OpenSees

UCD Computational Geomechanics:

Command Manual

by:
Boris Jeremić
with contributions by:
Zhao Cheng, Zhaohui Yang, Jinxiu Liao, Xiaoyan Wu, Guanzhou Jie, Kallol Sett, Vlado Vukadin, Matthias Preisig

Department of Civil and Environmental Engineering
University of California, Davis

Version: 5. September, 2005, 12:15

Contents

List of Figures

Chapter 1

OpenSees Commands

1.1 OpenSees Command to Create Elastic Isotropic/Anisotropic 3D Material

There are two types of 3D elastic isotropic material models (i.e. linear elastic and nonlinear or pressure sensitive elastic materials) and one cross anisotropic elastic material model that you can create.

1.1.1 ElasticIsotropic3D command

```
nDMaterial ElasticIsotropic3D matTag? E0? nu? rho?
```

The `ElasticIsotropic3D` material is the standard linear elastic isotropic three dimensional material implemented based on tensor operation. The arguments to construct the material are its tag, `matTag`, Young's Modulus at atmospheric pressure `E0`, Poisson's ratio `nu`, and mass density `rho`.

1.1.2 PressureDependentElastic3D command

```
nDMaterial PressureDependentElastic3D matTag? E0? nu? rho? n? pr? pc?
```

The `PressureDependentElastic3D` material is the standard nonlinear elastic isotropic three dimensional material implemented based on tensor operation. The first four arguments are the same as linear elastic command described above. The pressure dependent elastic modulus is to be determined using the following formula 1.1 . There are three more arguments for this command. `n` is the exponent, `pr` (p_{ref}) is the atmospheric pressure, while `pc` ($p_{cut-off}$) is the cut-off pressure. When p' (the mean effective normal stress) is less than $p_{cut-off}$, then $p' = p_{cut-off}$.

$$E = E_o \left(\frac{p'}{p_{ref}} \right)^n \tag{1.1}$$

1.1.3 Elastic Cross Anisotropic 3D command

```
nDMaterial ElasticCrossAnisotropic matTag? Eh? Ev? nuhv? nuhh? Ghv?
```

The `ElasticCrossAnisotropic` material is the standard linear elastic cross anisotropic three dimensional material implemented based on tensor operation. The arguments to construct the material are its tag, `matTag`, the elastic modulus in the cross plane, `Eh`, the elastic modulus in the plane vertical to the cross plane, `Ev`, Poisson's ratio between the cross plane and its vertical plane, `nuhv`, Poisson's ratio between in the cross plane, `nuhh`, and the shear modulus between the cross plane and its vertical plane, `Ghv`.

1.2 OpenSees Commands to Create Material Models using Template Elasto–Plastic Framework

1.2.1 Yield Surface Command

```
set ys "-YieldSurfaceType <parameter list>"
```

This command sets the yield surface variable `ys` to be the specified type. A list of paramaters can be passed to define the yield surface and the number of parameters depend on the type of yield surface. Valid strings for YieldSurfaceType are `DP`, `VM`, `CC`, and `RMC01`, which are described in the following subsections.

Drucker-Prager Yield Surface

```
set ys "-DP"
```

`DP` stands for Drucker-Prager type, i.e. cone shaped yield surface. In this case, no parameter needs to be supplied since the slope α is treated as an internal variable.

von Mises Yield Surface

```
set ys "-VM"
```

`VM` stands for von Mises type, i.e. cylinder shaped yield surface. In this case, no parameter needs to be supplied since the size of the cylinder is treated as an internal variable.

Cam-Clay Yield Surface

```
set ys "-CC M?"
```

`CC` stands for Cam-Clay type, i.e. ellipsoid shaped yield surface. For `CC` type yield surface, the slope of the critical state line in p–q space, i.e. M, need to be supplied.

Rounded Mohr-Coulomb (Willam-Warnke) Yield Surface

```
set ys "-RMC01"
```

`RMC01` stands for rounded Mohr-Coulomb (Willam-Warnke) type, i.e. cone shaped yield surface. In this case, no parameter needs to be supplied, this is similar to the Drucker-Prager yield surface.

1.2.2 Potential Surface Command

```
set ps "-PotentialSurfaceType <parameter list>"
```

This command sets the potential surface variable `ps` to be the specified type. A list of paramaters can be passed to define the potential surface and the number of parameters depend on the type of potential surface. Valid strings for PotentialSurfaceType are `DP`, `VM`, and `CC`, which are described in the following subsections.

