30 | fix node # 3 dofs all ;
31 | fix node # 4 dofs all ;
32 | fix node # 5 dofs all ;
33 | fix node # 6 dofs all ;
34 | ... 
35 | ... 
36 | fix node # 80 dofs all ;
37 | 
38 | new loading stage "self_weight";
39 | add acceleration field # 1 ax = 0*g ay = 0*g az = 1*m/s^2;
40 | add load # 1 to element # 1 type self_weight use acceleration field # 1;
41 | add load # 2 to element # 2 type self_weight use acceleration field # 1;
42 | add load # 3 to element # 3 type self_weight use acceleration field # 1;
43 | add load # 4 to element # 4 type self_weight use acceleration field # 1;
44 | add load # 5 to element # 5 type self_weight use acceleration field # 1;
45 | add load # 6 to element # 6 type self_weight use acceleration field # 1;
46 | ... 
47 | ... 
48 | add load # 400 to element # 400 type self_weight use acceleration field # 1;
49 | 
50 | define algorithm With_no_convergence_check ;
51 | define solver ProfileSPD;
52 | define load factor increment 1;
53 | simulate 1 steps using static algorithm;
54 | 
55 | bye;

The ESSI model fei/DSL files for this example can be downloaded [here](#).
707.15 One Dimensional DRM Model

Problem description:

A simple 1D DRM model is shown in Fig. (707.44). The "DRM element", "Exterior node" and "Boundary node" are required to be designated in the DRM HDF5 input. The format and script for the HDF5 input is available in DSL/input manual.

![Figure 707.44: 1D DRM model.](image)

Numerical model:

ESSI model fei/DSL file:

```plaintext
model name "DRM";

//Material for soil
add material # 1 type linear_elastic_isotropic_3d_LT
mass_density = 2000*kg/m^3
elastic_modulus = 1300*MPa
poisson_ratio = 0.3;

//Material for DRM layer
add material # 2 type linear_elastic_isotropic_3d_LT
mass_density = 2000*kg/m^3
elastic_modulus = 1300*MPa
poisson_ratio = 0.3;

//Material for exterior layer
add material # 3 type linear_elastic_isotropic_3d_LT
mass_density = 2000*kg/m^3
elastic_modulus = 1300*MPa
poisson_ratio = 0.3;
```

add node # 1 at (0.00*m, 0.00*m, 0.00*m) with 3 dofs;
add node # 2 at (5.00*m, 0.00*m, 0.00*m) with 3 dofs;
add node # 3 at (5.00*m, 5.00*m, 0.00*m) with 3 dofs;
add node # 4 at (0.00*m, 5.00*m, 0.00*m) with 3 dofs;
add node # 5 at (5.00*m, 0.00*m, 50.00*m) with 3 dofs;
add node # 6 at (5.00*m, 0.00*m, 5.00*m) with 3 dofs;
add node # 52 at (0.00*m, 5.00*m, -5.00*m) with 3 dofs;

//
add element # 1 type 8NodeBrickLT with nodes(1, 4, 3, 2, 24, 44, 34, 6) use ←
material # 1;
add element # 2 type 8NodeBrickLT with nodes(24, 44, 34, 6, 23, 43, 33, 7) use ←
material # 1;
...
add element # 12 type 8NodeBrickLT with nodes(48, 47, 45, 46, 52, 51, 49, 50) ←
use material # 3;

//
fix node # 1 dofs uy;
fix node # 1 dofs uz;

Figure 707.45: 1D DRM model.
The ESSI model fei/DSL files for this example can be downloaded [here](#).
The same model for this example with 27NodeBrickLT can be downloaded [here](#).

**Long 1D DRM model 1000:1**

To show the wave propagation explicitly, a long 1D model (1000:1) similar to the 1D DRM model above was made in this section.

The model description is same to Fig.(707.44) except this model use far more soil elements.

The general view is shown in Fig.(707.46) below.

There is still now outgoing waves at the exterior layers, which is shown in Fig(707.47).

The ESSI model fei/DSL files for this example can be downloaded [here](#).

The results can also be seen in this [animation](#).
Figure 707.46: Long 1D DRM model
Figure 707.47: Long 1D DRM model: exterior layer
707.16 Three Dimensional DRM Model

Problem description:

As shown in Fig.(707.48), the DRM layer is used to add the earthquake motion.

![Diagram of 3D Domain Reduction Method example](image.png)

Figure 707.48: The diagram for 3D Domain Reduction Method example.

Numerical result:

ESSI model fei/DSL file:

```plaintext
model name "DRM" ;

//Material for soil
add material # 1 type linear_elastic_isotropic_3d_LT
mass_density = 2000*kg/m^3
elastic_modulus = 1300*MPa
poisson_ratio = 0.3;

//Material for DRM layer
add material # 2 type linear_elastic_isotropic_3d_LT
mass_density = 2000*kg/m^3
elastic_modulus = 1300*MPa
poisson_ratio = 0.3;
```
Figure 707.49: Diagram for the 3D DRM model.

```
//Material for exterior layer
add material # 3 type linear_elastic_isotropic_3d_LT
mass_density = 2000*kg/m^3
elastic_modulus = 1300*MPa
poisson_ratio = 0.3;

//
add node # 1 at ( 0.00*m, 0.00*m, 0.00*m) with 3 dofs;
add node # 2 at ( 50.00*m, 0.00*m, 0.00*m) with 3 dofs;
add node # 3 at ( 5.00*m, 0.00*m, 0.00*m) with 3 dofs;
add node # 4 at ( 10.00*m, 0.00*m, 0.00*m) with 3 dofs;
add node # 5 at ( 15.00*m, 0.00*m, 0.00*m) with 3 dofs;
add node # 6 at ( 20.00*m, 0.00*m, 0.00*m) with 3 dofs;
add node # 7 at ( 25.00*m, 0.00*m, 0.00*m) with 3 dofs;
...
add node # 2925 at ( 55.00*m, 55.00*m, -5.00*m) with 3 dofs;

//
add element # 1 type 8NodeBrickLT with nodes( 1, 40, 41, 3, 150, 441, 603, 151)
```
use material # 1;
add element # 2 type 8NodeBrickLT with nodes( 3, 41, 50, 4, 151, 603, 684, 160) ←
use material # 1;
...
add element # 2352 type 8NodeBrickLT with nodes( 2925, 2924, 2922, 2923, 2921, ←
 2920, 2918, 2919) use material # 3;

//
fix node # 1332 dofs all;
fix node # 1334 dofs all;
...  
...  
fix node # 2924 dofs all;

new loading stage "3D";
add domain reduction method loading # 1
  hdf5_file = "input.hdf5";
define algorithm With_no_convergence_check;
define solver ProfileSPD;
define dynamic integrator Newmark with gamma = 0.5 beta = 0.25;
simulate 999 steps using transient algorithm time_step = 0.01*s;
bye;

The ESSI model fei/DSL files for this example can be downloaded here.
The same model for this example with 27NodeBrickLT can be downloaded here.
707.17 ShearBeam Element, Pisano Material

Problem description:

In the element type "ShearBeamLT", only one Gauss point exists. ShearBeamLT element was used here to test the Pisano material model.

