TECGRAF
PUC-RIO

GeMA – Fault Reactivation 2D

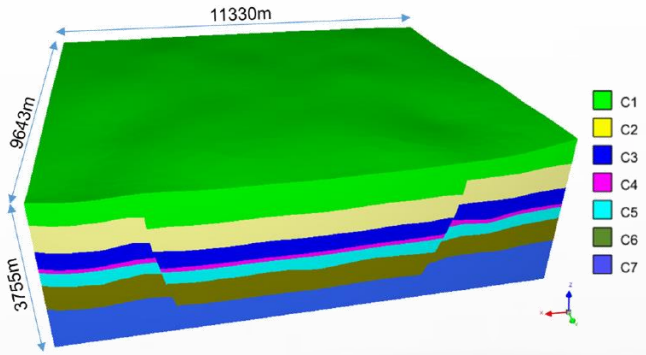
Version 1.0

Key Example Points

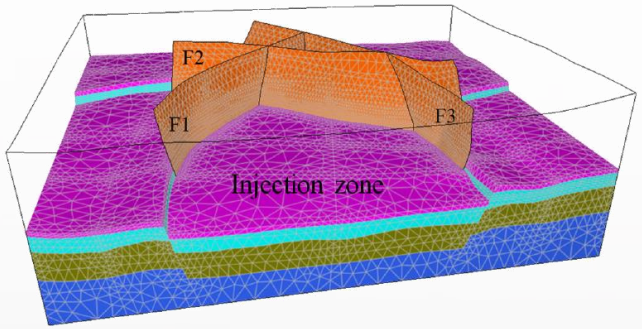
This example shows:

- How to setup models for fault reactivation analysis
- How to use a coupled HM interface element as a fracture to represent a geological fault.
- How to create a Multiphysics simulation in GeMA where one of the physics represents explicitly the fractures and another physics represent the porous media.
- Investigate the impact of the fluid injection on the natural fractures due to the changes on the stress field around the geological fault.

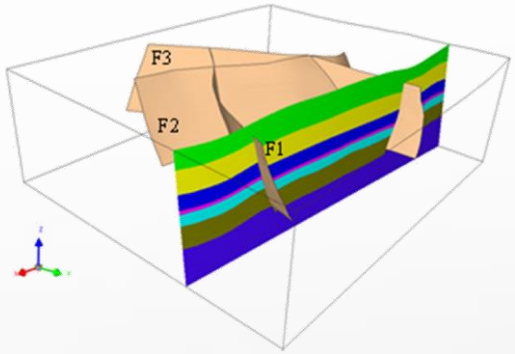
Example



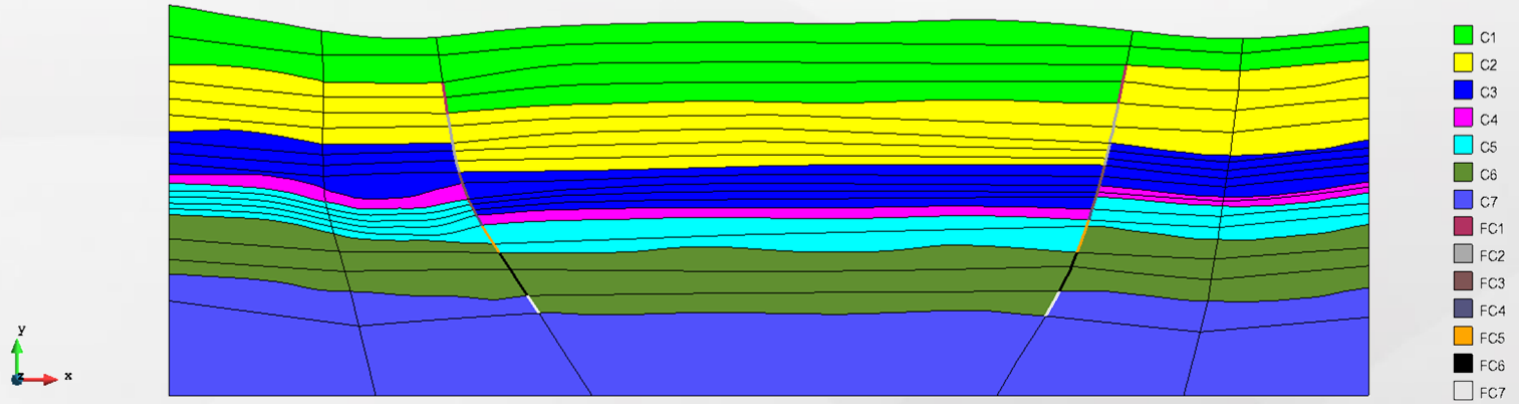
Tridimensional representation of the model



