GeMA – Numerical Modelling of Hydraulic fracturing

Version 1.0

Key Example Points

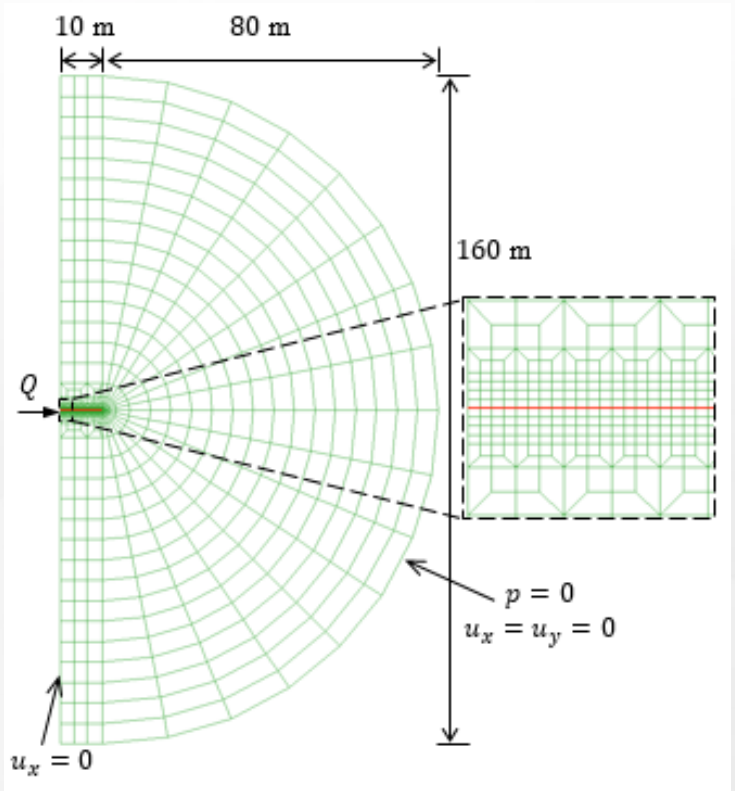
This example shows:

- How to simulate hydraulic fracturing using interface elements
- How to use a three-noded coupled HM interface element and cohesive constitutive model
- How to define initially open fractures
- How to setup specific options for interface elements
- Orchestration script for hydraulic fracturing in impermeable porous media.

1 – HYDRAULIC FRACTURE PROPAGATION IN PERMEABLE POROUS MEDIA (KGD MODEL)

1.- The problem

The KGD problem consists of a half-circle plate with one layer of coupled HM interface elements inserted in the middle. The geometry, boundary conditions and finite element mesh of the problem are detailed below.



The borehole is hydraulically pressurized until a fracture is initiated and propagated through the rock formation.

Simulation file: hydDrivenKGD.lua

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

The HM coupled physics for continuum and interface elements require the same state variables

Model file: Hydromechanical properties

```
PropertySet
{ id          = 'HMProp',
  typeName    = 'GemaPropertySet',
  description = 'Material properties',
  properties  = {
    {id = 'E',   description = 'Elasticity modulus',       unit = 'kPa'},
    {id = 'nu',  description = 'Poisson ratio'},
    {id = 'K',   description = 'Hydraulic permeability',   unit = 'm/s'},
    {id = 'gw',  description = 'Specific weight of water', unit = 'kN/m3'},
    {id = 'Pht', description = 'Porosity'},
    {id = 'Kn',  description = 'Normal elastic stiffness', unit = 'kPa/m'},
    {id = 'Ks',  description = 'Shear elastic stiffness',  unit = 'kPa/m'},
    {id = 'Sn',  description = 'Normal traction',          unit = 'kPa'},
    {id = 'Ts',  description = 'Shear traction',           unit = 'kPa'},
    {id = 'ed',  description = 'Equivalent separation',    unit = 'm'},
    {id = 'Gap', description = 'Initial gap opening',      unit = 'm'},
    {id = 'Ufw', description = 'Dynamic fluid viscosity',  unit = 'kPa*s'},
    {id = 'Lkt', description = 'Leakoff at top',           unit = 'm/(kPa*s)'},
    {id = 'Lkb', description = 'Leakoff at bottom',        unit = 'm/(kPa*s)'},
    {id = 'material', description = 'Mechanical material type', constMap = constants.CoupledHMFemPhysics.materialModels}
  },
  values = {
  {E = 15.0e+06, nu = 0.2, K = 9.8e-09, gw = 10.0, Pht = 0.2, Kn = 15.0e+09, Ks = 15.0e+09, Sn = 1000.0,
   Ts = 1000.0, ed = 1.92e-4, Gap = 2.0e-3, Ufw = 1e-6,  Lkt = 0.0e-6, Lkb = 0.0e-6, material = 'poroElastic'},
  {E = 15.0e+06, nu = 0.2, K = 9.8e-09, gw = 10.0, Pht = 0.2, Kn = 15.0e+09, Ks = 15.0e+09, Sn = 1000.0,
   Ts = 1000.0, ed = 1.92e-4, Gap = 2.0e-3, Ufw = 1e-6,  Lkt = 1.0e-8, Lkb = 1.0e-8, material = 'cohesiveLinearSoftening'},
  }
}
```