Vertical force $F_z$ was used to apply confinement to the element. Then, cyclic force $F_x$ is used to load. point.

![Figure 707.50: ShearBeam element.](image)

Results

Resulting stress-strain relationship is shown in Fig. (707.51).

ESSI model fei/DSL file:

```plaintext
model name "pisanoLT";
add node # 1 at (0*m,0*m,0*m) with 3 dofs;
add node # 2 at (0*m,0*m,1*m) with 3 dofs;
fix node # 1 dofs all;
```
Figure 707.51: Shear stress-strain response.

```plaintext
fix node # 2 dofs uy;

add material # 1 type New_PisanoLT
mass_density = 2000*kg/m^3
elastic_modulus_1atm = 325*MPa poisson_ratio = 0.3
M_in = 1.4 kd_in = 0.0 xi_in = 0.0 h_in = 700 m_in = 0.7
initial_confining_stress = 0*kPa n_in = 0 a_in = 0.0 eplcum_cr_in = 1e-6;

add element # 1 type ShearBeamLT with nodes (1, 2) 
cross_section = 1*m^2 use material # 1;

new loading stage "confinement";

add load # 1 to node # 2 type linear Fz = -200*kN;
define load factor increment 0.01;
define algorithm With_no_convergence_check ;
define solver UMFPack;
simulate 100 steps using static algorithm;

new loading stage "test01";
gamma_max = 3e-3;
add imposed motion # 2 to node # 2 dof ux

displacement_scale_unit = gamma_max*m displacement_file = "input_sine.txt"
velocity_scale_unit = gamma_max*m/s velocity_file = "input_sine.txt"
acceleration_scale_unit = gamma_max*m/s^2 acceleration_file = "input_sine.txt";

define load factor increment 0.0005;
```
34 define algorithm With_no_convergence_check;
35 define solver UMFPack;
36 simulate 2000 steps using static algorithm;
37 bye;

The ESSI model fei/DSL files for this example can be downloaded here.
707.18 8NodeBrickLT Element, Drucker-Prager Material, Armstrong-Frederick Rotational Kinematic Hardening

Problem description:

This example is used to test the materials properties, such as $G/G_{\text{max}}$ against strains. The element type is 8NodeBrickLT. And there are two stages of loading. The first loading stage is confinement and the second loading stage is shearing.

The boundary condition is specially designed such that each Gauss point has the same stress state.

Results

Resulting stress-strain relationship is shown in Fig.(707.52).

Figure 707.52: Shear stress-strain response.

ESSI model fei/DSL file:

```plaintext
1 // Drucker Prager Armstrong Frederick
2 // This model is created by Jose.
3 model name "druckeraf";
4 // Parameters:
5 phi = 5;
6 ha = 1000;
7 cr = 973;
```


```plaintext
9 \text{gam} = 0.01; 
10 \text{Ncyc} = 5; 
11 \text{Nsteps} = 1000; 
12 \text{H}=1; 
13 \text{vp}=1000*m/s; 
14 \text{vs}=500*m/s; 
15 \rho=2000*kg/m^3; 
16 \text{p}_0 = 250*kPa; 
17 \text{G} = \rho*vs^2; 
18 \text{M} = \rho*vp^2; 
20 \text{E} = \text{G}*(3*\text{M}-4*\text{G})/(\text{M}-\text{G}); 
21 \text{nu} = (\text{M}-2*\text{G})/(2*\text{M}-2*\text{G}); 
22 \text{K}0 = 1.0; 
23 \text{phirad} = \text{pi}*\text{phi}/180; 
24 \text{M} = 6*\sin(\text{phirad})/(3-\sin(\text{phirad})); 
25 // Define the material: 
26 \text{add material # 1 type DruckerPragerArmstrongFrederickLT} 
27 \text{mass density} = 0*kg/m^3 
28 \text{elastic modulus} = \text{E} 
29 \text{poisson ratio} = \text{nu} 
30 \text{druckerprager k} = \text{M} 
31 \text{armstrong frederick ha} = \text{ha*Pa} 
32 \text{armstrong frederick cr} = \text{cr*Pa} 
33 \text{isotropic hardening rate} = 0*\text{E} 
34 \text{initial confining stress} = 1*\text{Pa}; 
35 // define the node: 
36 \text{add node # 1 at (0*m,0*m,1*m) with 3 dofs}; 
37 \text{add node # 2 at (1*m,0*m,1*m) with 3 dofs}; 
38 \text{add node # 3 at (1*m,1*m,1*m) with 3 dofs}; 
39 \text{add node # 4 at (0*m,1*m,1*m) with 3 dofs}; 
40 \text{add node # 5 at (0*m,0*m,0*m) with 3 dofs}; 
41 \text{add node # 6 at (1*m,0*m,0*m) with 3 dofs}; 
42 \text{add node # 7 at (1*m,1*m,0*m) with 3 dofs}; 
43 \text{add node # 8 at (0*m,1*m,0*m) with 3 dofs}; 
44 // add equal degree of freedom in three directions 
45 \text{add constraint equal dof with master node # 2 and slave node # 3 dof to \leftarrow constrain \text{ux}}; 
46 \text{add constraint equal dof with master node # 2 and slave node # 6 dof to \leftarrow constrain \text{ux}}; 
47 \text{add constraint equal dof with master node # 2 and slave node # 7 dof to \leftarrow constrain \text{ux}}; 
48 \text{add constraint equal dof with master node # 3 and slave node # 4 dof to \leftarrow constrain \text{uy}};
```

add constraint equal dof with master node # 3 and slave node # 8 dof to constrain uy;
add constraint equal dof with master node # 3 and slave node # 7 dof to constrain uy;
add constraint equal dof with master node # 1 and slave node # 2 dof to constrain uz;
add constraint equal dof with master node # 1 and slave node # 3 dof to constrain uz;
add constraint equal dof with master node # 1 and slave node # 4 dof to constrain uz;

