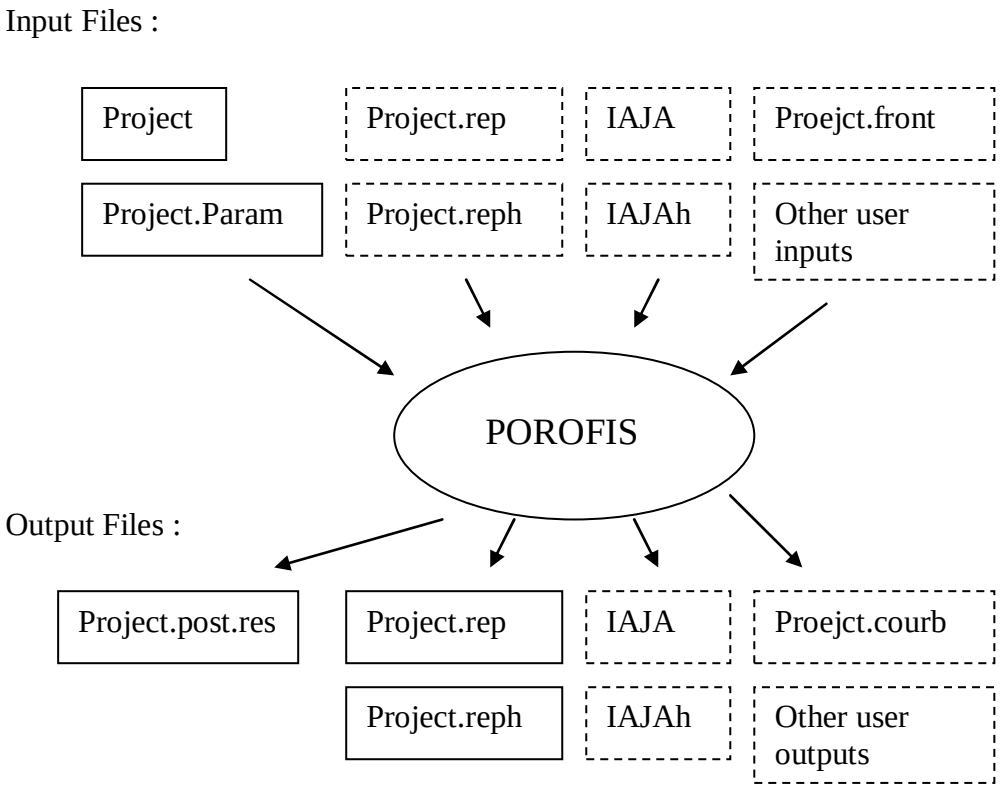
FRACSIMA

Fracture Simulation in Materials
4 Impasse Doron – 94110 – Arcueil – France
www.mecharock.com/Fracsima/

POROFIS

User Notice

I) General organization chart:



The files in ☐ are mandatory, those in ⬚ or optional depending on users' choices.

Input files:

- The file Project is created by GID (template PorofisV3c.bas)
- The file Project.param is written by the user (modification of an existing file)
- Project.rep contains the results of the previous phase of calculation for mechanical calculations. It is needed if the option "Reprise" is chosen (NB(21)=1 in Project.param) in mechanical calculation.
- Project.reph contains the results of the previous phase of calculation for hydraulic calculations. It is needed if the option "Reprise" is chosen (NB(21)=1 in Project.param) in hydraulic calculation.
- IAJA contains the connectivity matrix for mechanical model. It is read if NB(53)=1 in mechanical calculation.
- IAJAh contains the connectivity matrix for hydraulic model. It is read if NB(53)=1 in hydraulic calculation.
- Project.front is an auxiliary file allowing to define a part of the domain's boundary. It is read in the subroutine uecr-courb and is created by GID with the same template that for Project file.

Output Files:

- The main output file Project.post.res is created for post processing by GID
- The file project.courb is created for user defined curves: different variables evolution with time or with load increments.

Mainly, the user is concerned only by the two input files Project and project.param and output files project.post.res and project.courb.

II) The file "Project"

It is created by GId and has the following format:

•The three first lines contain:
Name, NB(12), NB(15)
NB(13), NB(14), var(1), var(2)
NB(1),NB(5),NB(6), NB(7), NB(4), NB(3)

For signification of NB() and var() tables see the corresponding sections.