Drucker-Prager Potential Surface

```
set ps "-DP D?"
```

`DP` stands for Drucker-Prager type, i.e. cone shaped potential surface. In this case, the slope D that describes the dilation angle needs to be specified.

von Mises Potential Surface

```
set ps "-VM"
```

`VM` stands for von Mises type, i.e. cylinder shaped potential surface. In this case, no parameter needs to be supplied since the size of the cylinder is treated as an internal variable.

Cam-Clay Potential Surface

```
set ps "-CC M?"
```

`CC` stands for Cam-Clay type, i.e. ellipsoid shaped potential surface. For `CC` type potential surface, the slope of the critical state line in p–q space, i.e. M, need to be supplied.

Rounded Mohr-Coulomb (Willam-Warnke) Potential Surface

```
set ps "-RMC01"
```

`RMC01` stands for rounded Mohr-Coulomb (Willam-Warnke) type, i.e. cone shaped Potential surface. In this case, no parameter needs to be supplied, this is similar to the Drucker-Prager Potential surface.

1.2.3 Evolution Law Command

```
set el "-EvolutionLawType <parameter list>"
```

This command sets the evolution law variable `el` to be the specified type. A list of paramaters can be passed to define the potential surface and the number of parameters depend on the type of potential surface. Valid strings for EvolutionLawType are `Leq`, `NLp`, and , which are described in the following subsections.

Linear Scalar Evolution Law

```
set el "-Leq a?"
```

`Leq` stands for Linear Scalar Evolution Law. This hardening rule is based on the equivalent deviatoric plastic strain ϵ_q^{pl}. In this case, linear hardening coefficient `a` needs to be supplied. This hardening rule can be applied to any scalar internal variable, such as the slope of Drucker–Prager yield surface, the diameter of von Mises yield surface, and so on.

Nonlinear Scalar Evolution Law

```
set el "-NLp e0? lambda? kappa? "
```

`NLp` stands for Nonlinear Scalar Evolution Law. This hardening rule is based on the volumetic plastic strain ϵ_p^{pl}. In this case, parameters including void ration `e0`, `lambda` and `kappa` need to be supplied. This hardening rule is primarily for the evolution of the tip stress $p_o^{'}$ in Cam-Clay model.

Linear Tensorial Evolution Law

```
set et "-LEij a?"
```

`LEij` stands for Linear Tensorial Evolution Law. This hardening rule is based on the plastic strain ϵ_{ij}^{pl}. In this case, linear hardening coefficient `a` needs to be supplied. This hardening rule can be applied to any tensorial internal variable, such as the the center α_{ij} of Drucker–Prager yield surface or von Mises yield surface, and so on.

Nonlinear Tensorial Evolution Law (Armstrong-Frederick model)

```
set et "-NLEij ha? Cr?"
```

`NLEij` stands for Nonlinear Tensorial Evolution Law from Armstrong–Frederick nonlinear model. This kinematic hardening law is based on the plastic strain ϵ_{ij}^{pl}. In this case, nonlinear hardening coefficients `ha` and `Cr` need to be supplied. This hardening rule can be applied to any tensorial internal variable, such as the the center α_{ij} of Drucker–Prager yield surface or von Mises yield surface, and so on.

1.2.4 EPState Command

```
<set sts "Sxx? Sxy? Sxz? Syx? Syy? Syz? Szx? Szy? Szz?">
```

```
set eps "<-NOD nt?> -NOS ns? sc1? sc2? ... <-stressp sts>"
```

First statement sets the initial stress tensor to variable `sts` (if it is not stated here, no initial stress by default). Second statement assigns to the Elasto-Plastic state variable `eps` the specified state parameters, including number of tensorial internal variables `nt` (if it is not stated here, $nt = 0$ by default), number of scalar internal variables `ns` and corresponding initial values `sc1`, `sc2`, ..., and initial stresses defined in `$sts` (if it has been previously defined).

1.2.5 Template Elasto-Plastic Material Command

```
nDMaterial Template3Dep mTag? ElmTag? -YS $ys? -PS $ps? -EPS $eps? <-ELS1 $el?>
<$-ELT1 et?>
```

A template elasto-plastic material is constructed using `nDMaterial` command. The argument `mTag` is used to uniquely identify this nDMaterial object among nDMaterial objects in the BasicBuilder object. The

`ElmTag` is the material tag of the previously defined elastic material objects, such as `ElasticIsotropic3D`, `PressureDependentElastic3D` and `ElasticCrossAnisotropic3D`. The other parameters include previously defined yield surface object `ys`, potential surface object `ps`, elasto-plastic state object `eps`, scalar evolution law object `el`, and tensorial evolution law object `et`.