// Define the element.
add element # 1 type 8NodeBrickLT with nodes (1, 2, 3, 4, 5, 6, 7, 8) use material # 1;

new loading stage "confinement";
fix node # 1 dofs ux uy;
fix node # 2 dofs uy;
fix node # 4 dofs ux;

fix node # 5 dofs ux uy uz;
fix node # 6 dofs uy uz;
fix node # 7 dofs uz;
fix node # 8 dofs ux uz;

sigma_z = -3*p0/(1+2*K0);
sigma_x = K0*sigma_z;
sigma_y = K0*sigma_z;

//Z-face
add load # 1 to node # 1 type linear Fz = sigma_z*m^2/4;
add load # 2 to node # 2 type linear Fz = sigma_z*m^2/4;
add load # 3 to node # 3 type linear Fz = sigma_z*m^2/4;
add load # 4 to node # 4 type linear Fz = sigma_z*m^2/4;

//X-face
add load # 5 to node # 2 type linear Fx = sigma_x*m^2/4;
add load # 6 to node # 6 type linear Fx = sigma_x*m^2/4;
add load # 7 to node # 7 type linear Fx = sigma_x*m^2/4;
add load # 8 to node # 3 type linear Fx = sigma_x*m^2/4;

add load # 9 to node # 3 type linear Fy = sigma_y*m^2/4;
add load # 10 to node # 7 type linear Fy = sigma_y*m^2/4;
add load # 11 to node # 8 type linear Fy = sigma_y*m^2/4;
add load # 12 to node # 4 type linear Fy = sigma_y*m^2/4;

Nsteps_static=100;
define load factor increment 1/Nsteps_static;

define solver UMFPack;
define convergence test Norm_Displacement_Increment
tolerance = 1e-6
maximum_iterations = 100
verbose_level = 4;
define algorithm Newton;

define NDMaterialLT constitutive integration algorithm Euler_One_Step
yield_function_relative_tolerance = 0.002
stress_relative_tolerance = 0.002
maximum_iterations = 1000;

simulate Nsteps_static steps using static algorithm;

new loading stage "shearing";
compute reaction forces;
add load # 13 to node # 1 type from_reactions;
add load # 14 to node # 4 type from_reactions;
free node # 1 dofs ux;
free node # 4 dofs ux;
fix node # 3 dofs uy;
fix node # 6 dofs ux;
fix node # 7 dofs ux uy;
fix node # 8 dofs uy;
add constraint equal dof with master node # 1 and slave node # 3 dof to constrain ux;
add constraint equal dof with master node # 1 and slave node # 4 dof to constrain ux;
add constraint equal dof with master node # 1 and slave node # 2 dof to constrain ux;
remove constraint equaldof node # 6;
remove constraint equaldof node # 7;
remove constraint equaldof node # 8;

n = 1;
while(n<=1)
{
    add load # 14+n to node # n type path_time_series
    Fx = 170.*kN
    series_file = "path.txt";
    n+=1;
}

define load factor increment 1/Nsteps;
define solver UMFPack;
define convergence test Norm_Displacement_Increment
tolerance = 1e-5
maximum_iterations = 100
verbose_level = 4;
define algorithm Newton ;

define NDMaterialLT constitutive integration algorithm Euler_One_Step
    yield_function_relative_tolerance = 0.0002
    stress_relative_tolerance = 0.002
    maximum_iterations = 1000;
simulate Ncyc*Nsteps steps using static algorithm;
bye;

The ESSI model fei/DSL files for this example can be downloaded here.
707.19 Contact Element Under Static Loading

Two Bar Normal Contact Problem Under Monotonic Loading.

This is an example of normal monotonic loading on a 1-D contact/interface between two bars separated by an initial gap of 0.1 unit. An illustrative diagram of the problem statement is shown below.

Figure 707.53: Illustration of Two Bar Normal Contact Problem under monotonic loading with initial gap

ESSI model fei/DSL file:

```
model name "Two_Bar_Contact_Under_Normal_Monotonic_Loading" ;

// Adding material
add material #1 type uniaxial_elastic elastic_modulus = 1*Pa ← viscoelastic_modulus = 0*Pa*s;

// Adding Nodes
add node #1 at (0*m,0*m,0*m) with 3 dofs;
add node #2 at (1*m,0*m,0*m) with 3 dofs;
add node #3 at (1.1*m,0*m,0*m) with 3 dofs;
add node #4 at (2.1*m,0*m,0*m) with 3 dofs;

// Adding Fixities
fix node #1 dofs ux uy uz;
fix node #4 dofs ux uy uz;
fix node #2 dofs uy uz ;
fix node #3 dofs uy uz ;

// Adding Truss Elements
add element #1 type truss with nodes (1,2) use material # 1 cross_section = ← 1*m^2 mass_density = 1*kg/m^3;
add element #2 type truss with nodes (3,4) use material # 1 cross_section = ← 1*m^2 mass_density = 1*kg/m^3;

// Adding Contact Element
add element #3 type FrictionalPenaltyContact with nodes (2,3) normal_stiffness = 1e10*N/m
tangential_stiffness = 1e10*Pa*m normal_damping = 0*kN/m*s tangential_damping = 0*kN/m*s friction_ratio = 0.3
```
contact_plane_vector = (1,0,0);

new loading stage "Adding_Normal_Load";

add load #1 to node #2 type linear Fx = 0.3*N;

Nsteps = 10;

tol = 5e-12;
define convergence test Norm_Displacement_Increment
tolerance = tol
maximum_iterations = 10
verbose_level = 4;

define algorithm Newton;
define solver UMFPack;

define load factor increment 1/Nsteps;
simulate Nsteps steps using static algorithm;

bye;

The displacement output of Node 2 and Node 3 are shown below.

![Displacement Graph](image)

Figure 707.54: Displacement of Nodes 2 and 3

The ESSI model fei/DSL files for this example can be downloaded here.
707.20 Four Bar Contact Problem With Normal and Shear Force Under Monotonic Loading

This is an example to show the normal and tangential behaviour (stick and slip case) of contacts/interfaces using four bars in 2-D plane. The bars in x-directions are in contact (initial gap=0).