• The following section contains the "coordinates" of nodes
Node number, x, y

• The following section the connectivity of elements and their material numbers
Element number, nodes numbers (2, 3 or 4 according to the elements type), material number

• The following section contains materials' information. To each material (number of materials = NB(3)) correspond two lines:
Material code, number of parameters
Parameter 1, …, Parameter n

For the materials codes and parameters see the "Materials Catalogue"

• The following section contains Boundary Conditions. There are 9 types of conditions prescribed on the boundary (or internal points in the domain):

1: Displacement Ux,	2: Displacement Uy,	3: Fluid pressure p
4: Normal stress σ_n,	5: Shear stress τ,	6: Fluid normal velocity V_n
7: Punctual force F_x	8: Punctual force F_y	9: Punctual fluid flow Q_n

Note that:
1) The conditions 1, 2, 4, 5, 7, 9 concern the mechanical model, and the 3,6,9 the hydraulic.
1) The conditions 1, 2, 3, 7,8, 9 are applied on points (nodes) whereas 4,5,6 on segments (between two nodes on the boundary)
For each type of the boundary conditions the information is recorded in the file in the following way:

Type of the boundary condition (1 to 9), number of points or segments concerned
Node number, value of the condition (for conditions types 1, 2, 3, 7,8, 9)
Node 1, Node 2, value of the condition (for conditions types 4,5,6)

III) Table NB()

NB(1) : Nombre de noeuds
NB(2) : Nombre de elements
NB(3) : Nombre de materiaux
NB(4) : Nombre de conditions limites
NB(5) : Nombre de elements lineairs (bars)
NB(6) : Nombre de elements triangulaires
NB(7) : Nombre de elements quatratiques
NB(8) : Nombre de elements joints
NB(10) : Nombre de degrés de liberté par neuds problem mecanique
NB(11) : Nombre maximum de points de Gauss / élément
NB(12) : Cas Mecanique (1), Hydraulique (2) , Hydromecanique(3)
NB(13) : Plan (1), Axisymétrique (2)
NB(14) : Déformation planes (1), Contraintes Planes (2)
NB(15) : =1 Evolution instantannee, =2 transitoire
NB(16) : Nombre d'incréments (de charge ou de temps)
NB(17) : Méthode de définition d'incrément (de charge ou de tems) (=1, incréments constants, completer définition pour d'autres types)
NB(18) : =1 Plasticité existe
NB(19) : Nombre d'itération maximum plasticité
NB(20) : =1, on fait de l'homogeneisation
NB(21) : =1, on fait une reprise des déformations et contraintes
NB(22) : =1 , on imprime f0Max a chaque iteration
NB(23) : = 1 , Viscosite existe
NB(24) : = 1 , Deformation libre utilisateur existe
NB(25) : = 1 , Forces volumiques utilisateur existe
NB(26) : Nombre de segments de frontières pour le calcul de bilans de flux
NB(27) : Version etendue ou retsreinte
NB(28) : ndimC Max, dimension des tableaux de contrainte et deformation
NB(29) : Methode de Stockage de la matrice de rigidite (0 :directe, 1 Morse)
NB(30) : ninc, nombre d'inconnus pour le système mecanique
NB(31) : Compt, dimension de matrice globale en stockage morse
NB(32) : Comptr, dimension de matrice reduite en stockage morse
NB(33) : =1, Non linearite elastique existe
NB(34) : Nombre d'itération maximum pour resolution methode gradient conjuque
NB(35), NB(36) le n° de nœud et d'élément pour lesquels les déplacements et conraintes seront écrit par uecr_courb
NB(37) : Nombre de nœuds initiaux avant dedoublement
NB(38) : Nombre de degres de libertes total réduit compte tenu des contraintes d'égalité imposés pour le problem hydraulique
NB(39) : Chargement en homogeneisation
NB(40) : Direction du chargement en homogeneisation 1-6 (Exx,Eyy,2Exy,Sxx,Syy,Sxy)
NB(41) : NOLINHK: Matrice de rigidite lineaire (0) ou non lineaire (1)
NB(42) : NOLINHM: Matrice de masse lineaire (0) ou non lineaire (1)
NB(43) : Niter, au bout de Niter pas de temps on affiche le résultat
NB(44) : Nombre de degres de libertes total réduit compte tenu des contraintes d'égalité imposés pour le problem mécanique