Natural faults view



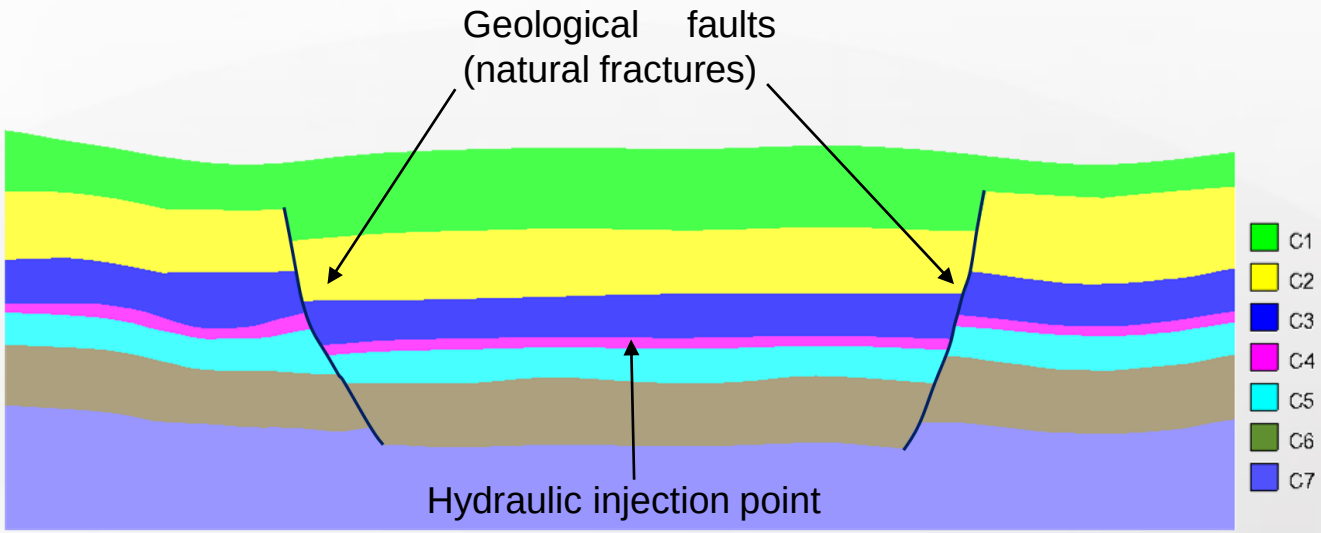
2D Section plane view in the 3D model



Layers and fault layers of the section

The problem

In this problem, a multi layer porous media with two main natural fractures is submitted to an hydraulic injection at the **reservoir** (pink layer). The bottom is vertically constrained and both sides are horizontally constrained. Natural fractures are represented by zero-thickness interface elements.



Simulation file: fault_reactivation.lua

Layer properties				
Layer	E(GPa)	ν	c (MPa)	ϕ(deg)
C1	4.0	0.38	0.5	20.0
C2	5.5	0.37	0.5	20.0
C3	6.0	0.36	0.5	20.0
C4	7.0	0.35	0.5	20.0
C5	8.0	0.34	0.5	20.0
C6	8.5	0.33	0.5	20.0
C7	9.0	0.32	0.5	20.0

Fault properties				
Layer	k_n(GPa/m)	k_s(GPa/m)	c (MPa)	ϕ(deg)
FC1	8.0	1.45	0.5	20.0
FC2	11.0	2.01	0.5	20.0
FC3	12.0	2.21	0.5	20.0
FC4	14.0	2.59	0.5	20.0
FC5	16.0	2.99	0.5	20.0
FC6	17.0	3.20	0.5	20.0
FC7	18.0	3.41	0.5	20.0

Model file: State variable

For a coupled Hydro-Mechanical analysis, displacement and pore pressure state variables are required.

```
-- State variables
StateVar{id = 'u', dim = 2, description = 'Displacements in the X and Y directions', unit = 'm',
         format = '8.4f', groupName = 'mechanic'}

StateVar{id = 'P',  description = 'Pore pressure', unit = 'kPa',
         format = '8.4f', groupName = 'hydraulic'}
```

The group name is used by the fem solver to group types of state variables and to define tolerances for each one