1.2.6 Examples

von Mises Model

```
# Elastic material
nDMaterial ElasticIsotropic3D 1 2.0e7 0.2 0.8e3

# Yield surface
set ys "-VM"

# Potential surface
set ps "-VM"

# Scalar evolution law: linear hardening coef = 1.0
set ES1  "-Leq  1.0"

# EPState
#______________k=f(Cu)
set EPS "-NOD 0 -NOS 1 20"

# Creating nDMaterial using Template Elastic-Plastic Model
# 2 is the material tag for the Template3Dep material
# 1 is the material tag of the previously defined elastic material
nDMaterial Template3Dep 2 1 -YS $ys -PS $ps -EPS $EPS -ELS1 $ES1
```

Drucker–Prager Model

```
# Elastic material
nDMaterial ElasticIsotropic3D 1 2.0e7 0.2 0.0

# Yield surface
set ys "-DP"

# Potential surface, must specify the dilation slope D as a input parameter
# D = 2 *sin(alpha) / (3^0.5) / (3-sin(alpha) ), alpha is the dilation angle
```

```
set ps "-DP 0.2"

# Scalar evolution law: linear hardening coef = 1.0
set ES1  "-Leq  1.0"

# Initial stress
set sts "0.01 0 0  0 0.01 0  0 0 0.01"

# EPState
#______________alpha___k
set EPS "-NOD 0 -NOS 2 0.2 0.0 -stressp $sts"
#
# where
# alpha = 2 *sin(phi) / (3^0.5) / (3-sin(phi) ), phi is the friction angle
# k = alpha*3*c/tan(phi), c is the cohesion

# Creating nDMaterial using Template Elastic-Plastic Model
# 2 is the material tag for the Template3Dep material
# 1 is the material tag of the previously defined elastic material
nDMaterial Template3Dep 2 1 -YS $ys -PS $ps -EPS $EPS -ELS1 $ES1
```

Cam-clay Model

```
# Elastic material
nDMaterial ElasticIsotropic3D 1 2.0e7 0.2 1.8e3

# Yield surface M = 1.2
set ys "-CC 1.2"

# Potential surface M = 1.2
set ps "-CC 1.2"

# Scalar evolution law___void ratio___Lamda___Kappa
set ES1  "-NLp           0.85        0.19   0.06"

# Initial stress
set sts "0.10 0 0  0 0.10 0  0 0 0.10"

#________________po
set EPS "-NOS 1 200.1 -stressp $sts"
```

```
# Creating nDMaterial using Template Elastic-Plastic Model
# 2 is the material tag for the Template3Dep material
# 1 is the material tag of the previously defined elastic material
nDMaterial Template3Dep 2 1 -YS $ys -PS $ps -EPS $EPS -ELS1 $ES1
```

1.3 OpenSees Commands to Create Brick Elements

1.3.1 The Eight Node Brick Element

```
element Brick8N  eletag? node1? node2? node3? node4? node5? node6? node7?
                 node8? matTag?  bf1? bf2? bf3? massDens?
```

The Brick8N element is the standard eight node three dimensional element implemented based on tensor operation. The arguments to construct the element are its tag, eletag, eight nodes ordered according to Figure 1.1, the material tag, matTag, the body forces, bf1, bf2, bf3, and the mass density, massDens. By default, $3 \times 3 \times 3$ integration points are used. Users will be able to specify number of integration points very soon.

The valid queries to a Brick8N element when creating an ElementRecorder are "force", "stiffness", "stress", "pq", "pqall", "gausspoint", "plastic" or "plasticGPC". For "stress" output, the six stress components from each Gauss point are output by the order: σ_x, σ_y, σ_z, τ_{xy}, τ_{xz}, τ_{yz}. The stresses can also be output in p and q format by using query "pq", where p is the hydrostatic pressure, while q is the equivalent deviatoric stress. In this case, the stress state at one gauss point is printed in the pq format. If the stress states at all gauss points need to be printed, use the query "pqall". For "gausspoint", the coordinates of all Gauss points are printed out. For "plastic", the equivalent deviatoric plastic strain from each Gauss point is output in the same order as the coordinates are printed. But the coordinates have to be output separated. If one needs to output the gauss point coordinates together with the plastic strain, the query "plasticGPC" needs to be used.