NB(45) : Nombre de degrés de liberté par neuds pour le problem hydraulique
NB(46) : =1, toutes les fractures en hydraulique de perméabilite transversale infinie (donc contrainte d'égalité des pressions et nombre de nœuds réduits)
NB(47) : Nombre de variables internes par éléments (Min 1, Max 10)
NB(50) : 1: Extraction of Excavation boundary forces
NB(51) : On applique les forces surfaciques représentant l'action d'excavation
NB(52) : Si different de zero, numero k de la variable interne Vin(n,k) représentant la force surfacique (de pression ou du produit bp, b Biot, p Pression) à appliquer aux parois des joints.
NB(53) : =1, on lit une matrice de stockage Morse existante
NB(57) : Si différent de zéro, numéro k de la variable interne Vin(n,k) représentant la contrainte de cisaillement (écoulement de fluide visqueux) à appliquer aux parois des joints.
NB(58) : Nombre de variables internes hydrauliques par éléments (Min 1, Max 10)

IV) Table var()

var(1) : value of the generalized stress or strain according to the NB(14)
var(2) : position d'axe de symmetrie
var(3) : espilon pour la convergence en plasticité
var(4) : force volumique Fx (due à l'accélération sismique) si NB(25)=1
var(5) : force volumique Fy (verticale, poids volumique) si NB(25)=1
var(6) : espilon pour convergence resolution methode gradient conjuque
var(7) : espilon pour la convergence en elasticite non lineaire
var(8) : Epsilon pour factorisation LU methode directe, module resolution
var(9) : Lambda pour la methode de résolution Euler mixte, Lambda = 0, explicite, Lambda = 1, 0<Lambda<1, methode mixte
var(10) : Distance minimum entre les points de maillage non dedoubles
var(11) : temps de départ calcul transitoire
var(12) : temps final calcul transitoire
var(13) : deltat calcul transitoire
var(14) : pris (taux de chargement maximum (comme taux de déconfinement))
var(15) : valeur de la charge pour chargement en mode homogeneisation

V) Modules Hydmec and Utilisateur

The Porofis code has some fixed parts non accessible to the user, and two modules hydmec.f90 and utilisateur.f90 which can be modified by the user.

- The module hydmec manages the general calculation process by calling to two other modules for hydraulic and mechanical calculations. It allows sequential calls to these two modules in order to establish couplings between hydraulic and mechanical phenomena.
- The module utilisateur contains a set of subroutines allowing the user to develop its own constitutive models or define user defined inputs or outputs.

V.1) Module Hydmec

The hydromechanical coupled modeling is managed in the subroutine hydromec in the following way:

! Initialisation ---- Hydraulic ----------
Here the hydraulic variabales are read and arrays are initialized (subroutine inithyd).
The subroutine FBCndh reads and initializes boundary conditions for hydraulic calculation

! Initialisation ---- Mechanics ----------
Here the mechanical variabales are read and arrays are initialized (subroutine initmec).
The subroutine FBCnd reads and initializes boundary conditions for mechanical calculation.
The subroutine Forv reads and initializes (if existing) volume forces.
The subroutine deflib reads and initializes (if existing) the free strains ($\boldsymbol{\varepsilon}^L$).

! ---------- Initialisation Increments de charge --------
Here the load increment *dch* for steady state calculations, or the time increment *deltat* for time depending calculations is defined.
For time independent calculation, the total load defined by boundary conditions corresponds to load level *vchar* =1, and *dch* represents the variation of *vchar*.
For time depending calculation, the time variable *temps* varies between an initial value and the final value *TempsF* with increments *deltat*.