Both continuum and interface physics use the same state variables

Model file: Hydromechanical properties

```
PropertySet
{
  id          = 'MatProp',
  typeName    = 'GemaPropertySet',
  description = 'Material properties',
  properties  = {
    {id = 'E',     description = 'Elasticity modulus',            unit = 'kPa'},
    {id = 'nu',    description = 'Poisson ratio'},
    {id = 'K',     description = 'Hydraulic permeability in x',   unit = 'm/s'},
    {id = 'gw',    description = 'Specific weight of water',      unit = 'kN/m^3'},
    {id = 'Kww',   description = 'Bulk modulus of water',         unit = 'kPa'},
    {id = 'Pht',   description = 'Porosity'},
    {id = 'Kn',    description = 'Normal elastic stiffness',      unit = 'kPa/m'},
    {id = 'Ks',    description = 'Shear elastic stiffness at direction 1', unit = 'kPa/m'},
    {id = 'Cf',    description = 'Faul cohesion',                 unit = 'kPa'},
    {id = 'Phif',  description = 'Fault friction angle',          unit = 'degree'},
    {id = 'Psif',  description = 'Fault dilation angle',          unit = 'degree'},
    {id = 'Tcut',  description = 'Tensile normal strength',       unit = 'kPa'},
    {id = 'Gap',   description = 'Initial gap opening',           unit = 'm'},
    {id = 'Ufw',   description = 'Dynamic fluid viscosity',       unit = 'kPa*s'},
    {id = 'Mfw',   description = 'fracture bulk compressibility', unit = 'kPa/m'},
    {id = 'Lkt',   description = 'Leakoff at top',                unit = 'm/kPa/s'},
    {id = 'Lkb',   description = 'Leakoff at bottom',             unit = 'm/kPa/s'},
    {id = 'Iopen', description = 'Intially open interface'},
    {id = 'materialHM', description = 'Hydro-mechanics material type',
     constMap = constants.CoupledHMFemPhysics.materialModels},
    {id = 'h',     description = 'Element thickness', unit = 'm'},
  },
```

Hydromechanical properties for continuum elements

Hydromechanical properties for interface elements

Model file: Hydromechanical properties

```
values = {
  -- Layer 1 (C1)
  {E = 4.00e+06, nu =   0.38, h =      1, materialHM = 'poroElastic',
   K =1.06e-10, gw = 1.00e+01, Pht =1.00e-01, Kww =2.20e+06},
  -- Layer 2 (C2)
  {E = 5.50e+06, nu =   0.37, h =      1, materialHM = 'poroElastic',
   K =1.06e-10, gw = 1.00e+01, Pht =1.00e-01, Kww =2.20e+06},
  ...

  -- Fault 1 (FC1)
  {Kn = 8.00e+06, Ks = 1.45e+06, Gap = 1.00e-03, Ufw = 1.00e-06,  Lkt = 1.00e+00, Lkb = 1.00e+00,
   Mfw = 0.00e+00, Cf = 500, Phif = 20, Psif = 20, Tcut = 0, material = 'poroInterfaceMC', Iopen = 0, h = 1},
  -- Fault 2 (FC2)
  {Kn = 1.10e+07, Ks = 2.01e+06, Gap = 1.00e-03, Ufw = 1.00e-06,  Lkt =1.00e+00, Lkb = 1.00e+00,
   Mfw = 0.00e+00, Cf = 500, Phif = 20, Psif = 20, Tcut = 0, material = 'poroInterfaceMC', Iopen = 0, h = 1},
  ...
}
}
```

Material properties for each layer in the model

Open fracture considers longitudinal fluid

Defines the initial fracture condition (open or closed).

```
-- Iopen = 0 closed fracture
-- Iopen = 1 opened fracture
```

Model file: Hydromechanical properties

```
mesh_groups_elements = dofile('$SIMULATIONDIR/IntMesh.lua')
```

Holds the interface elements positions

```
local mesh_elements =
{
  {cellType = 'quad4', cellGroup = 'QUP_1', cellList = mesh_QUAD_1_elements, MatProp = 1},
  {cellType = 'quad4', cellGroup = 'QUP_2', cellList = mesh_QUAD_2_elements, MatProp = 2},
  {cellType = 'quad4', cellGroup = 'QUP_3', cellList = mesh_QUAD_3_elements, MatProp = 3},
  {cellType = 'quad4', cellGroup = 'QUP_4', cellList = mesh_QUAD_4_elements, MatProp = 4},
  {cellType = 'quad4', cellGroup = 'QUP_5', cellList = mesh_QUAD_5_elements, MatProp = 5},
  {cellType = 'quad4', cellGroup = 'QUP_6', cellList = mesh_QUAD_6_elements, MatProp = 6},
  {cellType = 'quad4', cellGroup = 'QUP_7', cellList = mesh_QUAD_7_elements, MatProp = 7},
  {cellType = 'int2dl4', cellGroup = 'Int2UP_8', cellList = mesh_groups_elements[1], MatProp = 8},
  {cellType = 'int2dl4', cellGroup = 'Int2UP_9', cellList = mesh_groups_elements[2], MatProp = 9},
  {cellType = 'int2dl4', cellGroup = 'Int2UP_10', cellList = mesh_groups_elements[3], MatProp = 10},
  {cellType = 'int2dl4', cellGroup = 'Int2UP_11', cellList = mesh_groups_elements[4], MatProp = 11},
  {cellType = 'int2dl4', cellGroup = 'Int2UP_12', cellList = mesh_groups_elements[5], MatProp = 12},
  {cellType = 'int2dl4', cellGroup = 'Int2UP_13', cellList = mesh_groups_elements[6], MatProp = 13},
  {cellType = 'int2dl4', cellGroup = 'Int2UP_14', cellList = mesh_groups_elements[7], MatProp = 14},
}
```

