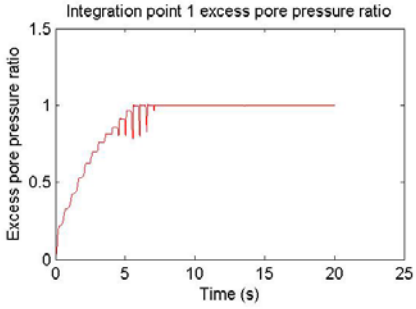
Stress-strain & stress path



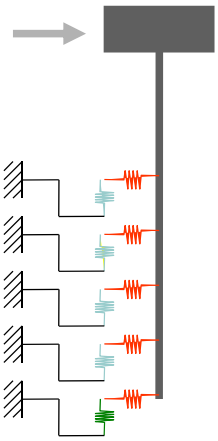


Pore pressures





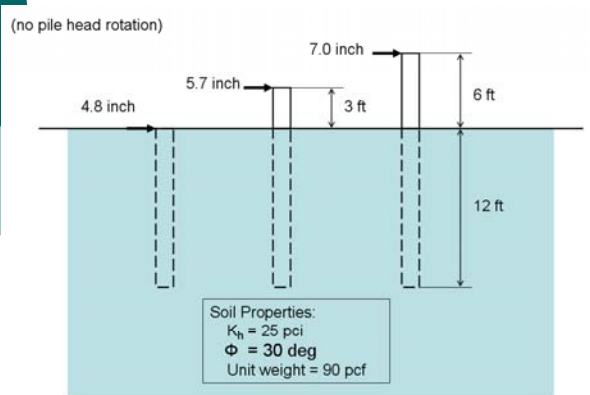
Pushover Pile Analysis using **OpenSees**

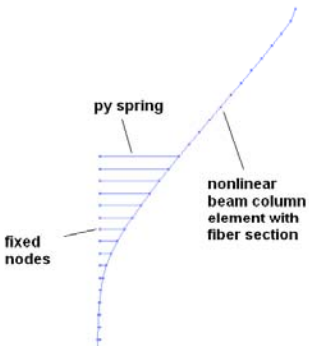


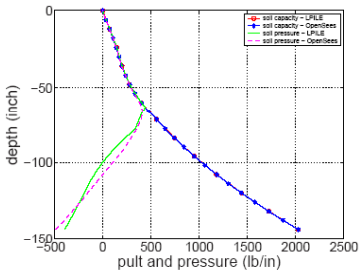
Pile pushover analysis

- Calculate σ_v at p-y location
- Calculate pult, y50 & construct p-y table
- Gravity analysis
- Define structure & connect interface springs
- Apply weight on structure
- Analysis

Pushover Pile Analysis using **OpenSees**

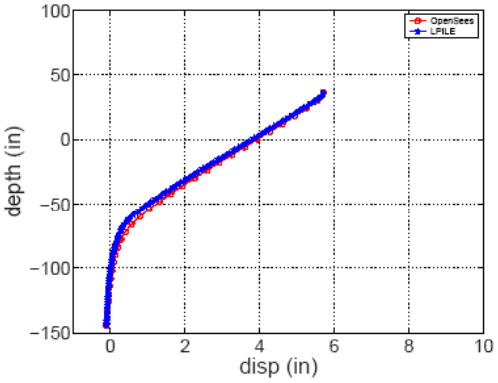




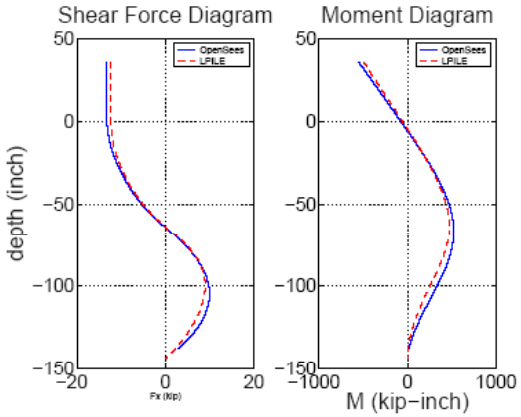


Mobilized Pult

Pushover Pile Analysis using **OpenSees**



Pile displacement (3ft head) loose sand



Shear force and moment diagrams

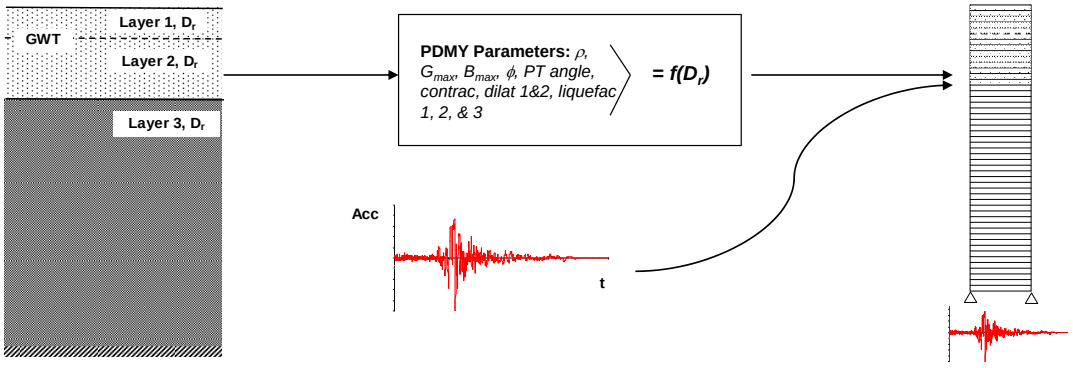
Free-Field Liquefaction Analysis using **OpenSees**

Profile information – layer depths, GWT depth, Dr for each layer, and # of elements per layer → ***openShakeV5.tcl***

Input acceleration time histories – file name, Dt, and # steps → ***earthquakeV5.tcl***

PDMYparametersV5.tcl computes material parameters for each layer based on Dr and position w.r.t. GWT

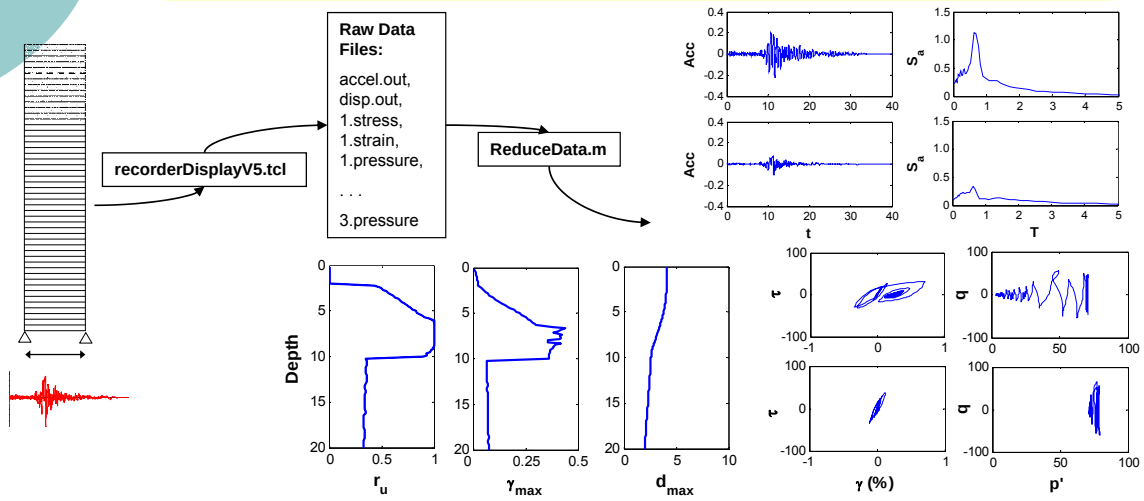
openShakeAnalysisV5.tcl builds nodal mesh, defines layer materials based on PDMY parameters, applies input acceleration to profile base, and performs dynamic analysis