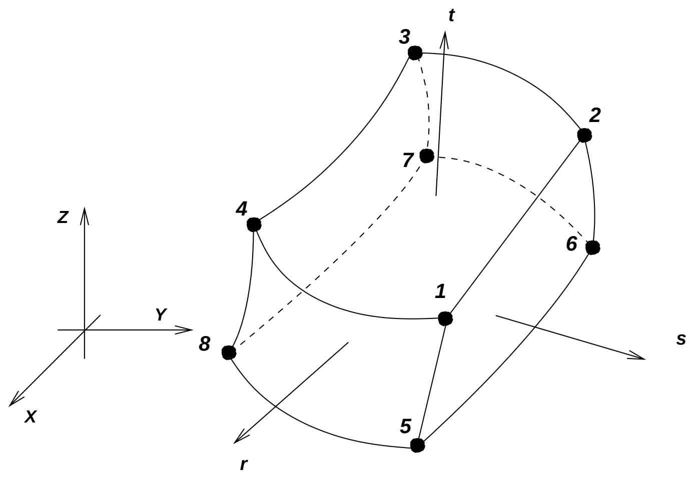

Figure 1.1: Node numbering for 8 node three dimensional element.

1.3.2 The Twenty Node Brick Element

```
element Brick20N eletag? node1? node2? node3? node4? node5? node6? node7?
        node8?  node9?  node10?  node11?  node12? node13? node14? node15?
        node16? node17? node18?  node19?  node20? matTag? bf1?  bf2? bf3?
        massDens?
```

The Brick8N element is the standard eight node three dimensional element implemented based on tensor operation. The arguments to construct the element are its tag, eletag, twenty nodes ordered according to Figure 1.2, the material tag, matTag, the body forces, bf1, bf2, bf3, and the mass density, massDens. By default, $3 \times 3 \times 3$ integration points are used. Users will be able to specify number of integration points very soon.

The valid queries to a Brick20N element when creating an ElementRecorder are “force”, “stiffness”, “stress”, “pq”, “gausspoint”, or “plastic”. For “stress” output, the six stress components from each Gauss point are output by the order: σ_x, σ_y, σ_z, τ_{xy}, τ_{xz}, τ_{yz}. The stresses can also be output in p and q format by using query “pq”, where p is the hydrostatic pressure, while q is the equivalent deviatoric stress. In this case, the stress state at one gauss point is printed in the pq format. For “gausspoint”, the coordinates of all Gauss points are printed out. For “plastic”, the equivalent deviatoric plastic strain from each Gauss point is output in the same order as the coordinates are printed.

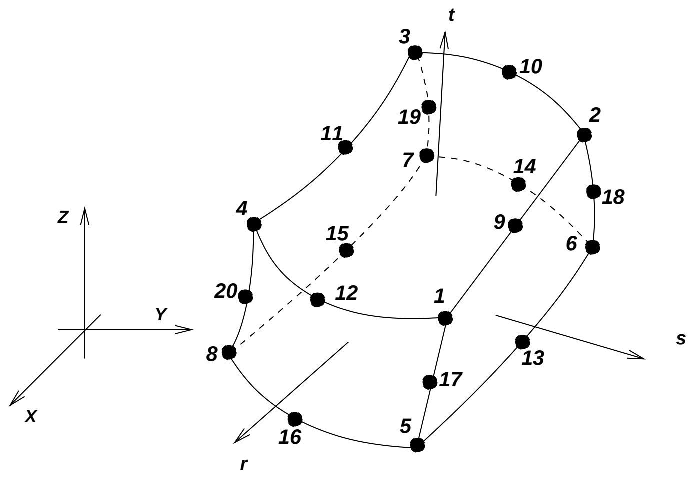

Figure 1.2: Node numbering for 20 node three dimensional element.

1.4 OpenSees Commands for u–p–U Elements

There are two different node elements:

- 8 node brick u-p-U element `Brick8N_u_p_U` and
- 20 node brick u-p-U element `Brick20N_u_p_U`

Due to the coexistence of the solid and fluid, there are 7 degrees of freedom at each node. The changes in the model building, analysis procedure commands are explained as below.