Groups of Interface elements

Groups of Continuum elements

Solution file: Physic definition

In order to start the simulation with the initial stresses, a Geostatic step is necessary

```
PhysicalMethod {
  id          = 'Geostatic',
  typeName = 'CoupledHMFemPhysics.PlaneStrain',
  type        = 'fem',

  mesh           = 'mesh',
  elementGroups = {'QUP_1', 'QUP_2', 'QUP_3',
                   'QUP_4', 'QUP_5', 'QUP_6', 'QUP_7'},
  materials      = {'poroElastic'},
  ruleSet        = 1,
  isoParametric = true,
  boundaryConditions = {'bc'},
  properties     = { material = 'materialHM' }
}
```

- Plane strain analysis from coupled HM Fem physic
- Sets of elements to represent the layers of the model
- Element with poro pressure d.o.f
- Option to set the boundary conditions to be taken into consideration

Solution file: Physic definition

After the Geostatic step, a second step for a transient analysis is performed, including other boundary conditions

```
PhysicalMethod {
  id          = 'Consolidation',
  typeName = 'CoupledHMFemPhysics.PlaneStrain',
  type        = 'fem',

  mesh           = 'mesh',
  elementGroups = {'QUP_1', 'QUP_2', 'QUP_3',
                   'QUP_4', 'QUP_5', 'QUP_6', 'QUP_7'},
  materials      = {'poroElastic'},
  ruleSet        = 1,
  isoParametric = true,
  boundaryConditions = {'bc', 'bfm'},
  properties     = { material = 'materialHM' }

}
```

← Plane strain analysis from coupled HM Fem physic

← Sets of elements to represent the layers of the model

← Element with poro pressure d.o.f

← Option to set the boundary conditions to be taken into consideration

Solution file: Physic definition

The coupled interface element is available through the interface object from the CoupledHMFemPhysics plugin.

```
PhysicalMethod {
  id       = 'interfaceHM',
  typeName = 'CoupledHMFemPhysics.Interface',
  type     = 'fem',

  mesh          = 'mesh',
  elementGroups = {'Int2UP_8', 'Int2UP_9', 'Int2UP_10',
                   'Int2UP_11', 'Int2UP_12','Int2UP_13',
                   'Int2UP_14'},
  materials     = {'poroInterfaceMC'},
  ruleSet       = 1,
  isoParametric = false,
  permeabilityUpdate = true,
  properties    = { material = 'materialHM' }
}
```

Coupled interface object from coupled HM Fem physic

Sets of interface elements to represent the fault changes between layers

Interface element with pore pressure d.o.f and Mohr-Coulomb criteria

Option to allow permeability as a function of fracture aperture update for longitudinal fluid flow

Solution file: Geostatic process

In order to compute the geostatic stresses, we must know the vertical difference between each node and the reference surface

```
function getRefCoord(xc)
   Data = dofile('$SIMULATIONDIR/SintReference.lua')
   local yc
   for i = 1, #Data-1  do
     local _data0 = Data[i]
     local _data2 = Data[i+1]
     if (xc >= _data0[2] and xc <= _data2[2]) then
        -- Calculate the vertical coordinate of the reference surface from a
        -- known horizontal coordinate.
        yc = _data0[3] + (xc - _data0[2])*(_data2[3] - _data0[3])/(_data2[2]-_data0[2])
     end
   end
   return yc
end
```

Holds the top nodes of the model. They should be organized ascending by its horizontal coordinate