Free-Field Liquefaction Analysis using **OpenSees** (cont.)

recorderDisplayV5.tcl records nodal accelerations and displacements and elemental stresses, strains, and excess porepressures for each analysis time step. Data is saved to files organized by layer and data type

ReduceData.m organizes the data into arrays and produces plots of analysis Results at user-specified locations in the profile



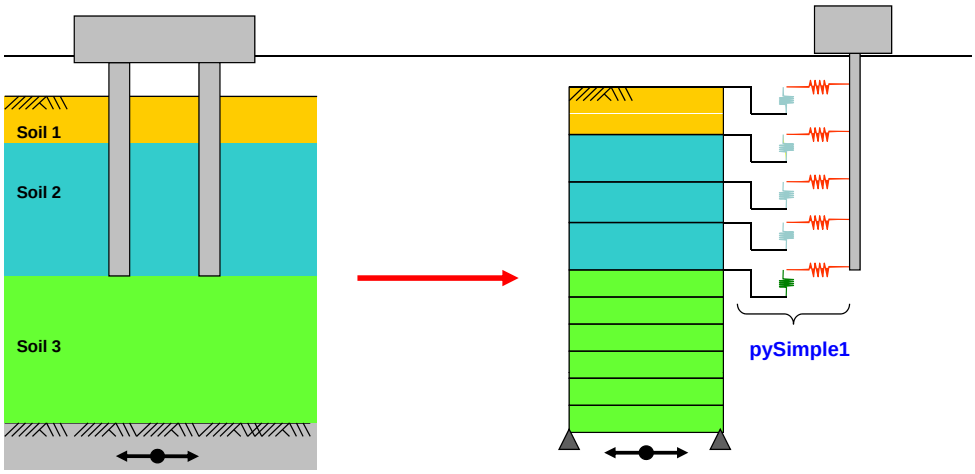
Other useful **tcl** scripts @

- **http://opensees.berkeley.edu/**
- **http://sokocalo.engr.ucdavis.edu/~jeremic**
- **http://cyclic.ucsd.edu/opensees/**
- **http://www.ce.washington.edu/~geotech/opensees/PEER/davis_meeting/**

Take advantage of pre- and post-processors

- Difficult to create **tcl** scripts for complex boundary value geotechnical problems.
  - complex foundation configurations
  - embankments
  - wharves
  - bridges
  - 3-D analysis
- Need to use pre- and post processors to create meshes and visualize results
- **GID**, just one possible alternative ☺

Dynamic Analysis of Piles in **OpenSees**

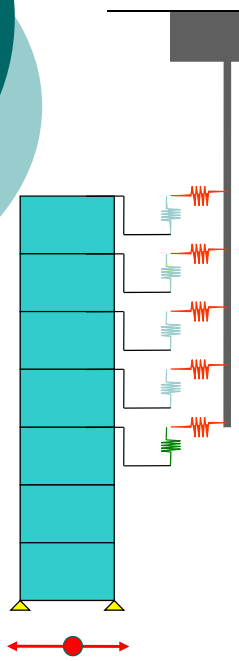


Soil → quad / up94 quad element
PDMY material (+ FluidSolidPorous Material)

Pile → elastic / nonlinear beam column element

Interaction → zeroLength element
pySimple1, tzSimple1, qzSimple1, or liquefiable p-y

Dynamic Analysis of Piles in **OpenSees**



Pre step

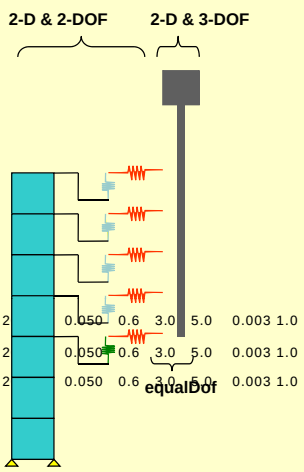
- Gravity analysis without pile information
- Search soil elements connected to p-y springs (matlab)
- Calculate pult, y50 & construct p-y table (matlab)

Pile dynamic analysis

- Gravity analysis (no structure)
- Define structure & connect interface springs
- Apply weight on structure
- Dynamic analysis

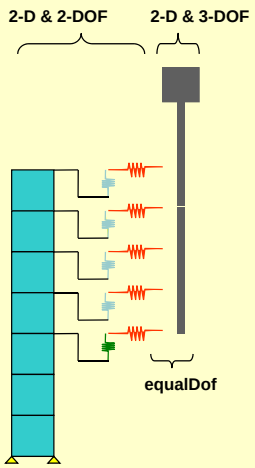
Dynamic Analysis of Piles in **OpenSees**

```
#-------------------------------------------------------------------------
#       Unit: m, s, kN
#-------------------------------------------------------------------------

model BasicBuilder -ndm 2 -ndf 2

# ================================================
# (1)      nodes (soil,py,dashpot,boundary,zeronode)
# ================================================
node       1   0.0000000000   0.0000000000
node       2   0.0000000000   0.5200000000
node       3   0.0000000000   1.0400000000
                    ...
node     569  93.4180000000 27.3000000000
node     570  93.4180000000 27.5600000000
node     571  93.4180000000 27.8200000000

# ================================================
# (2)      define nDmaterial for soil
# ================================================
nDMaterial PressureDependMultiYield   51  2  1.66  100000.0  300000.0   37.000   0.100   80.000   0.500  2   0.050   0.6   3.0   5.0   0.003 1.0
        updateMaterialStage -material     51 -stage 0
nDMaterial PressureDependMultiYield   52  2  1.66  100000.0  300000.0   37.000   0.100   80.000   0.500  2   0.050   0.6   3.0   5.0   0.003 1.0
        updateMaterialStage -material     52 -stage 0
nDMaterial PressureDependMultiYield   53  2  1.66  100000.0  300000.0   37.000   0.100   80.000   0.500  2   0.050   0.6   3.0   5.0   0.003 1.0
        updateMaterialStage -material     53 -stage 0

# ================================================
# (3)      define quadElement for soil (max-10 soils)
# ================================================

element quad     1    251   254    75    74 $eleThick PlaneStrain $matID1 $p 1.66 $gX [expr $gY/(1/1.66)]
element quad     2    254   258    76    75 $eleThick PlaneStrain $matID1 $p 1.66 $gX [expr $gY/(1/1.66)]
element quad     3    258   261    77    76 $eleThick PlaneStrain $matID1 $p 1.66 $gX [expr $gY/(1/1.66)]
element quad     4    261   269    78    77 $eleThick PlaneStrain $matID1 $p 1.66 $gX [expr $gY/(1/1.66)]
                                    ...

element quad   160   265   264   492   493 $eleThick PlaneStrain $matID3 $p 1.66 $gX [expr $gY/(1/1.66)]
element quad   161   264   263   491   492 $eleThick PlaneStrain $matID3 $p 1.66 $gX [expr $gY/(1/1.66)]
element quad   162   263   262   490   491 $eleThick PlaneStrain $matID3 $p 1.66 $gX [expr $gY/(1/1.66)]
```