! ===== Debut des increments de charge ==========
The incremental calculation starts here and finishes at:
5010 Continue; ! Fin des increments de charge
At each time or load increment:
The subroutine *Increhyd* calculates the new hydraulic variables at the end of the increment
The subroutine *Incremec* calculates the new mechanical variables at the end of the increment

Between these to calculations, in the section one can update the mechanical parameters function of the new hydraulic variables **(coupling from hydraulic to mechanics)**:

! Deb Passage des variables Hydro->Mecanique
! .... here one can change for instance the free strain function of the fluid pressure for poroelastic calculation (see theoretical papers). The pressure in the output *RP* of *increhyd* and the free strain, the input *EpL* of *incremec*.
! Fin Passage des variables Hydro->Mecanique

After mechanical increment, call to the subroutine *uecr_courbe* allows putting the results for this increment in a file to be displayed as plots of evolution function of time or to the load level.

A test on the value of the *vchar* or the *Temps* indicates if the calculation loop is finished or must continue for the next increment.

If the calculation has to continue, before passing the next increment, the hydraulic parameters are updated function of the mechanical results **(coupling from mechanics to hydraulic)**:
! Deb Passage des variables Mecanique-> Hydro

Here the hydraulic parameters are updated function of mechanical parameters before going to the next increment.
Very often, the change of hydraulic parameters implies the change of permeability and so the necessity of updating the permeability matrices and reassembling the rigidity matrix for hydraulic calculation. This down in the section:
! Hydraulique non lineare ++++++++++++++++++++++++++++++++++++++

After this has been down on return back to the address 300 which starts the new increment's calculation.

After the end of the calculation loop is reached, the final calculation results are put in corresponding file.
CalvitH calculates the average fluid velocity in the domain (for homogenization studies).
Ecr_reprise writes the mechanical results in the file Project.rep for a possible reprise in future phases of calculation.
ec_reprise_hyd writes the hydraulic results in the file Project.reph for a possible reprise in future phases of calculation
uecr_courbe completes the file *Project.courb* and closes it (flag 3)
ecrit_gid_mec writes the mechanical results in the file Project.post.res for post processing by GID
uecr_gid_hydmec writes the mechanical+hydraulic results in the file Project.post.res for post processing by GID

VI) Errors Analysis

• The error "File project.front does not exist" is generated in the Subroutine uecr_courbe in Utilisateur module: the line " call lit_front(konb)" calls the subroutine to read the file project.front. Or you have to created this file or to delete the line " call lit_front(konb)" .

The file project.front has the same format that the file project, and is created by the template "PorofisV3c.bas" from GID but with the name project.front. The part of the boundary to which is assigned the condition "Contrainte Normale" defines the set of segments read in the file and put in the array konb(), for different uses.

VII) En chantier :

ntyp(n)
ntyp =1 : N'existe pas
ntyp = 2 : Element bar
ntyp = 3 : triangle à 3 nœuds
ntyp = 4 : quadrilatère à quatre noeuds
ntyp =5 : élément joint

NB(25) ne prend que des valeurs de 0 (pas de force volumique)
ou 1 (exite force volulique). Dnas ce dernier cas, les forces volumique Fx et Fy sont

données par var(4) et var(5) (Fichier param).
Mais toutes ces forces concernent le calcul mécanique seul et non le calcul hydraulique.
La prise en compte des forces de pesanteur dans le calcul hydraulique ne passe pas pas NB(25) et var(5), mais doit se faire directement en considérant que que le tableau RP représente la charge hydraulique et il faut lui enlever ou ajouter l'effet de la pesanteur pour obtenir la pression.

VIII) Cas study Spalling : Modeling flow around a wellbore

At time 0, the wall normal stress and the fluid pressure are equal to 10 MPa which is equal to the initial pore pressure in the formation. The in situ stress is 20 MPa. The the fluid pressure suddenly drops to 6 MPa (well production phase).

Step 0:
Creating the initial state before well fluid pressure drop:
The file Spalling0 contains boundary conditions for

Infinite stress -20MPa, wall stress -10 MPa
Infinite pressure 10 MPa wall pressure 10 MPa
Evolution mode : **Steady state**

The file Spalling0.param : NB(16)= 100 (number of increments), NB(52)=2 (fluid pressure in fractures taken into account), NB(21)=0 (non reprise)

Outputs : spalling0.rep, spalling0.reph, spalling0.post.rep

At time 0^+, the wall normal stress and the fluid pressure drop to 6 MPa. This must be done on two steps one concerning the normal stress and the other, the pressure.