Figure 707.55: Illustration of Four Bar Normal Contact Problem With Normal and Shear Force Under Monotonic Loading with no initial gap

ESSI model fei/DSL file:
model name "Four_Bar_Contact_Under_Monotonic_Normal_and_Shear_Loading";

// Adding material
add material #1 type uniaxial_elastic elastic_modulus = 1*Pa
  viscoelastic_modulus = 0*Pa*s;

// Adding Nodes
add node #1 at (0*m,0*m,0*m) with 3 dofs;
add node #2 at (1*m,0*m,0*m) with 3 dofs;
add node #3 at (1*m,0*m,0*m) with 3 dofs;
add node #4 at (2*m,0*m,0*m) with 3 dofs;
add node #5 at (1*m,-1*m,0*m) with 3 dofs;
add node #6 at (1*m,1*m,0*m) with 3 dofs;

// Adding Truss Elements
add element #1 type truss with nodes (1,2) use material # 1 cross_section =
  1*m^2 mass_density = 1*kg/m^3;
add element #2 type truss with nodes (3,4) use material # 1 cross_section =
  1*m^2 mass_density = 1*kg/m^3;
add element #3 type truss with nodes (3,5) use material # 1 cross_section =
  1*m^2 mass_density = 1*kg/m^3;
add element #4 type truss with nodes (2,6) use material # 1 cross_section =
  1*m^2 mass_density = 1*kg/m^3;

// Adding Contact Element
add element #5 type FrictionalPenaltyContact with nodes (2,3)
  normal_stiffness = 1e12*N/m
  tangential_stiffness = 1e12*N/m
  normal_damping = 0*N/m*s
  tangential_damping = 0*N/m*s
  friction_ratio = 0.4
  contact_plane_vector = (1,0,0);

// Adding Fixities
fix node #1 dofs ux uy uz ;
fix node #4 dofs ux uy uz ;
fix node #5 dofs ux uy uz ;
fix node #6 dofs ux uy uz ;
fix node #2 dofs uz ;
fix node #3 dofs uz ;

new loading stage "Normal_Loading";

add load #1 to node #2 type linear Fx = 0.1*N;

tol = 1e-10;
define convergence test Norm_Displacement_Increment
tolerance = tol
maximum_iterations = 10
verbose_level = 4;
The displacement output of *Node 2 and Node 3* are shown below.
The ESSI model fei/DSL files for this example can be downloaded [here](#).
Figure 707.56: Displacement of Nodes 2 and 3 along y direction
707.21 3-D Truss example with normal confinement and Shear Loading

A simple 3-D truss example with Normal confinement in z-direction of $F_N = 0.5N$, friction coefficient $\mu = 0.2$ and shear loading of magnitude $F_s = 0.5N$. Figure 707.57 below, shows the description of the problem.

![Figure 707.57: Illustration of 3-D Truss Problem with confinement loading in z-direction of 0.5N and then shear loading of 0.5N in x-y plane](image)

ESSI model fei/DSL file:

```plaintext
model name "3-D_Contact_Under_Normal_And_Tangential_Loading" ;

// Adding material
add material #1 type uniaxial_elastic elastic_modulus = 1*Pa ←
    viscoelastic_modulus = 0*Pa*s;

// Adding Nodes
add node #1 at (0*m,0*m,0*m) with 3 dofs;
add node #2 at (0*m,0*m,0*m) with 3 dofs;
add node #3 at (-1*m,0*m,0*m) with 3 dofs;
add node #4 at (0*m,1*m,0*m) with 3 dofs;
add node #5 at (0*m,0*m,1*m) with 3 dofs;

// Adding Fixities
fix node #1 dofs ux uy uz;
fix node #3 dofs ux uy uz;
fix node #4 dofs ux uy uz;
fix node #5 dofs ux uy uz;

// Adding Truss Elements
add element #1 type truss with nodes (2,3) use material # 1 cross_section = ←
    1*m^2 mass_density = 1*kg/m^3;
add element #2 type truss with nodes (2,4) use material # 1 cross_section = ←
```

1*m^2 mass_density = 1*kg/m^3;
add element #3 type truss with nodes (2,5) use material # 1 cross_section = ←
1*m^2 mass_density = 1*kg/m^3;

// Adding Contact Element
add element #4 type FrictionalPenaltyContact with nodes (1,2)
normal_stiffness =1e10*N/m
tangential_stiffness = 1e10*Pa*m
normal_damping = 0*kN/m*s
tangential_damping = 0*kN/m*s
friction_ratio = 0.2
contact_plane_vector = (0,0,1);

new loading stage "Adding_Normal_Load";
add load #1 to node #2 type linear Fz = -0.5*N;
Nsteps = 1;
tol = 1e-10;
define convergence test Norm_Displacement_Increment
tolerance = tol
maximum_iterations = 1
verbose_level = 4;
define algorithm Newton;
define solver UMFPack;
define load factor increment 1/Nsteps;
simulate Nsteps steps using static algorithm;

new loading stage "Shear_Loading";
add load #2 to node #2 type linear Fx = 0.4;
add load #3 to node #2 type linear Fy = 0.3;
tol = 1e-12;
define convergence test Norm_Displacement_Increment
tolerance = tol
maximum_iterations = 10
verbose_level = 4;
define algorithm Newton;
Nsteps= 20;
define solver UMFPack;
define load factor increment 1/Nsteps;
simulate Nsteps steps using static algorithm;
bye;
The generalized displacement response of the tangential loading stage is shown below.

![Graph showing displacement response](image1)

**Figure 707.58:** Displacements of Node 2 with applied shear tangential load step.

![Graph showing force response](image2)

**Figure 707.59:** Resisting force by the contact/interface element with applied shear tangential load step.
707.22 Six Solid Blocks Example With Contact

This is a 3-D solid block example with initial normal and then tangential load on different surfaces as shown below.

Figure 707.60: Illustration of Six Solid Blocks Example with Contact having first normal and then tangential loading stages.