1.4.1 The model Command

The model command defines the ModelBuilder object to be constructed.

```
model BasicBuilder -ndm ndm? <-ndf ndf?>
```

The string -ndm that followed by an integer defines the dimension of the problem,i.e. one, two or three dimensional problem. The string -ndf that followed by an integer defines the number of degree-of-freedom at a node. The angle bracket around the -ndf ndf? mean these arguments are optional. By default, the number of degree-of-freedom at a node depends on the dimension of the problem. For ndm=1, ndf=1;for ndm=2,ndf=3;for ndm=3, ndf=6.

For upU element,each node has 3 solid displacements, one pore pressure, and 3 fluid displacements, so the degree-of-freedom of each node is 7. The ModelBuilder command for upU element is:

```
model BasicBuilder -ndm 3 -ndf 7
```

1.4.2 The fix Command

The fix command is used to construct the single-point boundary conditions.

```
fix nodeTag? (ndf values?)
```

where the nodeTag is the node needs to be constrained. The ndf values can be 1 or 0. If the value of i-th ndf is specified as 1, then the degree-of-freedom at the DOF is constrained. Otherwise 0 means that it is left free..

The fix command for upU element is:

```
fix nodeTag (7 values (1 or 0))
```

for example

```
fix 1 1 1 1 1 1 1 1
```

fix all the degree-of-freedom of Node 1.

```
fix 5 1 1 0 0 1 1 0
```

fix the solid and fluid displacement for Node 5 in x and y directions, while the z direction of solid ,fluid and pore pressure are left free.

1.4.3 The element Command

The element command is used to construct an Element object.

```
element eleType <specific element type args>
```

The second argument is the element type. The element names for the upU are given as Brick8N_u_p_U (for 8-node-brick-upU element) and Brick20N_u_p_U(for 20 node brick u-p-U element). The arguments in the angle bracket specify the properties of each type of element.

For upU element, the element command is:(for example 20-node-upU element):

```
element Brick20N_u_p_U eleTag? 1st_node? 2nd_node?...20th_node materialID?
x_body_force? y_body_force? z_body_force? porosity? alpha? solid_density?
fluid_density? x_permeability? y_permeability? z_permeability?
solid_bulk_modulus? fluid_bulk_modulus? pressure?
```

One upU element command might look like:

```
element Brick20N_u_p_U 1 5 6 7 8 1 2 3 4 13 14 15 16 9 10 11 12 17 18 19
20 1 0.0 0.0 -9.81 0.4 1.0 2.0 1.0 1.0e-5 1.0e-5 1.0e-5 1.0e5 1.0e5 0
```

The element is 20-node-brick-upU element Brick20N_u_p_U. The element tag is 1.The node numbers of the element is 5 6 7 8 1 2 3 4 13 14 15 16 9 10 11 12 17 18 19 20. The material tag is 1. The body forces in x and y directions are zeros, while in z direction is -9.81. The porosity of the element is 0.4, and alpha is 1.0. The solid density is 2.0, and fluid density is 1.0. The permeability in x, y and z directions are all 1.0e-5. The bulk moduli of solid and fluid are both 1.0e5, and the pressure is zero. Please note that the permeability here is not the classical Darcy's permeability with the unit of [length/time], but the classical Darcy's permeability divided by the unit weight of the fluid.

1.4.4 The pattern Command

The pattern command is used to construct the LoadPattern object. There are Load and Constraint objects for the pattern.

```
pattern patternType patternTag? <arguments for pattern type>
```

Currently, there are three valid types of pattern: Plain, UniformExcitation and MultipleSupport. The Plain pattern has the form:

```
pattern Plain patternTag? {TimeSeriesType and Args}{
load ...
sp ...
}
```

The arguments Plain is used to construct an ordinary LoadPattern object with a unique patterTag. The fourth argument is a list to construct the TimeSeries object associated with the LoadPattern object. The arguments load and sp are used to create a nodal load and single-point constraint.

In upU example, the Plain pattern of Linear type:

```
pattern Plain 2 Linear{
load 5  0 0 $p 0 0 0 0
load 6  0 0 $p 0 0 0 0
load 7  0 0 $p 0 0 0 0
load 8  0 0 $p 0 0 0 0
load 13 0 0 $np 0 0 0 0
load 14 0 0 $np 0 0 0 0
load 15 0 0 $np 0 0 0 0
load 16 0 0 $np 0 0 0 0
}
```

create a load pattern with vertical load $p(depends on the value p and np, if use command set p 5, then $p=5 and set np 10, then $np=10) acting on nodes 5 6 7 8 ,and $np acting on node 13 14 15 16.