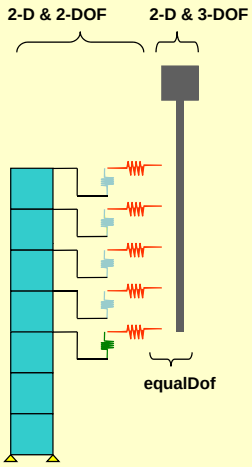
Dynamic Analysis of Piles in **OpenSees**

```
# ================================================
# (4)      define uniaxialMaterial for Py-Tz (imported data)
# ================================================
set channel9 [open "./data/pult.dat" r]        ;# pult imported
set ctr9 0;
foreach line9 [split [read -nonewline $channel9] \n] {
       set lineSize9 [llength $line9]
       set ctr9 [expr $ctr9+1];
       set lineData9($ctr9) $line9
       set lineNumber9 $ctr9
}
close $channel9
#-----------------------------------------------
for {set i 1} {$i <= $lineNumber9} {incr i 1} {
       for {set j 0} {$j < $lineSize9} {incr j 1} {
             set data9($i,$j) [lindex $lineData9($i) $j]
} }
#-----------------------------------------------
set kk0 0
for {set i 1} {$i <= $lineNumber9} {incr i 1} {
       set kk0                  [expr $kk0+1]
       set pyElement($kk0)      [expr int($data9($i,0))]
       set conect1($kk0)        [expr int($data9($i,1))]
       set conect2($kk0)        [expr int($data9($i,2))]
       set pyType($kk0)         [expr int($data9($i,3))]
       set y50($kk0)            $data9($i,4)
       set Cd($kk0)             $data9($i,5)
       set tzType($kk0)         [expr int($data9($i,6))]
       set z50($kk0)            $data9($i,7)
       set pult($kk0)           $data9($i,8)
       set tult($kk0)           $data9($i,9)
       set pyTzQzIndex($kk0)    $data9($i,11)
}
## ================================================
# (5)      define zeroLength element for Py-Tz & Qz
# ================================================
for {set i 1 } { $i <= $lineNumber9 } {incr i 1} {
      if {$pyTzQzIndex($i) == 1} {
            uniaxialMaterial PySimple1 [expr $i+400]  $pyType($i) $pult($i) $y50($i) $Cd($i) 0.01
            uniaxialMaterial TzSimple1 [expr $i+600]  $tzType($i) $tult($i) $z50($i) 0.01
            element zeroLength $pyElement($i) $conect1($i) $conect2($i) -mat [expr $i+400] [expr $i+600] -dir 1 2
      } else {
            uniaxialMaterial PySimple1 [expr $i+400]  $pyType($i) $pult($i) $y50($i) $Cd($i) 0.01
            uniaxialMaterial QzSimple1 [expr $i+600]  $tzType($i) $tult($i) $z50($i) 0.0 0.01
            element zeroLength $pyElement($i) $conect1($i) $conect2($i) -mat [expr $i+400] [expr $i+600] -dir 1 2
}}
```



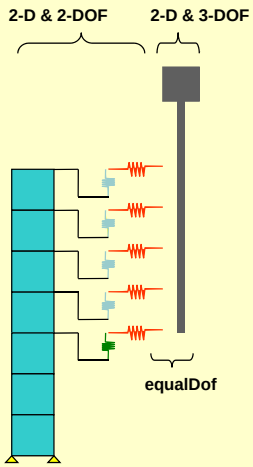
Dynamic Analysis of Piles in **OpenSees**

```
# ================================================
# (6)      define equal DOF (soil to soil)
# ================================================
equalDOF       2    84   1   2
equalDOF       3    85   1   2
equalDOF       4    86   1   2
    ....
equalDOF     470   568   1   2
equalDOF     472   569   1   2
equalDOF     475   570   1   2
equalDOF     479   571   1   2

# ================================================
# (7)      define the fixity
# ================================================
fix       1 1 1
fix      83 1 1
fix     262 1 1
fix     490 1 1

# ================================================
# (8) create the analysis method & analyze
# ================================================
system        ProfileSPD
test          NormDispIncr
algorithm     Newton
constraints   Transformation
integrator    LoadControl 1 1 1 1
numberer      Plain
analysis      Static

analyze       1

# ================================================
# (9) updateMaterialStage
# ================================================
updateMaterialStage -material     51 -stage 1
updateMaterialStage -material     52 -stage 1
updateMaterialStage -material     53 -stage 1
```



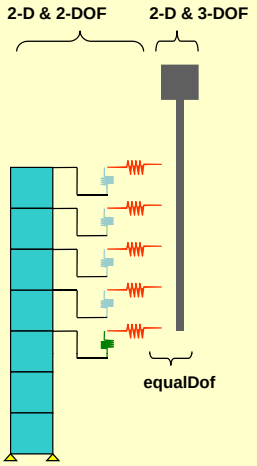
Dynamic Analysis of Piles in **OpenSees**

```
model basic -ndm 2 -ndf 3

# ========================================
# (10) define nodes and mass for pile
# ========================================
 node     110   42.900000  13.520000
 node     113   42.900000  13.780000
    ...
 node     488   50.518000  35.100000
 node     489   52.429000  33.410000

# ========================================
# (11) define nodes and mass for pile
# ========================================
 mass     110      0.09410      0.09410      0.09410
 mass     113      0.09410      0.09410      0.09410
             ...
 mass     486      0.59100      0.59100      0.59100
 mass     487   328.60000   328.60000   328.60000

# ================================
# (12) define element and material for pile
# ================================

#--- elastic pile
geomTransf Linear 1
element elasticBeamColumn     163    334    388 21.09 68.95e6 20.08 1
element elasticBeamColumn     164    388    474 21.09 68.95e6 20.08 1
element elasticBeamColumn     165    474    487 21.09 68.95e6 20.08 1

#--- fiber section pile
set integration_points    5
set section_tag           1
set trans_tag             1
set alum_fy             1.3e5
set alum_E             6.89e7

uniaxialMaterial Steel01  941  $alum_fy $alum_E  $alumHardening_ratio
section Fiber 1 {
          patch circ 941  $numSubDiv_circ  $numSubDiv_rad  $yCenter  $zCenter  $intRad  $extRad  $starAng  $endAng   ;# core concrete fibers
}

 element nonlinearBeamColumn     169    486    485 $integration_points $section_tag $trans_tag
 element nonlinearBeamColumn     170    485    484 $integration_points $section_tag $trans_tag
 element nonlinearBeamColumn     171    484    483 $integration_points $section_tag $trans_tag
 element nonlinearBeamColumn     172    483    482 $integration_points $section_tag $trans_tag
```



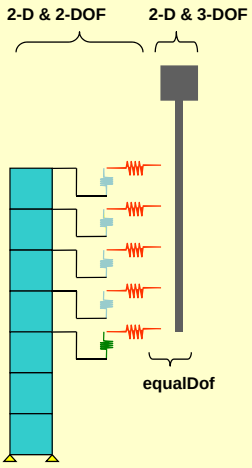
Dynamic Analysis of Piles in **OpenSees**

```
# ================================================
# (13)      define equal DOF (soil to pile)
# ================================================
 equalDOF    111    110   1   2
 equalDOF    112    113   1   2
       ...
 equalDOF    477    476   1   2
 equalDOF    478    480   1   2

# ================================
# (14) settime and wipeAnalysis
# ================================
wipeAnalysis
loadConst -time 0.0

# ================================
# (15) apply the weight of structures
# ================================
set numStep_sWgt 100
pattern Plain 1 "Linear" {
      load     110  0.000000 [expr   0.094100 /(-1/9.81)/$numStep_sWgt] 0.000000
      load     113  0.000000 [expr   0.094100 /(-1/9.81)/$numStep_sWgt] 0.000000
                ...
      load     486  0.000000 [expr   0.591000 /(-1/9.81)/$numStep_sWgt] 0.000000
      load     487  0.000000 [expr 328.600000 /(-1/9.81)/$numStep_sWgt] 0.000000
}

# ================================
# (16) Recorder (selfWgt steps)
# ================================
recorder Node      -file ./data/$f1/displacement_gStep.out -time -node all -dof 1 2 disp
recorder Element -file ./data/$f1/stress1_gStep.out   -time -eleRange      1     162 material 1 stress

# ================================
# (17) analysis method
# ================================
constraints   Penalty 1.0e12 1.0e12
test           NormDispIncr 1e-6 10 0
algorithm     Newton
numberer      Plain
system        ProfileSPD
integrator     LoadControl 1 1 1 1
analysis      Static

analyze  $numStep_sWgt
```