ESSI model fei/DSL file:

```plaintext
model name "Six_Solid_Blocks_Example_With_Contact";

// Adding material
add material #1 type linear_elastic_isotropic_3d_LT mass_density=2000*kg/m^3 ←
elastic_modulus=200*MPa poisson_ratio=0.3;

// Adding Nodes
add node # 1 at (-1.500000*m,-0.500000*m,0.000000*m) with 3 dofs;
add node # 2 at (-1.500000*m,0.500000*m,0.000000*m) with 3 dofs;
add node # 3 at (1.500000*m,-0.500000*m,0.000000*m) with 3 dofs;
add node # 4 at (1.500000*m,0.500000*m,0.000000*m) with 3 dofs;
add node # 5 at (-1.500000*m,-0.500000*m,-2.000000*m) with 3 dofs;
add node # 6 at (-1.500000*m,0.500000*m,-2.000000*m) with 3 dofs;
add node # 7 at (0.500000*m,-0.500000*m,0.000000*m) with 3 dofs;
add node # 8 at (0.500000*m,0.500000*m,0.000000*m) with 3 dofs;
add node # 9 at (0.500000*m,-0.500000*m,0.000000*m) with 3 dofs;
add node # 10 at (0.500000*m,0.500000*m,0.000000*m) with 3 dofs;
add node # 11 at (-0.500000*m,0.500000*m,-2.000000*m) with 3 dofs;
add node # 12 at (0.500000*m,0.500000*m,-2.000000*m) with 3 dofs;
add node # 13 at (-0.500000*m,0.500000*m,-2.000000*m) with 3 dofs;
```
add node # 14 at (0.500000*m,0.500000*m,-2.000000*m) with 3 dofs;
add node # 15 at (0.500000*m,-0.500000*m,-2.000000*m) with 3 dofs;
add node # 16 at (-0.500000*m,-0.500000*m,-2.000000*m) with 3 dofs;
add node # 17 at (-1.500000*m,-0.500000*m,-1.000000*m) with 3 dofs;
add node # 18 at (-1.500000*m,0.500000*m,-1.000000*m) with 3 dofs;
add node # 19 at (1.500000*m,0.500000*m,-1.000000*m) with 3 dofs;
add node # 20 at (1.500000*m,-0.500000*m,-1.000000*m) with 3 dofs;
add node # 21 at (-0.500000*m,0.500000*m,-1.000000*m) with 3 dofs;
add node # 22 at (0.500000*m,0.500000*m,-1.000000*m) with 3 dofs;
add node # 23 at (-0.500000*m,-0.500000*m,-1.000000*m) with 3 dofs;
add node # 24 at (0.500000*m,-0.500000*m,-1.000000*m) with 3 dofs;
add node # 25 at (-0.500000*m,-0.500000*m,0.000000*m) with 3 dofs;
add node # 26 at (0.500000*m,-0.500000*m,0.000000*m) with 3 dofs;
add node # 27 at (-0.500000*m,0.500000*m,0.000000*m) with 3 dofs;
add node # 28 at (0.500000*m,0.500000*m,0.000000*m) with 3 dofs;
add node # 29 at (-0.500000*m,0.500000*m,-1.000000*m) with 3 dofs;
add node # 30 at (0.500000*m,0.500000*m,-1.000000*m) with 3 dofs;
add node # 31 at (-0.500000*m,-0.500000*m,-1.000000*m) with 3 dofs;
add node # 32 at (0.500000*m,-0.500000*m,-1.000000*m) with 3 dofs;

// Adding Solid 8 Node Brick Elements
add element #1 type 8NodeBrickLT with nodes (21,23,17,18,11,9,1,2) use ← material #1;
add element #2 type 8NodeBrickLT with nodes (13,16,5,6,21,23,17,18) use ← material #1;
add element #3 type 8NodeBrickLT with nodes (30,32,31,29,28,26,25,27) use ← material #1;
add element #4 type 8NodeBrickLT with nodes (14,15,16,13,22,24,23,21) use ← material #1;
add element #5 type 8NodeBrickLT with nodes (19,20,24,22,4,3,10,12) use ← material #1;
add element #6 type 8NodeBrickLT with nodes (7,8,15,14,19,20,24,22) use ← material #1;

//Adding some variables
Kn = 1e12*N/m; // normal penalty stiffness
Kt = 1e12*N/m; // tangential penalty stiffness
Cn = 0*N/m*s; // normal penalty damping
Ct = 0*N/m*s; // tangential penalty damping
nu = 0.4; // friction ratio

// Adding Contact Element
add element #7 type FrictionalPenaltyContact with nodes (9,25)
normal_stiffness = Kn
tangential_stiffness = Kt
normal_damping = Cn
tangential_damping = Ct
friction_ratio = nu
contact_plane_vector = (1,0,0);
add element #8 type FrictionalPenaltyContact with nodes (10,26)
  normal_stiffness = Kn
  tangential_stiffness = Kt
  normal_damping = Cn
  tangential_damping = Ct
  friction_ratio = nu
  contact_plane_vector = (-1,0,0);

add element #9 type FrictionalPenaltyContact with nodes (11,27)
  normal_stiffness = Kn
  tangential_stiffness = Kt
  normal_damping = Cn
  tangential_damping = Ct
  friction_ratio = nu
  contact_plane_vector = (1,0,0);

add element #10 type FrictionalPenaltyContact with nodes (12,28)
  normal_stiffness = Kn
  tangential_stiffness = Kt
  normal_damping = Cn
  tangential_damping = Ct
  friction_ratio = nu
  contact_plane_vector = (-1,0,0);

add element #11 type FrictionalPenaltyContact with nodes (21,29)
  normal_stiffness = Kn
  tangential_stiffness = Kt
  normal_damping = Cn
  tangential_damping = Ct
  friction_ratio = nu
  contact_plane_vector = (1,0,0);

add element #12 type FrictionalPenaltyContact with nodes (22,30)
  normal_stiffness = Kn
  tangential_stiffness = Kt
  normal_damping = Cn
  tangential_damping = Ct
  friction_ratio = nu
  contact_plane_vector = (-1,0,0);

add element #13 type FrictionalPenaltyContact with nodes (23,31)
  normal_stiffness = Kn
  tangential_stiffness = Kt
  normal_damping = Cn
  tangential_damping = Ct
  friction_ratio = nu
  contact_plane_vector = (1,0,0);

add element #14 type FrictionalPenaltyContact with nodes (24,32)
  normal_stiffness = Kn
tangential_stiffness = K_t
normal_damping = C_n
tangential_damping = C_t
friction_ratio = nu
contact_plane_vector = (-1,0,0);

add element #15 type FrictionalPenaltyContact with nodes (21,29)
  normal_stiffness = K_n
tangential_stiffness = K_t
normal_damping = C_n
tangential_damping = C_t
friction_ratio = nu
contact_plane_vector = (0,0,1);

add element #16 type FrictionalPenaltyContact with nodes (22,30)
  normal_stiffness = K_n
tangential_stiffness = K_t
normal_damping = C_n
tangential_damping = C_t
friction_ratio = nu
contact_plane_vector = (0,0,1);

add element #17 type FrictionalPenaltyContact with nodes (23,31)
  normal_stiffness = K_n
tangential_stiffness = K_t
normal_damping = C_n
tangential_damping = C_t
friction_ratio = nu
contact_plane_vector = (0,0,1);

add element #18 type FrictionalPenaltyContact with nodes (24,32)
  normal_stiffness = K_n
tangential_stiffness = K_t
normal_damping = C_n
tangential_damping = C_t
friction_ratio = nu
contact_plane_vector = (0,0,1);