Solution file: Geostatic process

```
function initCond ()
    local mesh = modelData:mesh('mesh')
    -- Stress accessor for the old sress state
    local accS = mesh:gaussAttributeAccessor('S', 1, true)
    -- Node coordinate accessors
    local coordAc = mesh:nodeCoordAccessor()

    local gammaSub = 12.5
    local K0x = 0.53
    local K0z = 0.53
    local Y0  = 0.0  -- sea level
    --------------------
    -- Initial Stress --
    --------------------
    for i=1, mesh:numCells() do
        local e   = mesh:cell(i)
        -- integration rule
        local ir  = mesh:elementIntegrationRule(e:type(), 1)
        -- element shape
        local shp = assert(e:shape())
```

```
    if (e:type() == 'quad4') then      <-- solid elements
       -- elemental node coordinates
       local Xnode = e:nodeMatrix(coordAc, true)
       -- For each integration point
       for j=1, ir:numPoints() do
          -- get integration point coordinates
          local ip, w = ir:integrationPoint(j)
          -- integration point in Cartesian coordinates
          local coord = shp:naturalToCartesian(ip, Xnode)
          -- get vertical coord of reference surface
          local _Yref = getRefCoord(coord(1))
          -- fill stress according
          local Sv = gammaSub*(coord(2) - _Yref)
          local Sh = K0x*Sv
          local SH = K0z*Sv
          local v = {Sh, Sv, SH, 0}
          accS:setValue(e, j, v)
       end
    end
```

solid elements

Solution file: Geostatic process

Interface elements – Stress rotation from local reference to global reference

```
if ( e:type() == 'int2dl4') then
   -- elemental node coordinates
   local Xnode = e:nodeMatrix(coordAc, true, 'vertices')
   assert(ir:numPoints() == 2)

   -- For each integration point
   for j=1, ir:numPoints() do
     -- Get integration point coordinates in
     -- Cartesian coordinates
     local ip, w = ir:integrationPoint(j)
     local coord = shp:naturalToCartesian(ip, Xnode)

     -- fill stress
     local Sv, Sh, Sn, Tau, theta
     local x1 = Xnode(1, 1)
     local y1 = Xnode(2, 1)
     local x2 = Xnode(1, 2)
     local y2 = Xnode(2, 2)
     local delta_x = x2-x1
     local delta_y = y2-y1
```

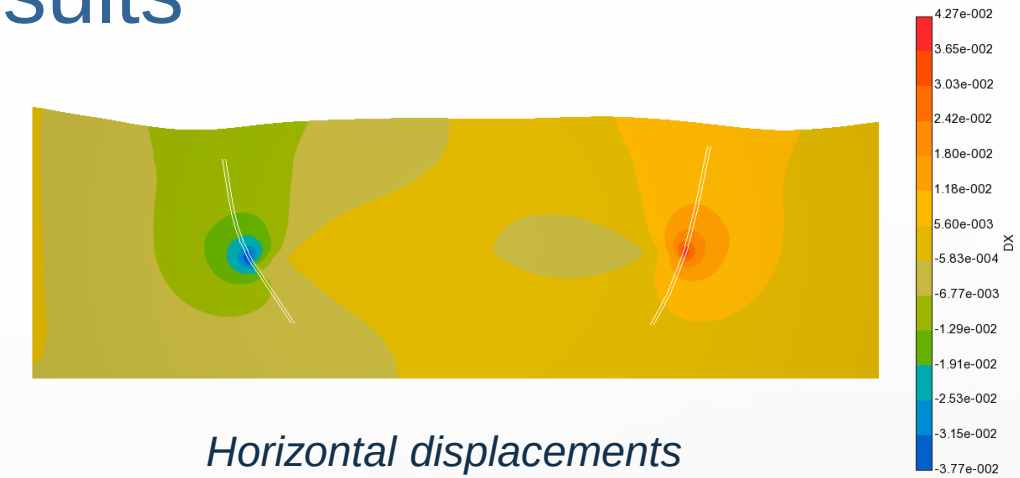
Interface elements

```
     -- Theta angle in radians
     local theta = math.atan(delta_y/delta_x)
     if (delta_x < 0) then
       theta = math.pi + theta
     end

     -- Get vertical coord of reference surface
     local _Yref = getRefCoord(coord(1))

     -- Fill stress according
     local Sv = gammaSub*(coord(2) - _Yref)
     local Sh = K0x*Sv

     Sn=Sh*math.sin(theta)*math.sin(theta) +
        Sv*math.cos(theta)*math.cos(theta)
     Tau=(Sv-Sh)*math.sin(theta)*math.cos(theta)

     local v = {0, Sn, 0, Tau}
     accS:setValue(e, j, v)
   end
 end
end
```