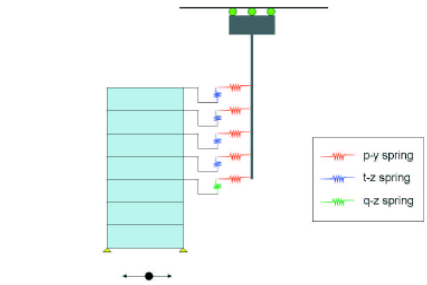


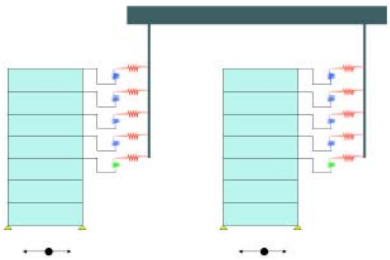
Dynamic Analysis of Piles in **OpenSees**

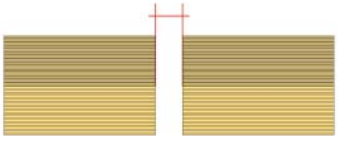
```
# ================================================
# (18)      earthquake excitation
# ================================================
wipeAnalysis
loadConst -time 0.0

set accelSeries1 "Path -filePath ./OpenBaseMotions/CFG2_ax_base_02g_avg.txt -dt 0.0127 -factor 9.81"
pattern UniformExcitation  2 1 -accel $accelSeries1

# ================================================
# (19)      recorder  (earthquake step)
# ================================================
remove recorders
recorder Node     -file ./data/$f1/displacement.out     -time -node all -dof 1 2 disp
recorder Node     -file ./data/$f1/acceleration.out     -time -node all -dof 1 2 accel
recorder Element -file ./data/$f1/pileForce.out         -time -eleRange     163   292 force
recorder Element -file ./data/$f1/pyTzForce.out         -time -eleRange     293   404 force
recorder Element -file ./data/$f1/pyTzDeformation.out -time -eleRange     293    404 deformation
recorder Element -file ./data/$f1/stress1.out           -time -eleRange      1     162 material 1 stress
recorder Element -file ./data/$f1/stress2.out           -time -eleRange      1     162 material 2 stress
recorder Element -file ./data/$f1/stress3.out           -time -eleRange      1     162 material 3 stress
recorder Element -file ./data/$f1/stress4.out           -time -eleRange      1     162 material 4 stress
recorder Element -file ./data/$f1/strain1.out           -time -eleRange      1     162 material 1 strain
recorder Element -file ./data/$f1/strain2.out           -time -eleRange      1     162 material 2 strain
recorder Element -file ./data/$f1/strain3.out           -time -eleRange      1     162 material 3 strain
recorder Element -file ./data/$f1/strain4.out           -time -eleRange      1     162 material 4 strain
recorder Element -file ./data/$f1/pressure1.out         -time -eleRange      1     162 material 1 pressure
recorder Element -file ./data/$f1/pressure2.out         -time -eleRange      1     162 material 2 pressure
recorder Element -file ./data/$f1/pressure3.out         -time -eleRange      1     162 material 3 pressure
recorder Element -file ./data/$f1/pressure4.out         -time -eleRange      1     162 material 4 pressure

# ================================
# (20)                analysis method
# ================================
constraints    Penalty 1.0e12  1.0e12
test           NormDispIncr 1e-4 10 0
algorithm      Newton
numberer       Plain
system         ProfileSPD
integrator     Newmark 0.6 0.3025 0.0 0.0 0.001 0.0
analysis       Transient

analyze        4096 0.0127
```

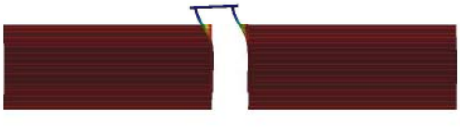


Dynamic Analysis of Piles in **OpenSees** using **GID**



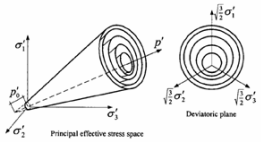






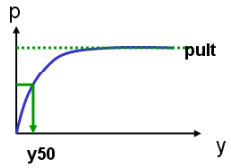
SPSI modeling in OpenSees

Soil model (by Yang & Elgamal)



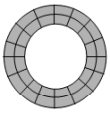
pressure dependent multi-yield material

P-y spring model (by Boulanger)

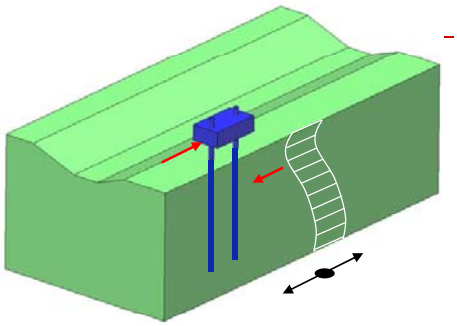


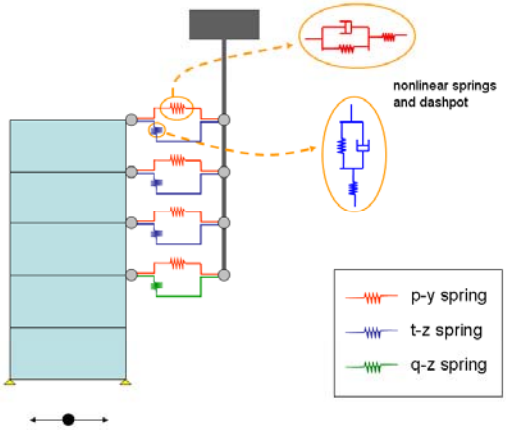
API (1993)

Pile model

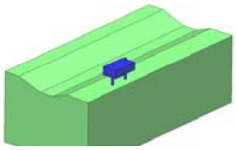


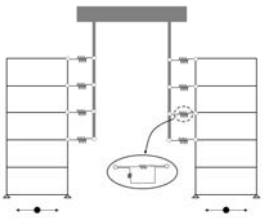
nonlinear beam column elements using fiber section