Hydromechanical properties for continuum elements

Hydromechanical properties for interface elements

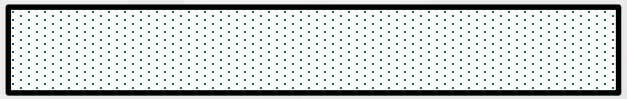
Defines the cohesive model: cohesive linear softening or cohesive exponential softening. The material must be in agreement with the defined parameters

Model file: Hydromechanical properties

```
PropertySet
{
  id          = 'SecProp',
  typeName    = 'GemaPropertySet',
  description = 'Section properties',
  properties  = {
    {id = 'h',     description = 'Element thickness', unit = 'm'},
    {id = 'Iopen', description = 'Intially open interface'},
  },
  values = {
    {h =  1.0, Iopen = 0}, -- Iopen = 0 close fracture
    {h =  1.0, Iopen = 1}, -- Iopen = 1 open fracture
  }
}
```

Defines the initial fracture condition (open or closed).

Iopen=1: Initially open fracture allows longitudinal fluid



Iopen=0: No flow inside closed fracture

Solution file: Physic definition

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

Hydraulic fracturing simulation requires a cohesive constitutive mode: cohesive linear softening or cohesive exponential softening.

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

  mesh                = 'mesh',
  permeabilityUpdate = true,
  materials           = {'cohesiveLinearSoftening'},
  isoParametric       = false,
  ruleSet             = 1,
  boundaryConditions = {'bc2','bc3'},
}
```

Coupled interface object from coupled HM Fem physic.

Option to allow permeability in function of fracture aperture for longitudinal fluid flow.

This option defines the cohesive constitutive model.

Solution file: Orchestration

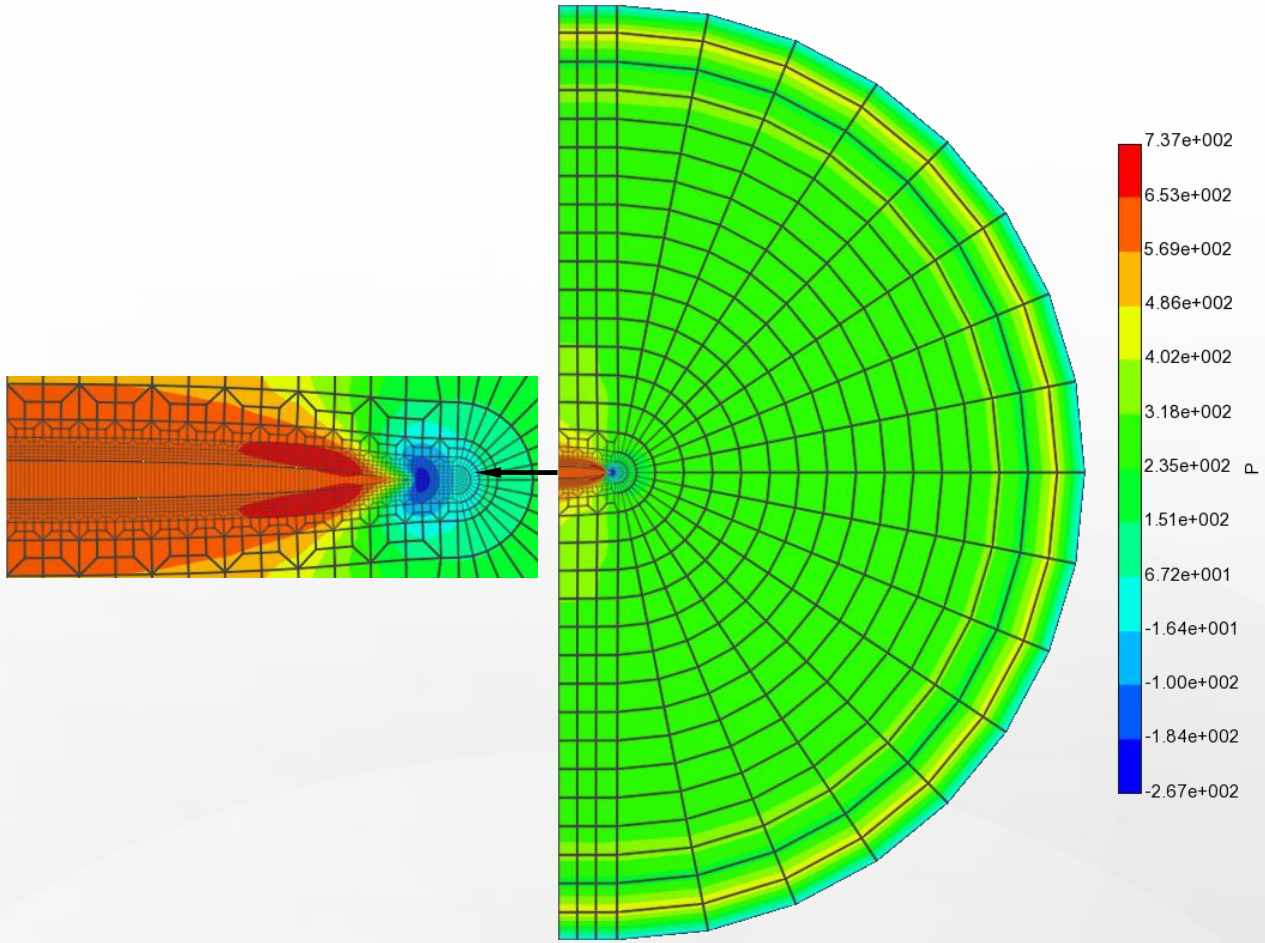
Two physics objects are needed, one for the continuum and another for interface elements