After careful inspecting the derivations and resulting equations, following recommendations are made for application of loads on coupled systems:

- For drained (vertical) distributed load f: all of the load f must be loaded to the solid matrix, no load goes to the fluid part and the fluid DOF must remain free. That is, the loading may be considered as only effective load without any pore pressure load.
- For undrained (vertical) distributed load f: $(1-n)f$ is to be loaded to the solid matrix, nf is to be loaded load to the fluid part and the fluid DOF must be constrained.

1.5 OpenSees Command for the Domain Reduction Method

The syntax of the DRM command is :

`pattern PBowlLoading` loadpattrnTag? `-pbele` PBElmentFile? `-acce` acceFile? `-disp` dispFile? `-dt` dt? <`-factor` factor?> `-xp` +X? `-xm` -X? `-yp` -Y? `-ym` -Y? `-zp` +Z? `-zm` -Z?

where pattern and PBowlLoading are key words. loadpattrnTag? is the tag for this loadPattern. A user needs to supply an element file, acceleration file, displacement file following the identifiers `-pbele`, `-acce` and `-disp`, respectively. dt? is the time increment in the input data and is a parameter following identifier `-dt`. factor? is an optional parameter for scaling the equivalent forces. If not supplied, a default value 1.0 will be used. In particular, the interior boundary of the boundary layer is input by a set of coordinates, i.e. +X, -X, +Y, -Y, +Z and -Z, as shown in Figure 1.3.

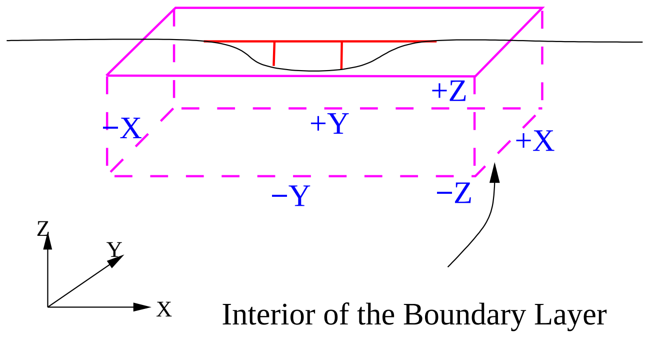

Figure 1.3: Sketch showing coordinates of the interior boundary.

1.6 Total Lagrangian Finite Deformation Brick

```
element TLFD20nBrick eleTag? nd01? nd02? ... nd20? bf1? bf2? bf3?
```

The TLFD20nBrick element is the the three dimensional element implemented based on the tensor operation. The argument to construct the element are its tag, `TLFD20nBrick`, element tag, `eleTag`, ordered twenty nodes, `nd01` to `nd20`, and the body forces, `bf1`, `bf2`, `bf3`. By default, $3 \times 3 \times 3$ integration points are used. The valid material tags for this element are hyperelastic material (see Tcl Commands for Large Deformation Hyperelastic Material Models) and hyperelasto-plastic material (see Tcl Commands for Large Deformation Hyperelasto-Plastic Material Models).

1.7 Large Deformation Hyperelastic Material Model Commands

1.7.1 Compressible Neo-Hookean Material Commands

```
nDMaterial FDElastic3D matTag? NeoHookean3D K? G? <rho?>
```

or

```
nDMaterial FDElastic3D matTag? NeoHookeanCompressible K? G? <rho?>
```

This model is a coupled model for its volumetric and isochoric parts. A compressible Neo-Hookean material is constructed using `nDMaterial` command. The argument `matTag` is used to uniquely identify this nDMaterial object among nDMaterial objects in the BasicBuilder object. The parameter `K` defines the material reference bulk modulus. The parameter `G` defines the material reference shear modulus. The optional parameter `rho` defines the material reference density, the default value of rho is zero.

Although this is named compressible Neo-Hookean material, it can also be used for near-compressible or incompressible material. In this case, make `K` a value far bigger than `G`.

1.7.2 Decoupled Neo-Hookean Material Commands

```
nDMaterial FDElastic3D matTag? DecoupledNH3D K? G? <rho?>
```

or

```
nDMaterial FDElastic3D matTag? DecoupledNeoHookean3D K? G? <rho?>
```

This model is a decoupled model for its volumetric and isochoric parts. A decoupled Neo-Hookean material is constructed using `nDMaterial` command. The argument `matTag` is used to uniquely identify this nDMaterial object among nDMaterial objects in the BasicBuilder object. The parameter `K` defines the material reference bulk modulus. The parameter `G` defines the material reference shear modulus. The optional parameter `rho` defines the material reference density, the default value of rho is zero.