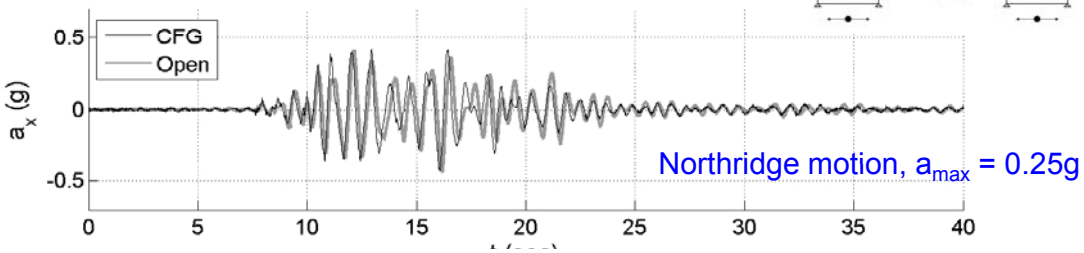




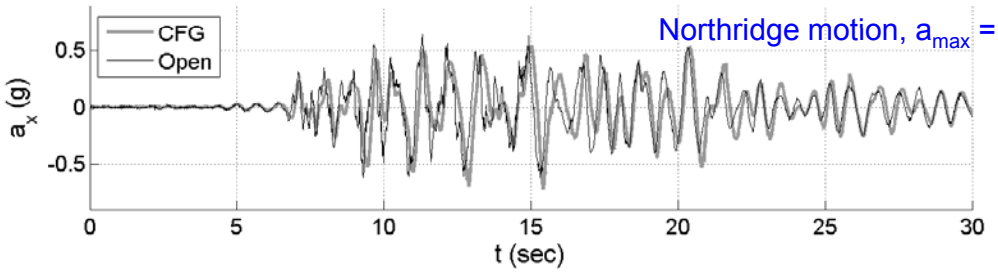
Two-column Bent





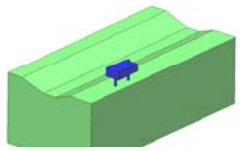


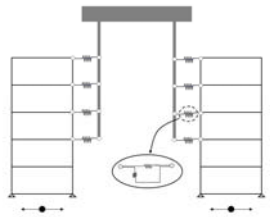
Northridge motion, a_{max} = 0.25g

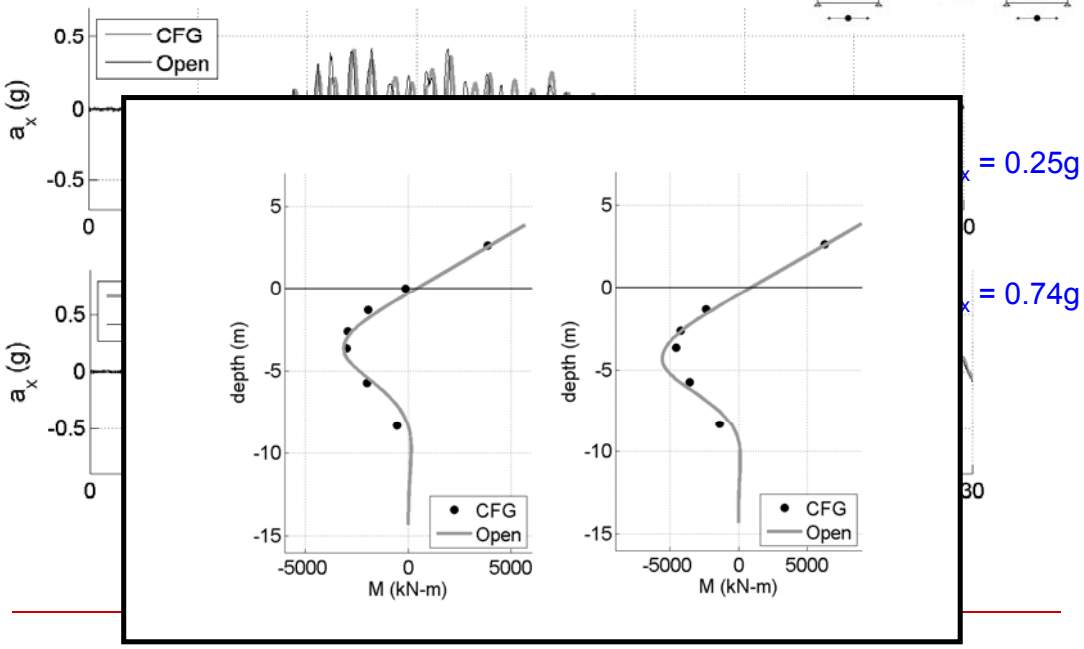


Northridge motion, a_{max} = 0.74g

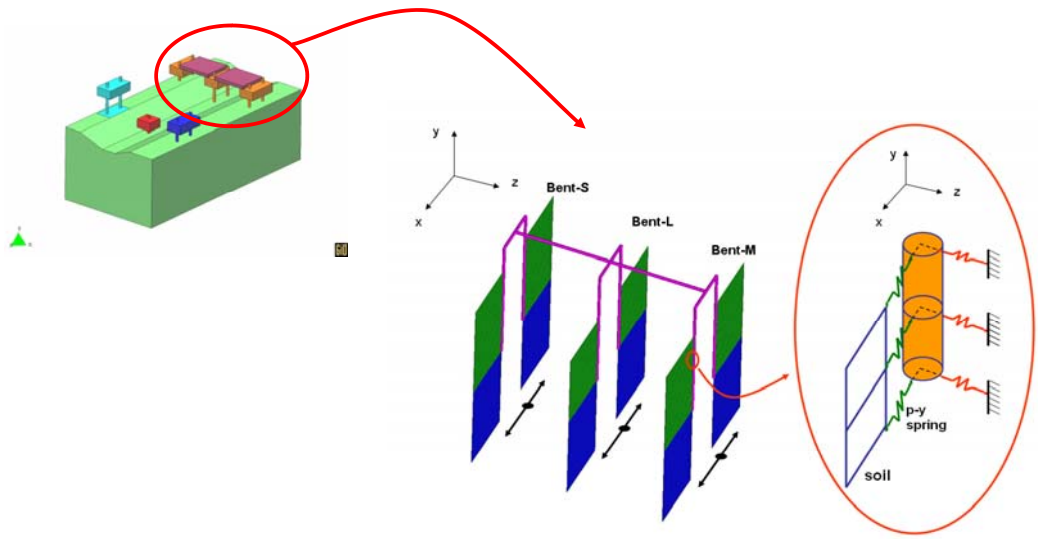
Two-column Bent



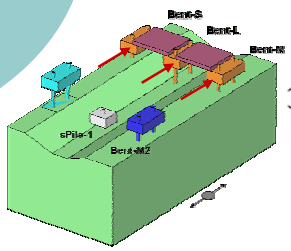


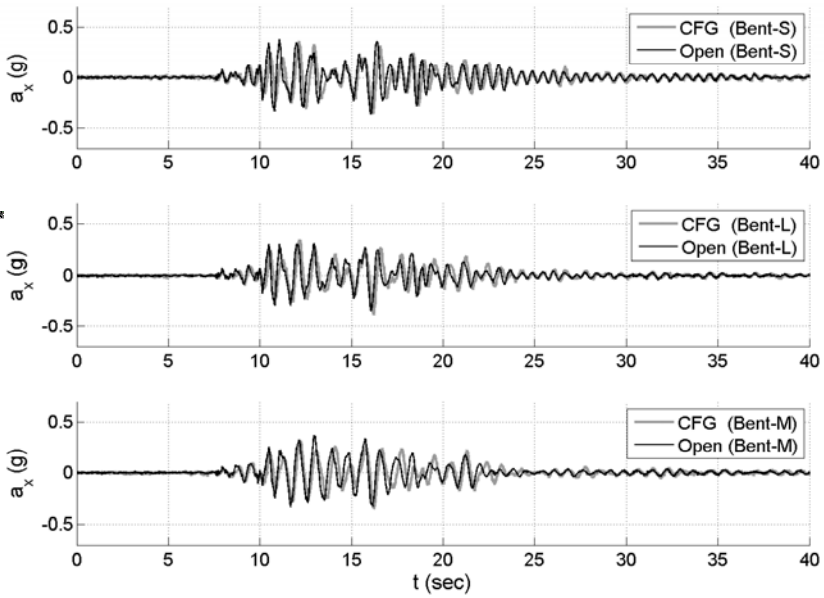


Two-span bridge

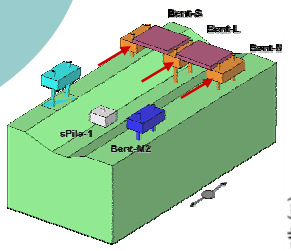


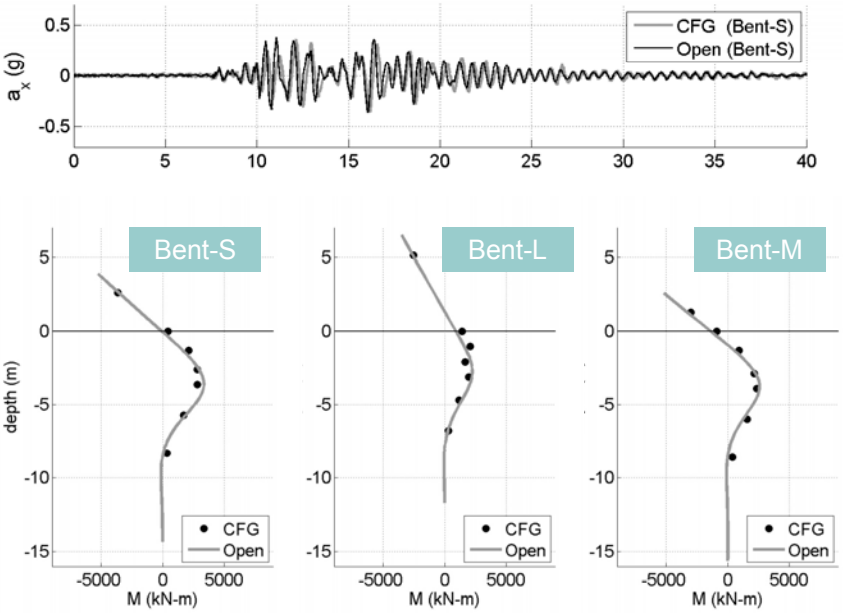
Two-span bridge





Two-span bridge





SPSI of a complete bridge system on liquefiable soil

- Five-span bridge
- pile group foundation
- abutment
- liquefiable soil / various layers