// Adding Fixities
fix node #5 dofs ux uy uz;
fix node #6 dofs ux uy uz;
fix node #13 dofs ux uy uz;
fix node #16 dofs ux uy uz;
fix node #15 dofs ux uy uz;
fix node #14 dofs ux uy uz;
fix node #7 dofs ux uy uz;
fix node #8 dofs ux uy uz;
fix node #17 dofs ux uy;
fix node #18 dofs ux uy;
fix node #1 dofs ux uy;
fix node #2 dofs ux uy;
168 | fix node #20 dofs ux uy;
169 | fix node #19 dofs ux uy;
170 | fix node #3 dofs ux uy;
171 | fix node #4 dofs ux uy;
172 | fix node #9 dofs uy;
173 | fix node #10 dofs uy;
174 | fix node #23 dofs uy;
175 | fix node #24 dofs uy;
176 | fix node #11 dofs uy;
177 | fix node #21 dofs uy;
178 | fix node #12 dofs uy;
179 | fix node #22 dofs uy;
180 | fix node #25 dofs uy;
181 | fix node #26 dofs uy;
182 | fix node #27 dofs uy;
183 | fix node #28 dofs uy;
184 | fix node #29 dofs uy;
185 | fix node #30 dofs uy;
186 | fix node #31 dofs uy;
187 | fix node #32 dofs uy;
188 |
189 | new loading stage "Normal_Loading";
190 |
191 | add load #1 to element #3 type surface at nodes (25,26,27,28) with magnitude $(-1*Pa)$;
192 |
193 | tol = 1e-12;
194 | define convergence test Norm_Displacement_Increment
195 | tolerance = tol
196 | maximum_iterations = 100
197 | verbose_level = 4;
198 |
199 | define algorithm Newton;
200 |
201 | Nsteps= 10;
202 | define solver UMFPack;
203 | define load factor increment 1/Nsteps;
204 | simulate Nsteps steps using static algorithm;
205 |
206 | new loading stage "Shear_Loading";
207 |
208 | add load #2 to element #3 type surface at nodes (26,28,30,32) with magnitude $(-1*Pa)$;
209 |
210 | tol = 1e-12;
211 | define convergence test Norm_Displacement_Increment
212 | tolerance = tol
213 | maximum_iterations = 100
214 | verbose_level = 4;
215 |
216 | define algorithm Newton;
Nsteps = 10;
define solver UMFPack;
define load factor increment 1/Nsteps;
simulate Nsteps steps using static algorithm;

bye;

The generalized displacement field of the two loading stages **normal loading** and **tangential loading** is shown below.

![Generalized displacement magnitude visualization of normal loading](image1)

**Figure 707.61**: Generalized displacement magnitude visualization of normal loading

![Generalized displacement magnitude visualization of tangential loading](image2)

**Figure 707.62**: Generalized displacement magnitude visualization of tangential loading
The ESSI model fei/DSL files for this example can be downloaded here.
707.23 Pure shear model for G/Gmax plot

Problem description:

The pure shear model for G/Gmax plot

![Figure 707.63: The pure shear model for (a) confinement and (b) shearing](image)

ESSI model fei/DSL file:

```plaintext
model name "Gmax" ;
// Parameters:
phi = 0.0135713590083;
ha = 2.94767923453;
cr = 1854.31984573;
rho=1922.5;
depth=0.1524/2;
confinstress=9.8*depth*rho;
G=12388.33;
p0 = confinstress*Pa;
phirad = pi*phi/180;
M = 6*sin(phirad)/(3-sin(phirad));
nu=0.3;
add material # 1 type DruckerPragerArmstrongFrederickLT
  mass_density = rho*kg/m^3
  elastic_modulus = 2*G*(1+nu)*Pa
  poisson_ratio = nu
  druckerprager_k = M
  armstrong_frederick_ha = ha*Pa
  armstrong_frederick_cr = cr*Pa
```

Jeremić et al., Real-ESSI Lecture Notes
UCD and LBNL
version: 2. March, 2020, 6:58
isotropic_hardening_rate = 0*Pa
initial_confining_stress = 10*Pa;

add node # 1 at (1.0000 *m, 0.0000 *m, 0.0000 *m) with 3 dofs;
add node # 2 at (0.0000 *m, 1.0000 *m, 0.0000 *m) with 3 dofs;
add node # 3 at (1.0000 *m, 2.0000 *m, 0.0000 *m) with 3 dofs;
add node # 4 at (2.0000 *m, 1.0000 *m, 0.0000 *m) with 3 dofs;
add node # 5 at (1.0000 *m, 0.0000 *m, 1.0000 *m) with 3 dofs;
add node # 6 at (0.0000 *m, 1.0000 *m, 1.0000 *m) with 3 dofs;
add node # 7 at (1.0000 *m, 2.0000 *m, 1.0000 *m) with 3 dofs;
add node # 8 at (2.0000 *m, 1.0000 *m, 1.0000 *m) with 3 dofs;
add element # 1 type 8NodeBrickLT with nodes(1,2,3,4,5,6,7,8) use material # 1;

// fix the y direction for node 2,4,6,8
fix node # 2 dofs uy;
fix node # 4 dofs uy;
fix node # 6 dofs uy;
fix node # 8 dofs uy;

// fix the x direction for node 1,3,5,7
fix node # 1 dofs ux;
fix node # 3 dofs ux;
fix node # 5 dofs ux;
fix node # 7 dofs ux;

// Stage 1: confinement
new loading stage "confinement";
add load # 1 to node # 1 type linear Fy= p0*m^2;
add load # 2 to node # 3 type linear Fy= -p0*m^2;
add load # 3 to node # 5 type linear Fy= p0*m^2;
add load # 4 to node # 7 type linear Fy= -p0*m^2;
add load # 5 to node # 2 type linearFx= p0*m^2;
add load # 6 to node # 4 type linearFx= -p0*m^2;
add load # 7 to node # 6 type linearFx= p0*m^2;
add load # 8 to node # 8 type linearFx= -p0*m^2;

// confinement at z direction
add load # 101 to node # 1 type linear Fz= p0*m^2;
add load # 102 to node # 2 type linear Fz= p0*m^2;
add load # 103 to node # 3 type linear Fz= p0*m^2;
add load # 104 to node # 4 type linear Fz= p0*m^2;
add load # 105 to node # 5 type linear Fz= -p0*m^2;
add load # 106 to node # 6 type linear Fz= -p0*m^2;
add load # 107 to node # 7 type linear Fz= -p0*m^2;
add load # 108 to node # 8 type linear Fz= -p0*m^2;