1.7.3 Decoupled Logarithmic Material Commands

```
nDMaterial FDElastic3D matTag? DecoupledLog3D K? G? <rho?>
```

or

```
nDMaterial FDElastic3D matTag? DecoupledLogarithmic3D K? G? <rho?>
```

This model is a decoupled model for its volumetric and isochoric parts. A decoupled logarithmic material is constructed using `nDMaterial` command. The argument `matTag` is used to uniquely identify this nDMaterial object among nDMaterial objects in the BasicBuilder object. The parameter `K` defines the material reference bulk modulus. The parameter `G` defines the material reference shear modulus. The optional parameter `rho` defines the material reference density, the default value of rho is zero.

1.7.4 Decoupled Mooney-Rivlin-Simo Material Commands

```
nDMaterial FDElastic3D matTag? DecoupledMRS3D c1? c2? K? <rho?>
```

or

```
nDMaterial FDElastic3D matTag? DecoupledMooneyRivlinSimo3D c1? c2? K? <rho?>
```

This model is a decoupled model for its volumetric and isochoric parts. Its isochoric part is Mooney-Rivlin model and its volumetric part is Simo-Pister model. A decoupled Mooney-Rivlin-Simo material is constructed using `nDMaterial` command. The argument `matTag` is used to uniquely identify this nDMaterial object among nDMaterial objects in the BasicBuilder object. The parameter `c1` and `c2` define the material constants. The parameter `K` defines the material reference bulk modulus. The optional parameter `rho` defines the material reference density, the default value of rho is zero.

1.7.5 Decoupled Ogden-Simo Material Commands

```
nDMaterial FDElastic3D matTag? DecoupledOS3D N? c1? ... cN? m1? ... mN? K? <rho?>
```

or

```
nDMaterial FDElastic3D matTag? DecoupledOgdenSimo3D N? c1? ... cN? m1? ... mN? K? <rho?>
```

This model is a decoupled model for its volumetric and isochoric parts. Its isochoric part is Ogden model and its volumetric part is Simo-Pister model. A decoupled Ogden-Simo material is constructed using `nDMaterial` command. The argument `matTag` is used to uniquely identify this nDMaterial object among nDMaterial objects in the BasicBuilder object. The parameter `N`, `c1`, ..., `cN`, and `m1`, ..., `mN` define the material constants. The parameter `K` defines the material reference bulk modulus. The optional parameter `rho` defines the material reference density, the default value of rho is zero.

1.7.6 Decoupled Mooney-Rivlin Material Commands

```
nDMaterial FDElastic3D matTag? DecoupledMR3D c1? c2? <rho?>
```

or

```
nDMaterial FDElastic3D matTag? DecoupledMooneyRivlin3D c1? c2? <rho?>
```

This model is a decoupled isochoric model. A decoupled Mooney-Rivlin-Simo material is constructed using `nDMaterial` command. The argument `matTag` is used to uniquely identify this nDMaterial object among nDMaterial objects in the BasicBuilder object. The parameter `c1` and `c2` define the material constants. The optional parameter `rho` defines the material reference density, the default value of rho is zero.

1.7.7 Decoupled Ogden Material Commands

```
nDMaterial FDElastic3D matTag? DecoupledOgden3D N? c1? ... cN? m1? ... mN? <rho?>
```

This model is a decoupled isochoric model. A decoupled Ogden-Simo material is constructed using `nDMaterial` command. The argument `matTag` is used to uniquely identify this nDMaterial object among nDMaterial objects in the BasicBuilder object. The parameter `N`, `c1`, ..., `cN`, and `m1`, ..., `mN` define the material constants. The optional parameter `rho` defines the material reference density, the default value of rho is zero.

1.7.8 Decoupled Simo-Pister Material Commands

```
nDMaterial FDElastic3D matTag? DecoupledSP3D K? <rho?>
```

or

```
nDMaterial FDElastic3D matTag? DecoupledSimoPister3D K? <rho?>
```

This model is a decoupled volumetric model. A decoupled Simo-Pister material is constructed using `nDMaterial` command. The argument `matTag` is used to uniquely identify this nDMaterial object among nDMaterial objects in the BasicBuilder object. The parameter `K` defines the material reference bulk modulus. The optional parameter `rho` defines the material reference density, the default value of rho is zero.