Soil-pile-bridge-abutment interaction

- lateral spreading
- earthquake intensities
- uncertainties

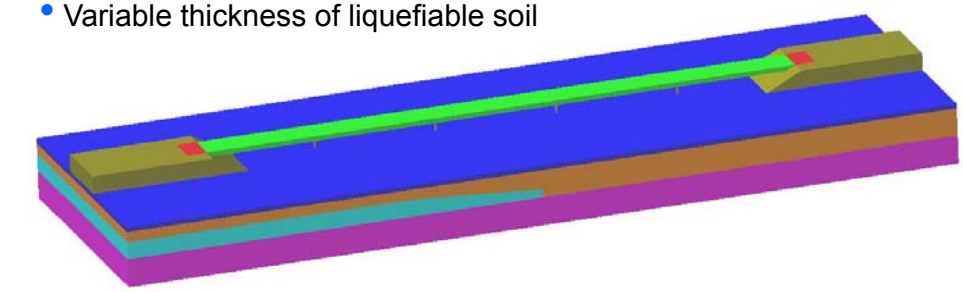
Just a couple of runs ☺

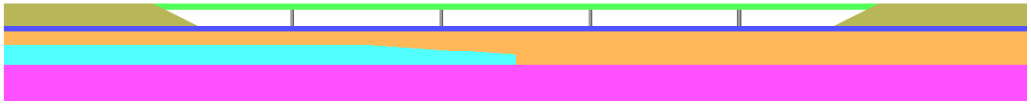
Thousands of runs!!! ☹

Numerical Simulation of a complete bridge system using OpenSees → **Performance Based Earthquake Engineering**

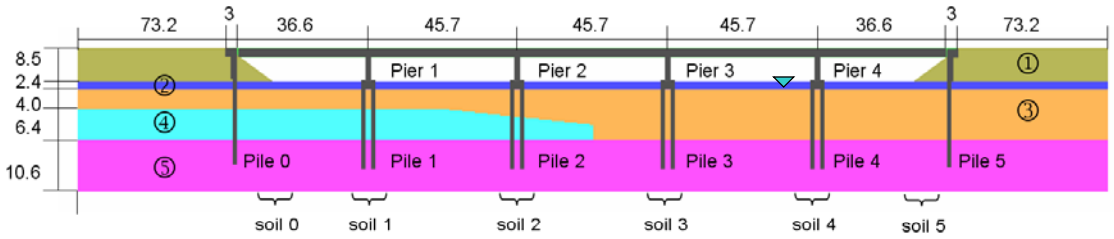
Example: Bridge on liquefiable soil deposit

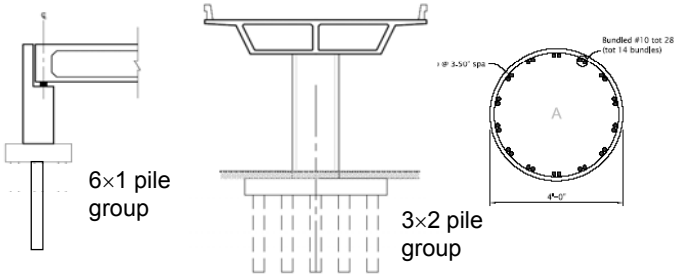
- Five-span bridge
- Approach embankments
- Variable thickness of liquefiable soil





Target bridge system





6×1 pile group

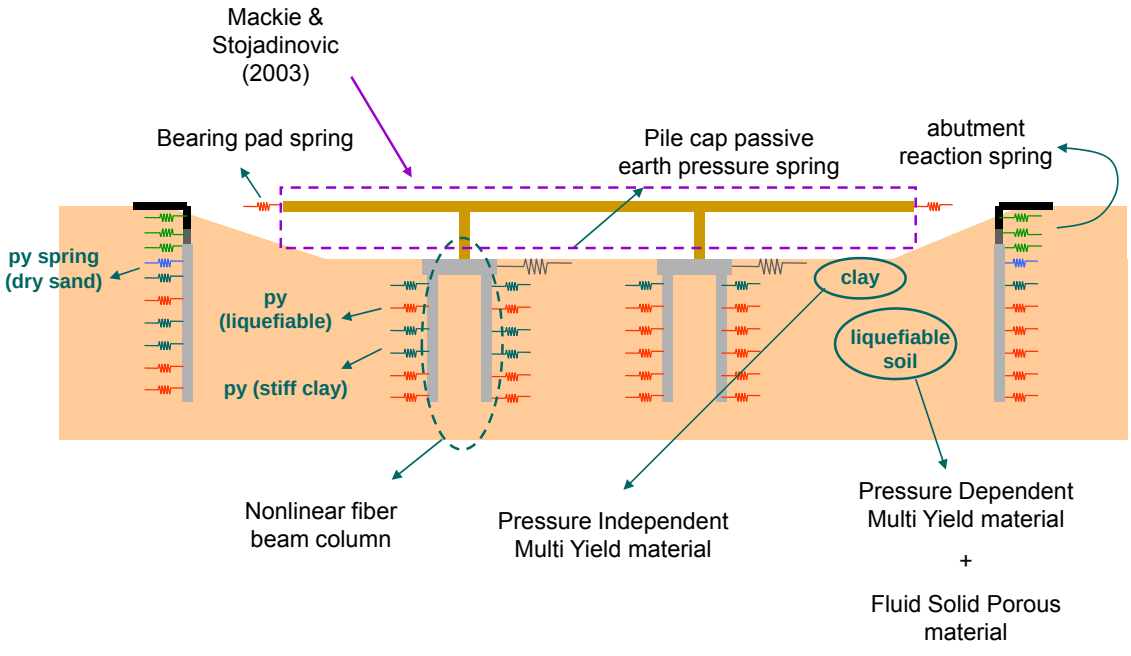
3×2 pile group

open-ended steel pipe pile

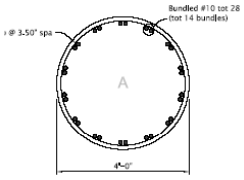
(ϕ = 2 ft, t = 0.5 in.)