// add algorithm and solver
Nsteps=100;
define load factor increment 1/Nsteps;
define solver ProfileSPD;
define convergence test Norm_Displacement_Increment
tolerance = 1e-5
maximum_iterations = 100
verbose_level = 4;
// define algorithm With_no_convergence_check;
define algorithm Newton;
define NDMaterialLT constitutive integration algorithm Euler_One_Step
yield_function_relative_tolerance = 0.00002
stress_relative_tolerance = 0.0002
maximum_iterations = 1000;
simulate Nsteps steps using static algorithm;
// ------------------------------
// Stage 2: shear
new loading stage "shear";
// fix all the uz, since we want plane strain.
i=1;
while (i<9) {
    remove load # 100+i;
    fix node # i dofs uz;
    i=i+1;
};
shearforce=1.6*kN;
add load # 9 to node # 1 type linear Fy= shearforce;// series_file = "path.txt" ; 
add load # 10 to node # 3 type linear Fy=-shearforce;// series_file = "path.txt" ;
add load # 11 to node # 5 type linear Fy= shearforce;// series_file = "path.txt" ;
add load # 12 to node # 7 type linear Fy=-shearforce;// series_file = "path.txt" ;
add load # 13 to node # 2 type linear Fx=-shearforce;// series_file = "path.txt" ;
add load # 14 to node # 4 type linear Fx= shearforce;// series_file = "path.txt" ;
add load # 15 to node # 6 type linear Fx=-shearforce;// series_file = "path.txt" ;
add load # 16 to node # 8 type linear Fx= shearforce;// series_file = "path.txt" ;
// add algorithm and solver
Nsteps=1e4;
declare static integrator displacement_control using node # 1 dof uy increment →
1e-2/Nsteps*m;
declare convergence test Norm_Displacement_Increment tolerance = 0.000001 →
maximum_iterations = 100 verbose_level = 0;
declare solver ProfileSPD;
declare algorithm Newton;
declare NDMaterialLT constitutive integration algorithm Euler_One_Step
yield_function_relative_tolerance = 0.00002
stress_relative_tolerance = 0.0002
maximum_iterations = 1000;
simulate Nsteps steps using static algorithm;
bye;

![Graph showing G/Gmax results vs strain](image)

Figure 707.64: The G/Gmax results

The ESSI model fei/DSL files for this example can be downloaded [here](link).
707.24 Multi-yield-surface von-Mises for G/Gmax plot

Problem description:

This model illustrates the G/Gmax input to multi-yield-surface von-Mises material. This example is based on one Gauss-point with multi-yield-surface von-Mises material. The G/Gmax is converted to material modeling parameters (yield-surface size and hardening parameter) inside the DSL.

ESSI model fei/DSL file:

```plaintext
model name "GGmax";
add material # 1 type vonMisesMultipleYieldSurfaceGoverGmax
  mass_density = 0.0*kg/m^3
  initial_shear_modulus = 3E8 * Pa
  poisson_ratio = 0.0
  total_number_of_shear_modulus = 9
  GoverGmax = "1,0.995,0.966,0.873,0.787,0.320,0.109,0.063"
  ShearStrainGamma = "0,1E-6,1E-5,5E-5,1E-4, 0.0005, 0.001, 0.005, 0.01"
;
incr_size = 0.000001 ;
max_strain= 0.005 ;
num_of_increm = max_strain/incr_size -1 ;
simulate constitutive testing strain control pure shear use material # 1
  confinement_strain = 0.0
  strain_increment_size = incr_size
  maximum_strain = max_strain
  number_of_increment = num_of_increm;
bye;
```

Computed G/Gmax curve exactly matches the one used for input at control points. The difference in G/Gmax between control points can be reduced by using more than just 9 control points as in this example.
Material Behavior: Stress-Strain

Figure 707.65: Stress-Strain Relationship

Multi-Yield-Surface vonMises G/Gmax

Figure 707.66: The G/Gmax results.
Figure 707.67: Damping Ratio Plot
707.25 Multi-yield-surface Drucker-Prager for G/Gmax plot

Problem description:

This model illustrates the G/Gmax input to multi-yield-surface Drucker-Prager material. Purely deviatoric plastic flow is used in this material, which means that the parameter dilation_scale is set to zero. If user wants to model change of volume (dilation or compression) for this material, then G/Gmax curve need to be iterated upon manually by changing yield surface size directly, which is done using different DruckerPragerMultipleYieldSurface command. This example is based on one Gauss-point which use multi-yield-surface Drucker-Prager material. The G/Gmax is converted to the yield-surface size and hardening parameter inside the DSL.

ESSI model fei/DSL file:

```plaintext
model name "GGmax";
add material # 1 type DruckerPragerMultipleYieldSurfaceGoverGmax
mass_density = 0.0*kg/m^3
initial_shear_modulus = 3E8 * Pa
poisson_ratio = 0.0
initial_confining_stress = 1E5 * Pa
reference_pressure = 1E5 * Pa
pressure_exponential_n = 0.5
cohesion = 0. * Pa
dilation_angle_eta = 1.0
dilation_scale = 0.0
total_number_of_shear_modulus = 9
GoverGmax = "1,0.995,0.966,0.873,0.787,0.620,0.109,0.063"
ShearStrainGamma = "0,1E-6,1E-5,5E-5,1E-4, 0.0005, 0.001, 0.005, 0.01"
;
incr_size = 0.000001;
max_strain= 0.005;
um_of_increm = max_strain/incr_size -1;
simulate constitutive testing strain control pure shear use material # 1
confinement_strain = 0.0
strain_increment_size = incr_size
maximum_strain = max_strain
number_of_increment = num_of_increm;
bye;
```

Inside the DSL, the yield surface radius is calculated as $\sqrt{3}\sigma_y$, where $\sigma_y$ is the yield stress of the corresponding yield surface. Then, the radius is divided by the confinement to obtain the slope (opening angle).
The hardening parameter is calculated as

\[ \frac{1}{H'_i} = \frac{1}{H_i} - \frac{1}{2G} \]  

(707.1)

where \( H'_i \) is the current hardening parameter corresponding to yield surface \( i \). \( H_i \) is the current tangent shear modulus to surface \( i \), namely, \( H_i = 2\left(\frac{\tau_i + 1 - \gamma_i}{\gamma_i + 1 - \gamma_i}\right) \). And \( G \) is the initial shear modulus.
Figure 707.69: Nested-Yield-Surface Drucker-Prager $G/G_{\text{max}}$ results

Figure 707.70: Damping Ratio Plot
Appendix 708

Work Organization (1989)
This section describes in some detail work organization related to the development of FEI modeling and computational system.