```
function ProcessScript()
  -- Create the solver model
  local solver = fem.init({'Rock', 'HMCohesive'}, 'solver', printOptions, solverOptions)

  -- Prepare the file where results will be saved
  local file = io.prepareMeshFile('mesh', '$SIMULATIONDIR/out/hydDrivenKGD', 'nf', {'u','P'}, {'S', 'E','Sdv'},
                                  {split = true, saveDisplacements = true})

  -- Saves initial state to the file
  io.addResultToMeshFile(file, 0.0)

  -- Fluid injection step
  local dt        = solverOptions.timeInitIncrement
  local FinalTime = solverOptions.timeMax
  local Time      = dt
  local LastStep  = false
  while (Time <= FinalTime) do
     print('------------------')
     print(tr('Injection step - time = %1 s'):num(Time))
     print('------------------')
     -- Run transient analysis.  Solver returns a suggested time-step
     -- for the next iteration
     dt = fem.step(solver, dt)

     -- Save results
     io.addResultToMeshFile(file, Time)
```

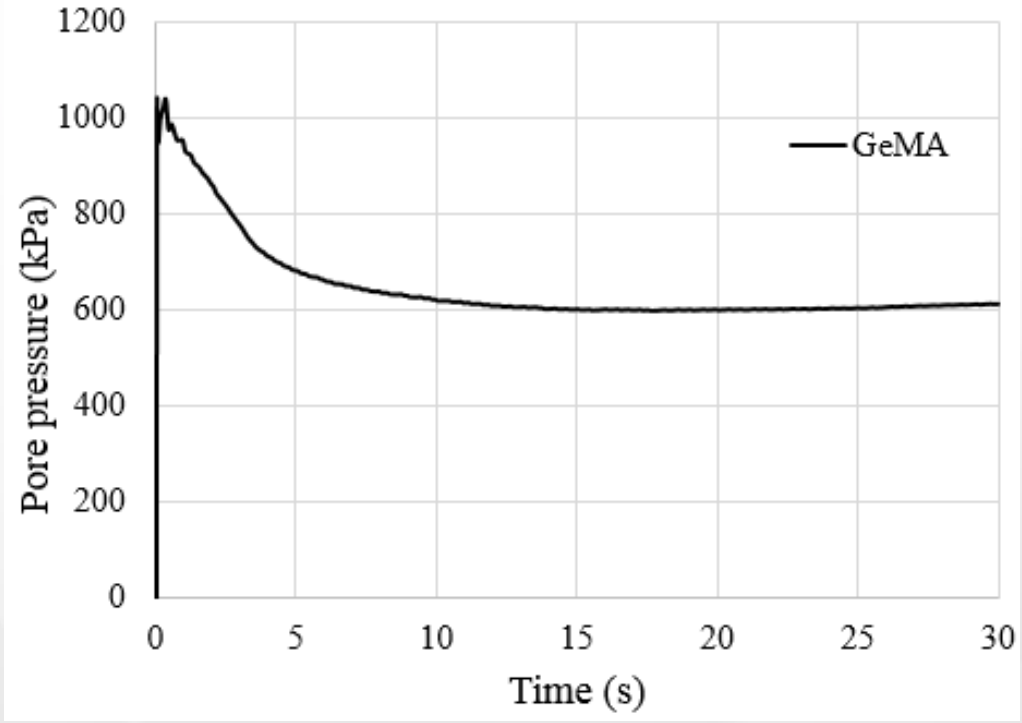
Solution file: Orchestration script