① embankment
② med. stiff clay
③ loose-med sand
④ stiff clay
⑤ dense sand

Numerical modeling for target bridge system

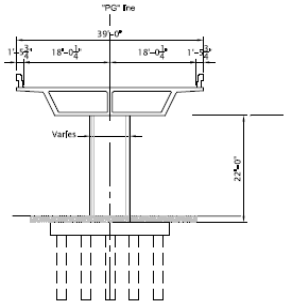


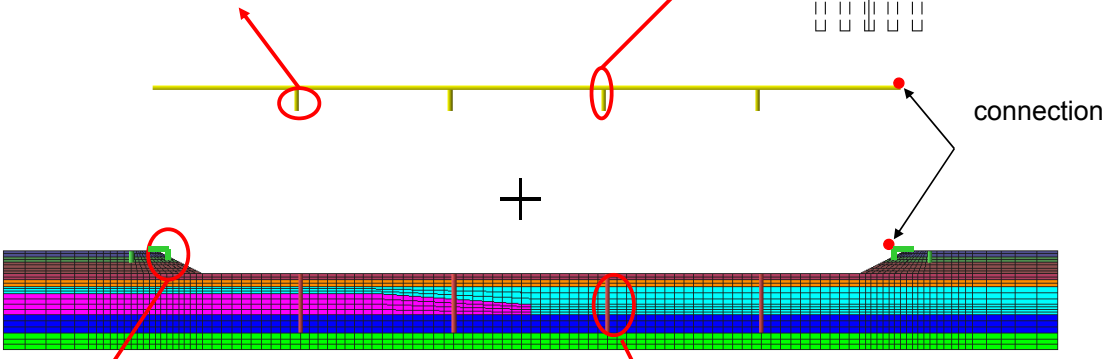
reinforce con'c column
(column A – 4 ft)



prestressed
reinforced con'c
bridge

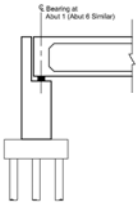
(type 1 – 22ft)





connection

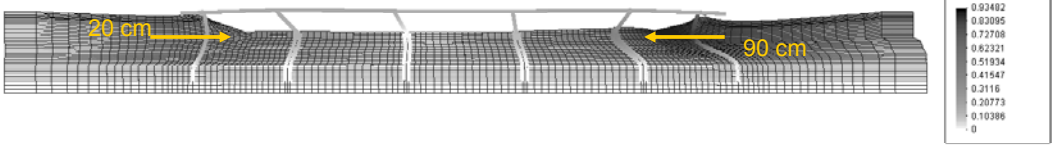
abutment



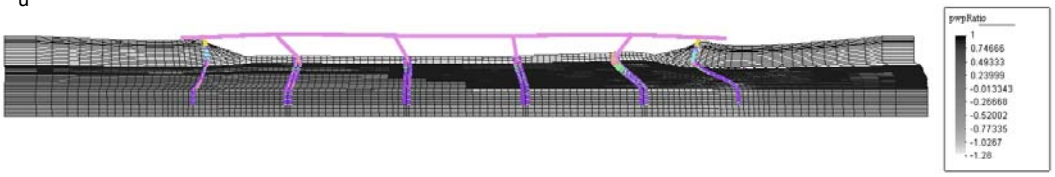
Pile group
3x2

Bridge system response

displacement

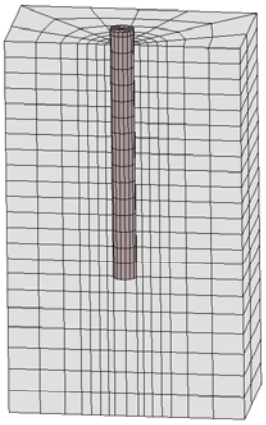


r_u

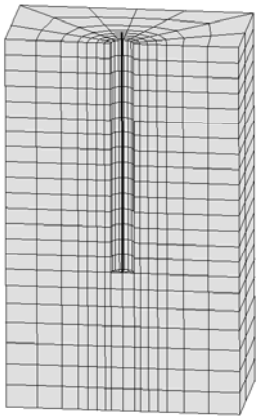


Erzincan, Turkey 1992 (a_{max} = 0.70g)

3D Pile Analysis



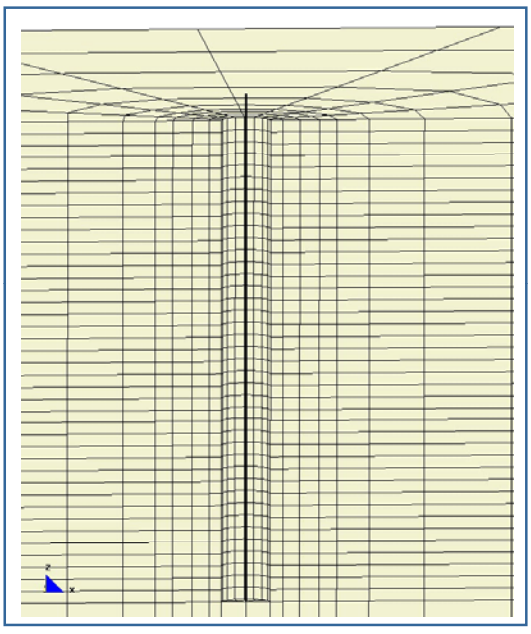
Solid-Solid Model



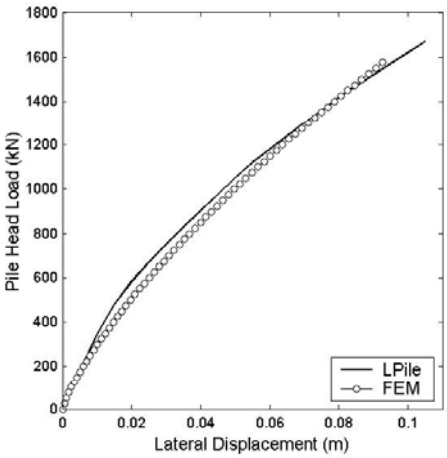
Beam-Solid Model

Laterally Loaded Piles (solid-beam contact element)

- Perform numerical load test



- Compare results



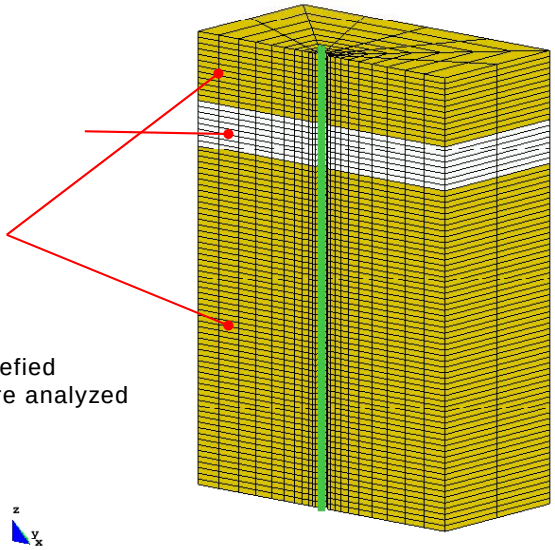
3x Magnification

3D Modeling Approach

The soil is modeled with brick elements and a Drucker-Prager constitutive model to capture pressure-dependent strength