### 708.1 Communication

Tablets for skype calls

### 708.2 Writing (Notes, Code, &c.) Version Control

#### 708.2.1 Source Code

**Memory Leaks** Memory leaks are best discovered by running Valgrind (http://valgrind.org/).

There are a number of tools that can be used with Valgrind. Mentioned are some of the most important ones, with example commands\(^1\)

use of tcsh is assumed, with a time stamp (used in commands below) set as: `set TIMESTAMP \=
= `date +%h_%d_%Y_%Hh_%Mm_%Ss_%A``

- `valgrind -v --leak-check=yes --show-reachable=yes --num-callers=32 --trace-malloc=yes --error-limit=no --tool=massif $argv[1]`

\(^1\)Examples use syntax from few years ago, so should be proper syntax should be verified using excellent Valgrind documentation.
708.2.2 Lecture Notes

708.2.3 Bibliography

Bibliography List.

Papers of interest are organized in bibtex files (managed through git version control. A list of those paper is compiled and available at:

http://sokocalo.engr.ucdavis.edu/~jeremic/research/bibmech.pdf
http://sokocalo.engr.ucdavis.edu/~jeremic/research/bibcomp.pdf
http://sokocalo.engr.ucdavis.edu/~jeremic/research/bibeduc.pdf

Bibliography Repository.

Most listed papers are available at:

http://sokocalo.engr.ucdavis.edu/~jeremic/PAPERSlocalREPO/.

708.3 Backup

708.4 Calendar

708.5 Useful Programs and Scripts

708.5.1 Backup Scripts

708.5.2 Domain Reduction Method Processing Programs and Scripts

DRM Node Extraction for fk.

fk Output Processing for DRM.
708.5.3 Pre Processing Programs and Scripts

708.5.4 Post Processing Programs and Scripts

708.5.5 Parallel Computer Architecture

http://www.open-mpi.org/projects/hwloc/
Appendix 709

Collected Bibliography

Compilation of all collected bibliography, over years, not necessarily cited in this book.
Bibliography

by:

Jeremić CompMech Group
Department of Civil and Environmental Engineering
University of California, Davis
Bibliography


[Ala ] Fadel Alameddine. Private communications. ..., 2003 –.


[A.TXX] L. A. Taber. Application of shell theory to cardiac mechanics. XX, XX.


Kenneth W Campbell and Yousef Bozorgnia. NGA ground motion model for the geometric mean horizontal component of PGA, PGV, PGD and 5% damped linear elastic response spectra for periods ranging from 0.01 to 10 s. *Earthquake Spectra*, 24(1):139–171, 2008.


[FD05] Gregory Fenves and Mathew Dryden. Nees sfsi demonstration project. NEES project meeting, TX, Austin, August 2005.


---

*Jeremić et al.*

*UCD and LBNL*

*version: 2. March, 2020, 6:58*


Qing He, Houle Gan, and Dan Jiao. An explicit time-domain finite-element method that is unconditionally stable. Purdue e-Pubs; ECE Technical Reports 421, Purdue University, 2011.


E. Hinton, T. K. Hellen, and L. P. R. Lyons. On elasto-plastic benchmark philosophies. ... pages 389–407, ...


[Hin11] Klaus-G. Hinzen. Rotation of vertically oriented objects during earthquakes. published via email to rotation@lists.geophysik.uni-muenchen.de group, July 2011.


[HKB17] Xu Huang, Oh-Sung Kwon, Evan Bentz, and Julia Tchermer. Method for evaluation of concrete containment structure subjected to earthquake excitation and internal pressure increase. Earthquake Engineering & Structural Dynamics, pages n/a–n/a, 2017. EQE-17-0330.R3.


Bor-Shouh Huang. Ground rotational motions of the 1999 Chi-Chi, Taiwan earthquake as inferred from dense array observations. Geophysical Research Letters, 30(6), 2003.


[Jer89] Boris Jeremić. Dynamic analysis of axisymmetric solids subjected to non-symmetric loading by the finite element method. Diploma thesis., Faculty of Civil Engineering, Belgrade University, July 1989. in Serbian. (Борис Jeremić Динамика Анализа Ротационо Симетричних Тела Оптерећених Несиметричним Оптерећенима Методом Конечних Елемента, Дипломски Рад Грађевинскис Факултет Универзитета у Београду.


[JS96b] Boris Jeremić and Stein Sture. Refined solution procedures for finite element analysis in geotechnics, presentation at the CAMM seminar 96/2, Center for Acoustics, Mechanics and Materials, University of Colorado, October 1996.


[KMMK0] H. Kato, T. Mori, N. Murota, and M. Kikuchi. Analytical model for elastoplastic and creep–like behavior of high-damping rubber bearings. ASCE Journal of Structural Engineering, 0(0):04014213, 0.


Jeremić et al., Real-ESSI


of irregular polygonal particles. International Journal of Solids and Structures, 42(24-25):6356–6375,
December 2005.

[NM00] A. Naess and V. Moe. Efficient path integration methods for nonlinear dynamic systems. Probabilistic


1985.

229, 1990.

[NNO95] Sia Nemat-Nasser and Naoyuki Okada. Direct observation of deformation of granular materials through
Mechanics Division of the American Society of Civil Engineers, May 1995.


[Nog16] Silvana Montoya Noguera. Assessment and mitigation of liquefaction seismic risk : numerical modeling

[Noh06] Hyuk Chun Noh. Effect of multiple uncertain material properties on the response variability of in-plane
April 2006.


[Nov87] Milos Novak. Discussion of “dynamic response of arbitrarily shaped foundations: Experimental verifica-

chini, editors, Constitutive Equations for Granular Non–Cohesive Soils, pages 501–519. A. A. Balkema,


[NPF15] Yuri P. Nazarov, Elena Poznyak, and Anton V. Filimonov. A brief theory and computing of seismic
ground rotations for structural analyses. Soil Dynamics and Earthquake Engineering, 71(0):31 – 41,
2015.


403–414. __, 19__.

[NRHO14] Ehsan Nikbakht, Khalim Rashid, Farzad Hejazi, and Siti A Osman. A numerical study on seismic response
of self-centring precast segmental columns at different post-tensioning forces. Latin American Journal

Valanis, editors, Damage Mechanics and Localization, volume AMD-142, MD-34, pages 109 – 123,
Engineers.


NVIDIA Corporation. CUDA zone. GPGPU compiler, 2007.


Jeremić et al., Real-ESSI


[Ric11] Lewis Fry Richardson. The approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 210:307–357, 1911.


[SCW+08] MD Symans, FA Charney, AS Whittaker, MC Constantinou, CA Kircher, MW Johnson, and RJ McNa- 


[Swa98] Travis Swatson. 3D data visualization and modeling with a haptic interface. UCDAvis seminar slides, May 1998.