```
    -- 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
      if equal(dt, 0.0) then break end -- Test special case when Time == FinalTime
    else
      Time = Time + dt
    end

    print('Time = ' .. Time)
  end

  -- Closes the result file
  io.closeMeshFile(file)
end
```

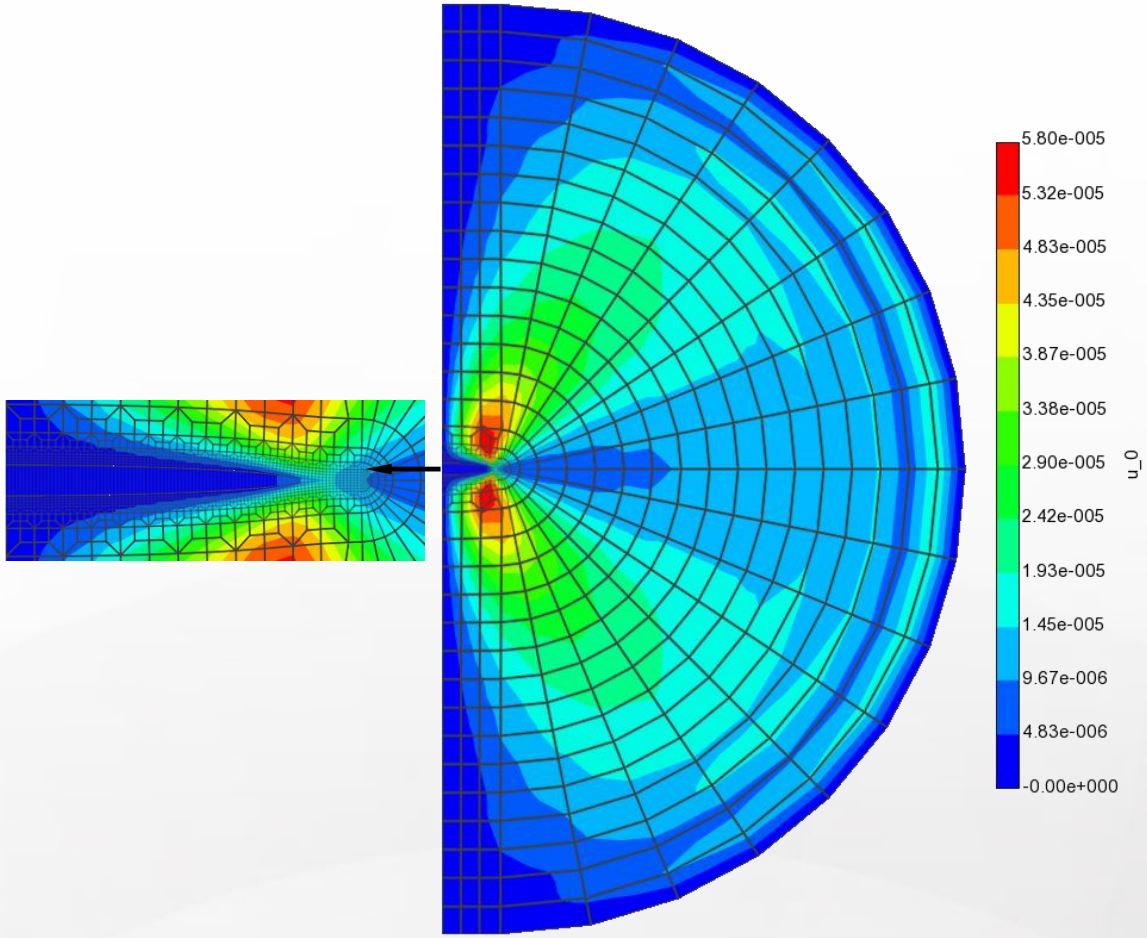
Results at time 30.0 s



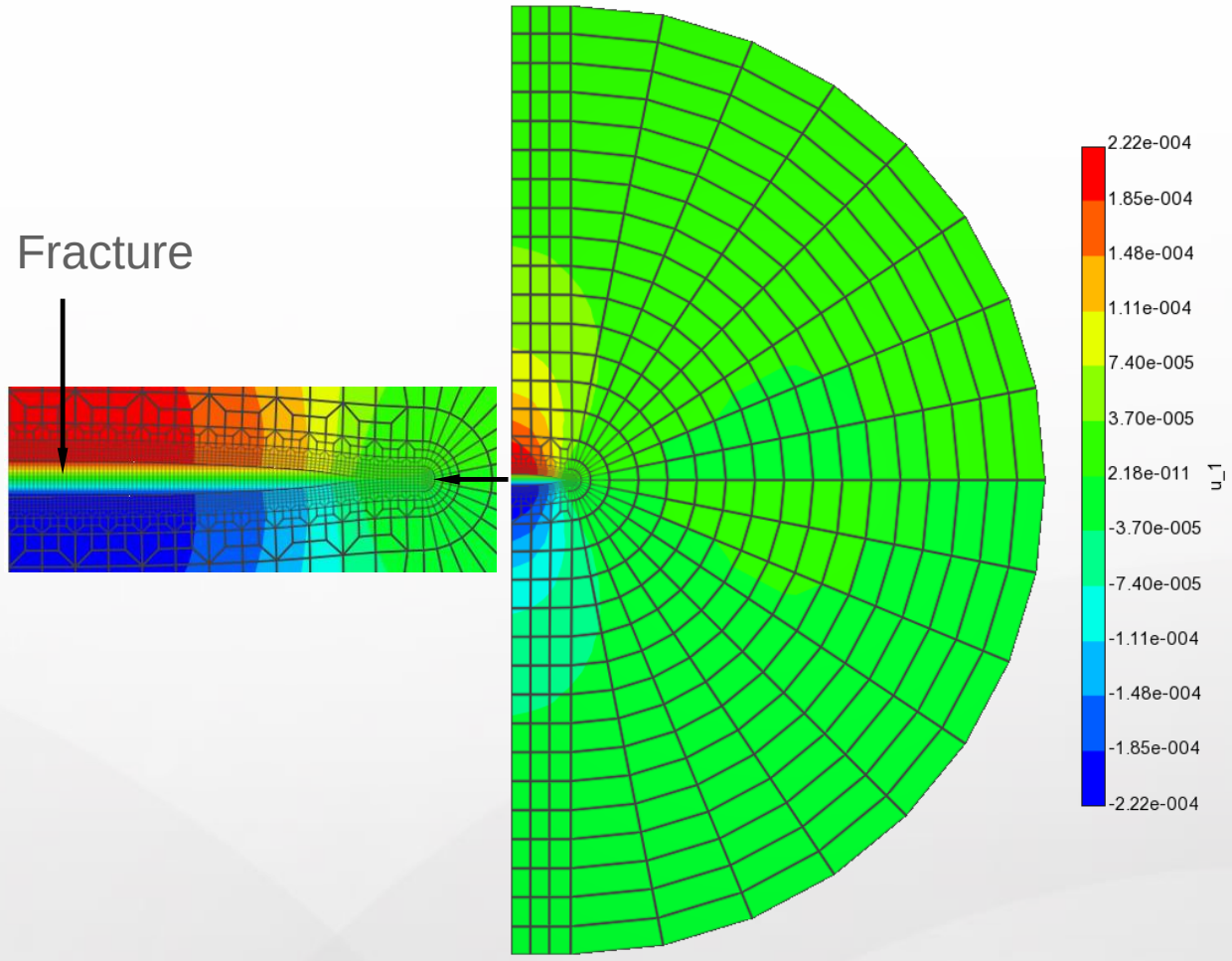
Pore pressure (kPa)



Results at time = 30.0 s



Horizontal displacement (m)



Vertical displacement (m)