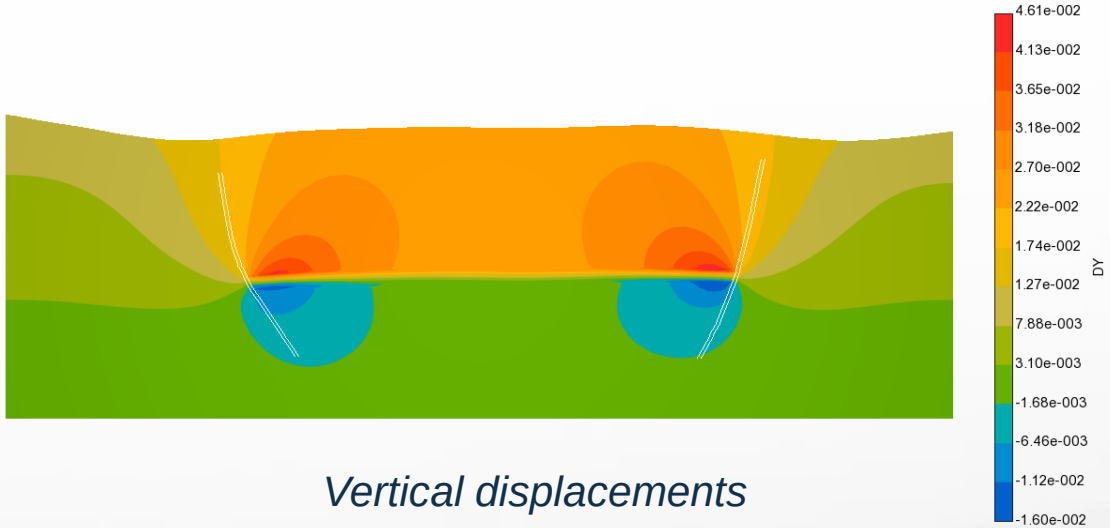
Solution file: Orchestration

```
function ProcessScript()
  ----------------------
  -- Initial condition
  ----------------------
  initCond()
  -----------------
  -- Geostatic step
  -----------------
  -- Create the solver model
  local solverG = fem.init({'interfaceHM','Geostatic'},
                           'solver', solverOptions)
  -- Prepare the file where results will be saved
  local file = io.prepareMeshFile('mesh',
        '$SIMULATIONDIR/out/$SIMULATIONNAME', 'nf',
        {'u', 'P'}, {'S', 'E','STR'},
        {split = true, saveDisplacements = true})
  fem.geostatic(solverG)
  -- Saves initial state to the file
  io.addResultToMeshFile(file, 0.0)
  -------------------------------
  -- Consolidation step
  -------------------------------
  local solver = fem.init({'interfaceHM', 'Consolidation'},
                          'solver', solverOptions)
  local dt        = solverOptions.timeInitIncrement
  local FinalTime = solverOptions.timeMax
  local Time      = dt
  local LastStep  = false
  while (Time <= FinalTime) do
     print('------------------')
     print(('Consolidation step - time = %1 s'):num(Time))
     print('------------------')
     -- Run fully coupled analysis.  Solver returns a suggested
     -- time-step for the next iteration
     dt = fem.step(solver, dt)
     -- Save results
     io.addResultToMeshFile(file, Time)

     -- Adjust time to guarantee that the last iteration will be
     -- on the requested final simulation time
     if (Time + dt >= FinalTime and not LastStep) then
        dt       = FinalTime - Time
        Time     = FinalTime
        LastStep = true
     else
        Time = Time + dt
     end
  end
  -- Closes the result file
  io.closeMeshFile(file)
end
```