1.8 Large Deformation Hyperelasto-Plastic Material Model Commands

1.8.1 Yield Surface Commands

```
set fdy "-fdYieldSurfaceType <parameter list>"
```

This command sets the yield surface variables `fdY` or `fdYield` to be the specific type. A list of parameters can be defined the yield surface and the number of parameters depend on the type of yield surface. Currently, fdYieldSurfaceType cab be Von-Mises and Druker-Prager yield surfaces, which are described in the following subsections.

Von-Mises Yield Surface Commands

```
set fdy "-VM Y0?"
```

`VM` stands for Von-Mises type, i.e. cylinder shaped yield surface. The parameter `Y0` defines the material yielding strength, i.e. when uniaxial loading, `Y0` is then the uniaxial yielding strength.

Druker-Prager Yield Surface Commands

```
set fdy "-DP fa? c? <ConeIndex?>"
```

`DP` stands for Druker-Prager type, i.e. cone shaped yield surface. The parameter `fa` defines the material friction angle with the unit of degree. The parameter `c` defines the material cohesion strength. The optional parameter `ConeIndex` is the index to show what the type of cone is used:

- `ConeIndex` = 0, compressive (outer) cone;
- `ConeIndex` = 1, tensile (inner) cone;
- `ConeIndex` = 2, mean cone;
- `ConeIndex` = 3, inner-tangent cone;
- default, compressive (outer) cone.

1.8.2 Potential Surface (Flow Rule) Commands

```
set fdf "-fdPotentialSurfaceType <parameter list>"
```

This command sets the potential surface variables `fdY` or `fdYield` to be the specific type. A list of parameters can be defined the potential surface and the number of parameters depend on the type of potential surface. Currently, fdPotentialSurfaceType cab be Von-Mises and Druker-Prager potential surfaces, which are described in the following subsections.

Von-Mises Potential Surface Commands

```
set fdf "-VM Y0?"
```

`VM` stands for Von-Mises type, i.e. cylinder shaped potential surface. The parameter `Y0` equals to the material yielding strength, i.e. when uniaxial loading, `Y0` is then the uniaxial yielding strength.

Druker-Prager Potential Surface Commands

```
set fdf "-DP da? <ConeIndex?>"
```

`DP` stands for Druker-Prager type, i.e. cone shaped yield surface. The parameter `da` equals to the material dilatant angle with the unit of degree. The optional parameter `ConeIndex` is the index to show what the type of cone is used:

- `ConeIndex` = 0, compressive (outer) cone;
- `ConeIndex` = 1, tensile (inner) cone;
- `ConeIndex` = 2, mean cone;
- `ConeIndex` = 3, inner-tangent cone;
- default, compressive (outer) cone.

1.8.3 Evolution Law Commands

There are two types of evolution laws:

- Isotropic (scalar) evolution law: `fdES` or `fdEvolution_S`;
- Kinematic (tensor) evolution law: `fdET` or `fdEvolution_T`.

Linear and/or Saturated Isotropic Evolution Law

```
set fdes "-LS H? <qs? beta?>"
```

`LS` stands for Linear and/or Saturated Isotropic Evolution Law. The parameter `H` is the linear isotropic hardening modulus. The optional parameters `qs` and `beta` are saturated type isotropic hardening constants; if they are used, then it is the combination of linear and saturated isotropic hardening. This hardening rule can be applied to any scalar internal variable, such as the diameter of Von-Mises yield surface.

Linear Kinematic Evolution Law

```
set fdet "-Linear H?"
```

`Linear` stands for Linear kinematic Evolution Law. The parameter `H` is the linear isotropic hardening modulus. This hardening rule can be applied to any tensorial internal variable, such as the center of Von-Mises yield surface.

1.8.4 Hyperelasto-Plastic (Finite Deformation EP) Material Commands

```
nDMaterial FDEP3D mTag? hemTag? -fdY $fdy -fdF $fdf <-fdES $fdes> <-fdET $fdet>
```

A hyperelasto-plastic material is constructed using `nDMaterial` command. The argument `mTag` is used to uniquely identify this nDMaterial object among nDMaterial objects in the BasicBuilder object. The key word `FDEP3D` cab be replaced by `FiniteDeformationEP3D`. The argument `hemTag` is the previously defined finite deformation elastic (hyperelastic) material tag. The other parameters include previously defined yield surface object `fdY`, potential surface object `fdF`, optional isotropic evolution law object `fdES`, and optional kinematic evolution law object `fdET`. The key word `fdY` can be replaced by `fdYield`, and `fdF` can be replaced by `fdFlow`.