Note : the files used and created by Porofis has not the number spalling1, spalling2, spalling1.rep etc. The user copies these files with these names to save them for further use.

Step 1:
Decreasing the normal stress to -6 MPa on the wall, without pressure change (time 0^+). This is down by introducing new loads which are superposed *for mechanics only* (function Reprise=1) to the preceding ones.

The file Spalling1 contains boundary conditions for

Infinite stress 0MPa, wall stress +4 MPa
Infinite pressure 10 MPa wall pressure 10 MPa
Evolution mode : **Steady state**

The file Spalling0.param : (NB(16) =1000), NB(52)=2 (fluid pressure in fractures taken into account), NB(21)=1 (reprise).

The input files spalling.rep and spalling.reph are equal to spalling0.rep, spalling0.reph
Outputs : spalling1.rep, spalling1.reph, spalling1.post.rep

Step 2:

Decreasing the wall pressure to 6 MPa without changing the stress (time 0^{++}). This is down by in transient mode in one increment time in order to not allow the pressure diffusion in the matrix and have time to only change the boundary condition: deltat = final time.
In transient mode, boundary conditions are considered as rates. So, if with the function Reprise=1, zero value boundary conditions are applied, this means that the boundary values for the preceding step will be maintained constant.

The file Spalling2 contains boundary conditions for

Infinite stress 0MPa, wall stress 0 MPa
Infinite pressure 0 MPa wall pressure -4 MPa
Evolution mode : **Transient**

(Put the last parametre in the first line of the file Project equal to 2 (transient) instead of 1 (steady state).

The file Spalling2.param : (NB(16) non effect because transient evolution), NB(52)=0 (we want to just modify the boundary condition, the effect of fluid pressure in fractures will be taken into account in the following stage of transient evolution modeling), NB(21)=1 (reprise), var(12)=var(13)=1 (to have one only increment of deltat=1).
The input files spalling.rep and spalling.reph are equal to spalling1.rep, spalling1.reph
Outputs : spalling2.rep, spalling2.reph, spalling2.post.rep

Step 3:
Modelling the transient evolution due to fluid flow.
Now all the boundary conditions and initial state are correctly in place.
The transient flow and its effect on the mechanical fields is modeled in the time interval [0^{++}, TempsF] with time increments delatat.

All the rates on the boundary (pressure, stresses) are put equal to zero to maintain constant the boundary conditions.

The file Spalling3 contains boundary conditions for

Infinite stress 0MPa, wall stress 0 MPa
Infinite pressure 0 MPa wall pressure 0 MPa
Evolution mode : **Transient**

(Put the last parametre in the first line of the file Project equal to 2 (transient) instead of 1 (steady state).

The file Spalling2.param : (NB(16) non effect because transient evolution), NB(52)=0 (we want to just modify the boundary condition, the effect of fluid pressure in fractures will be taken into account in the following stage of transient evolution modeling), NB(21)=1 (reprise).
The input files spalling.rep and spalling.reph are equal to spalling2.rep, spalling2.reph
Outputs : spalling3.rep, spalling3.reph, spalling3.post.rep, spalling3.ucourb (the first time file courb can be useful).

Note: The objective of the steps 0 to 2 is to create the initial stress and pressure filed and for this reason a linear model for the fractures and the matrix can be used for simplicity. Non linear materials law can be introduced at the step 3.

Also, at the stage 3, it is interesting to study two cases: with and without taking into account the shear forces due to the fluid veiscosity in fractures.

*==========

Volume forces: NB(25)=1, and Var(4), Var(5)=…) works for standard materials, i.e, existing in the catalogue. For user defined materials, one has to put NB(25)=2, and then define the volumetric force directly in the subroutine uForvol. The array Fv(*n*,*i*) represents the unit force in direction i (i=1,2 for x,y) for the element *n*. This unit force specified by the user will be multiplied by the surface of the element *n* in the code to obtain the total force on that element.

Warning: as every other load, the value introduced for Fv is considered as a rate in transient evolution mode. So, one has to introduce this volume force in a first step in the same way that for boundary conditions, and then put it equal to zero to maintain it constant for the remaining transition evolution phases.