Results

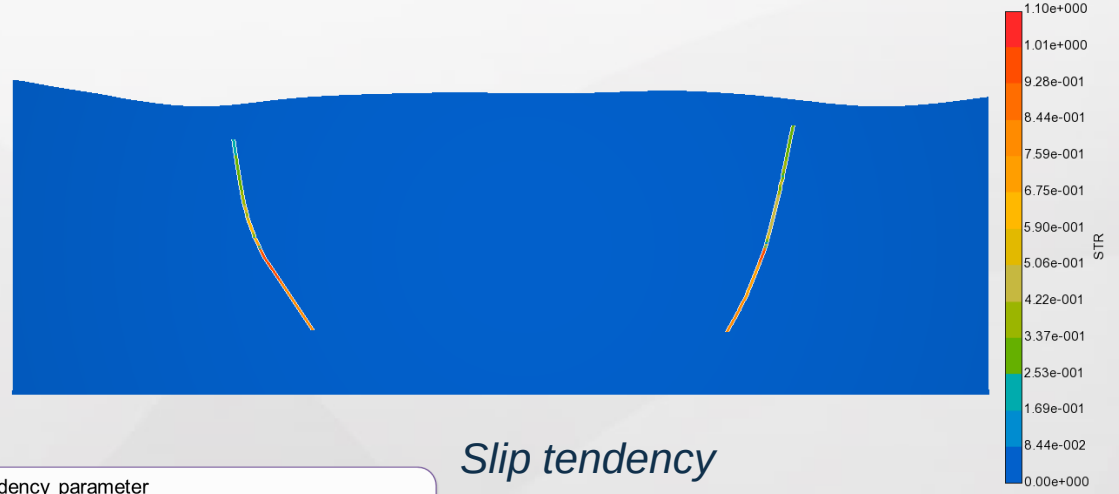


Horizontal displacements



Vertical displacements

Poro pressure distribution



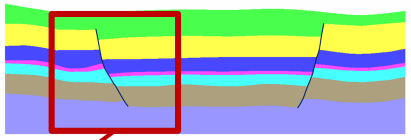
Slip tendency

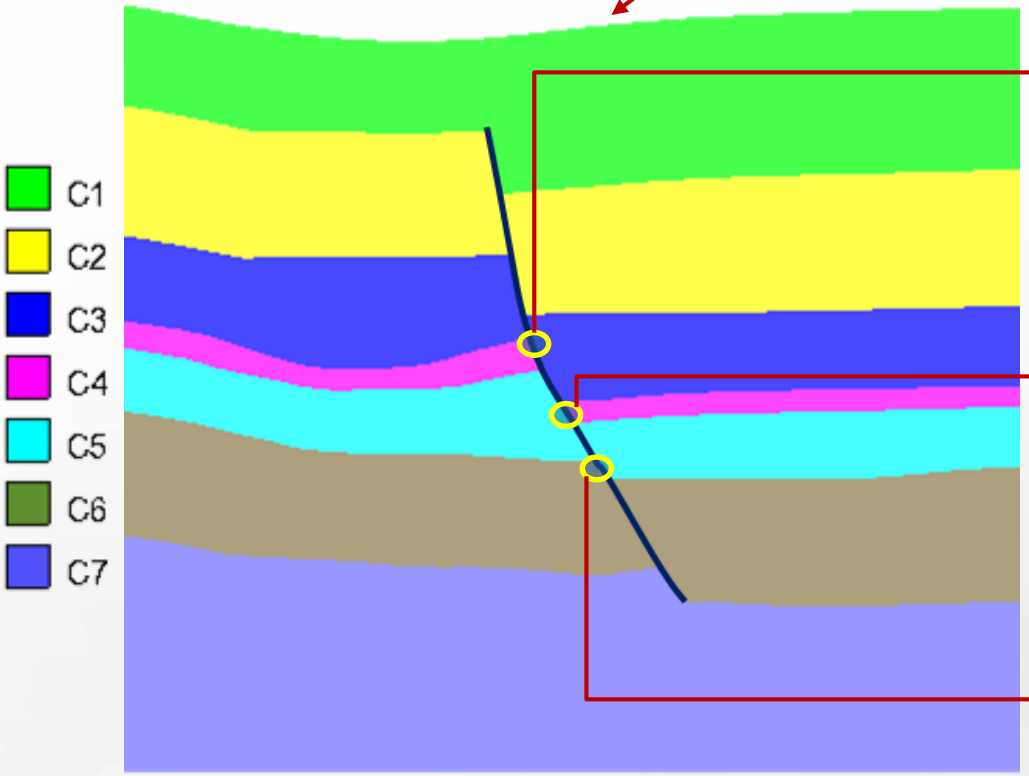
Slip Tendency parameter

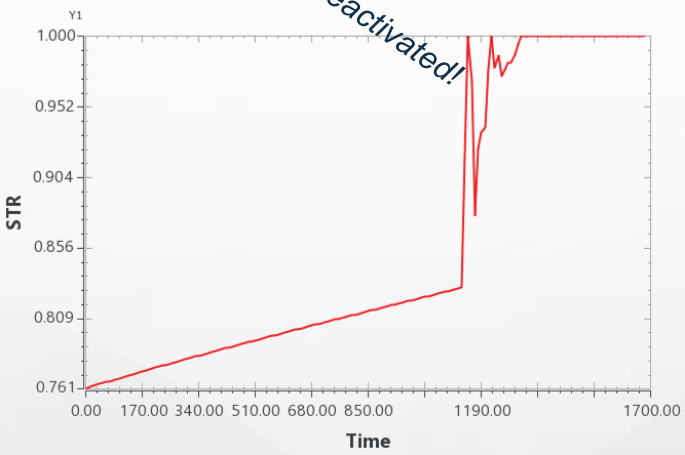
$$ST = \frac{\tau_s}{c' + \sigma'_{nor} tg\phi} \qquad 0 \le ST \le 1$$