Liquefied Layer:
reduced strength

Unliquefied Layers:
dense Sand

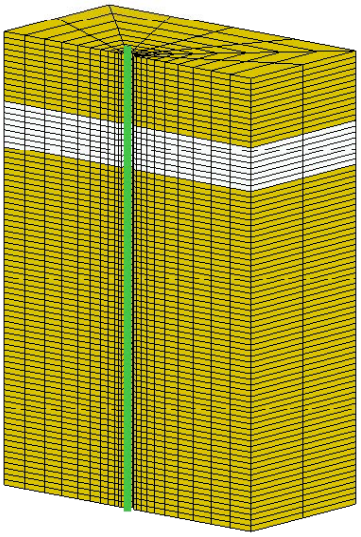


Various combinations of liquefied layer depth and thickness are analyzed

3D Modeling Approach

The pile is modeled with beam-column elements

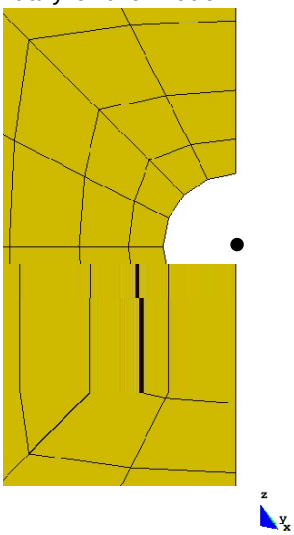
Three pile diameters are considered (0.61, 1.37, 2.5 m)

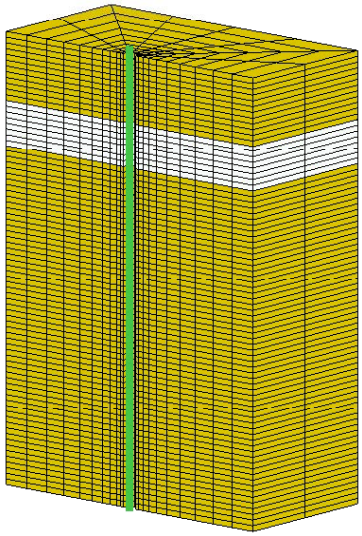


3D Modeling Approach

Beam-solid contact elements are used to model the soil-pile interface

Lateral spreading is modeled by applying a free-field displacement profile to the boundary of the model

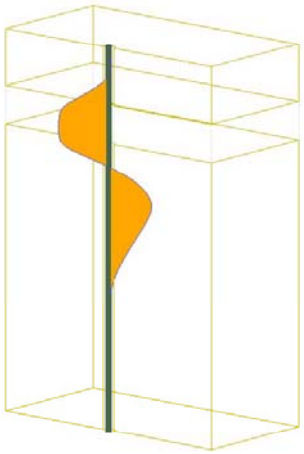


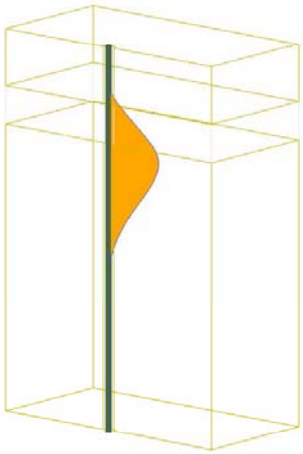


Beam-Solid Contact Elements

The beam-solid contact elements enable the use of standard beam-column elements for the pile

This allows for simple recovery of the shear force and bending moment demands placed upon the pile

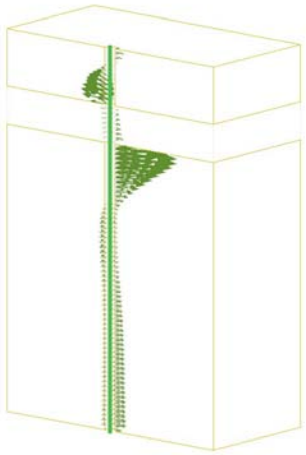


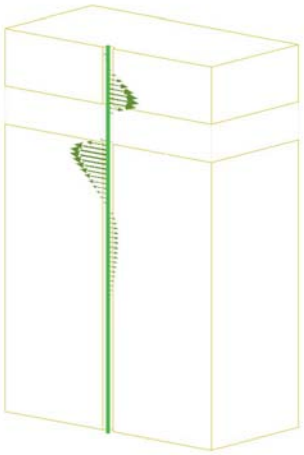


Beam-Solid Contact Elements

The beam-solid contact elements enable the use of standard beam-column elements for the pile

Additionally, the surface traction acting on the pile-soil interface can be recovered and resolved into the forces applied by the soil to the pile



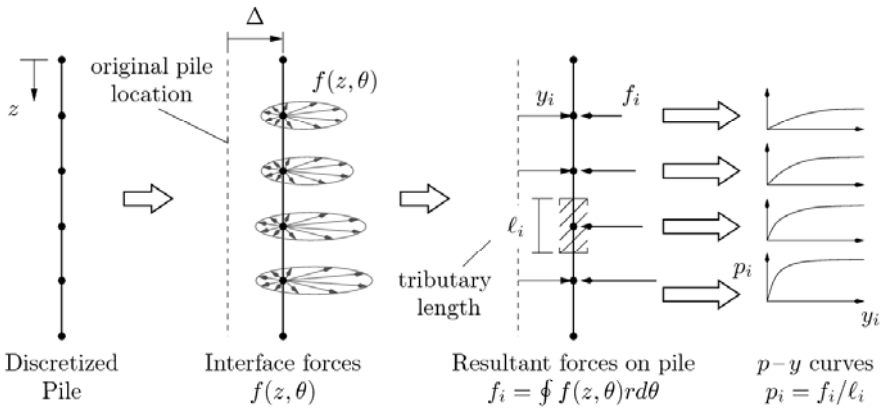


Computing *p-y* Curves: Rigid Kinematic

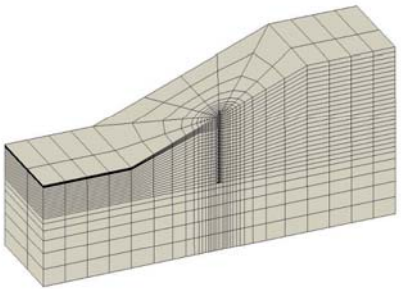
Work with 3D FE models has shown that use of a general pile deformation creates *p-y* curves which are influenced by the selected pile kinematics

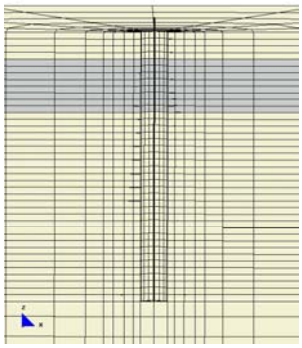
A rigid pile kinematic is used to evenly activate the soil response with depth and to obtain *p-y* curves which are free from the influence of pile kinematics, reflecting only the response of the soil.

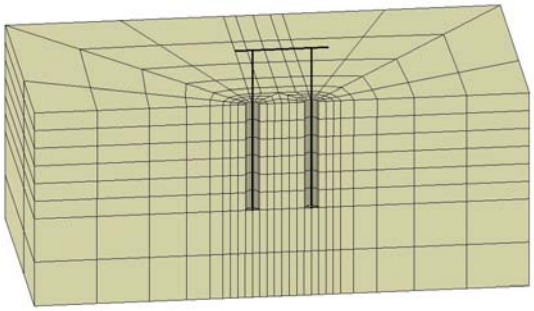
Computational process



Other Applications: Piles in Sloping Ground, Bridge bent analysis







Questions?