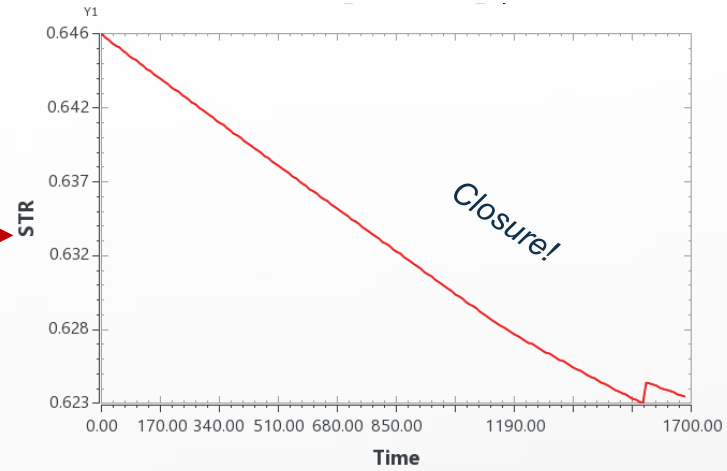
$ST = 1 \rightarrow$ Fault reactivation

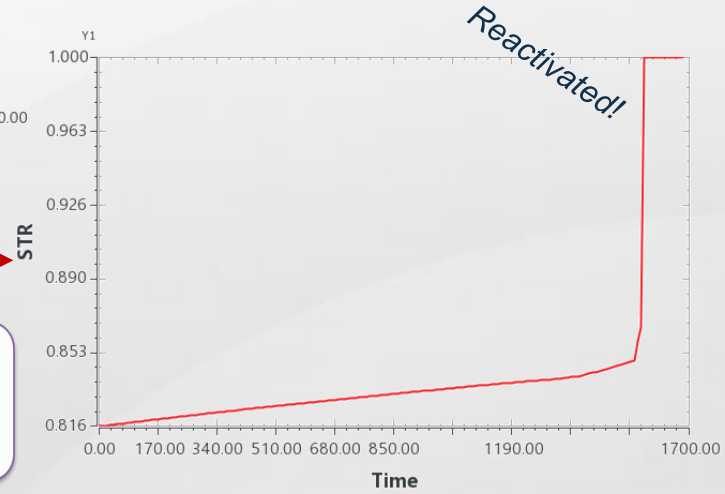
Results











Slip tendency at different levels of layers.

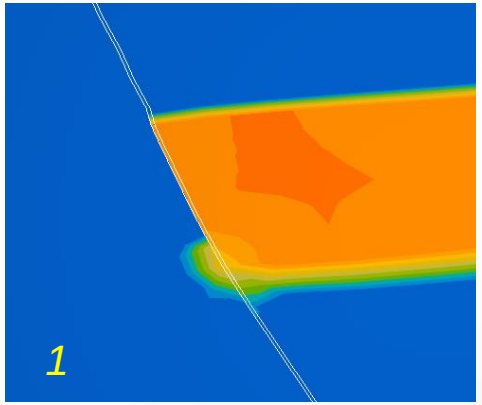
Slip Tendency parameter

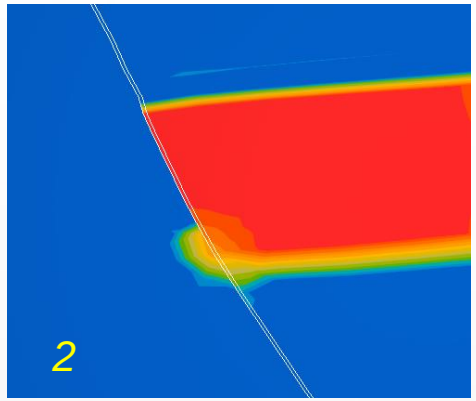
$$ST = \frac{\tau_s}{c' + \sigma'_{nor} tg\phi}$$

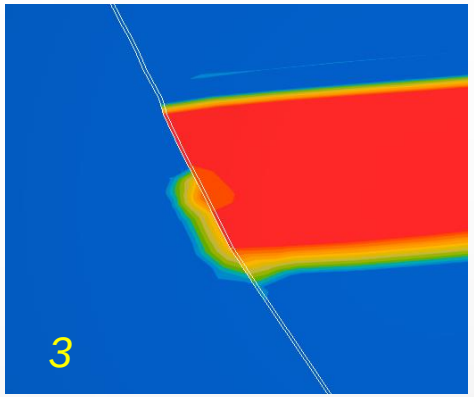
$$0 \leq ST \leq 1$$

$ST = 1 \rightarrow$ Fault reactivation

Results







Pore pressure evolution at the left fault