Annex A: Matlab code QUICK scheme

Main

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Finite Volume method, QUICK scheme : Discretisation nz volumes, nz+2 nodes%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%% The goal of this program is to provide the evolution different parameters
%% in a syringe (void ratio, stress, permeability, filtration velocity) depending
%% on the ram position and speed.
%%
%% For this, the geometrical, physical and initial parameters of the problem are
%% given to initiate the computation. The principle of the program is to solve
%% the equation of advection-diffusion of void ratio taken as variable using
%% a QUICK scheme. Here we consider diffusion to be constant, and convection is
%% driven by the pore pressure gradient on an element.
%%
%% The velocity of filtration, derived from Darcy's law depends on the void ratio
%% (change in permeability, and stress). Meaning that this advection velocity is
%% not constant over the domain.
%%
%% Each time step of computation leads to the extrusion of a slice of cement having
%% a void ratio content, each temporal steps leads to a movement of the ram represented
%% by the replacement of the values for ram position j-1 by 0. (No flow, no permeability)
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;
format long;

 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Initial configuration parameters %%%
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

e0=0.98 ;                                    % Initial void ratio
emin=0.36 ;                                  % Minimal void ratio (close packing of sphere 0.64)

phid0=1/(1+e0) ;                             % Initial solid volume fraction (corresponding to e0)
```

```
phidmax=0.64 ;                               % Maximal solid volum fraction (corresponding to emin)

phimax=22;                                   % Maximum friction angle

a=22.18;                                     % Fitting parameters describing yield stress
b=-2.64;                                     % Fitting parameters describing yield stress

 %%%%%%%%%%%%%%%%%%%%%%%%%
%%% Material parameters %%%
 %%%%%%%%%%%%%%%%%%%%%%%%%

Cc = 0.34;                                   % Compressibility coefficient
Ck = 1.2;                                    % Taylor fitting parameter
K0 = 3.2e-9;                                 % Initial permeability for e=e0 (m.s-1)
rhow = 2500;                                    % Specific gravity of liquid phase
g = 9.8;                                     % Acceleration due to gravity
m = 0.65;


 %%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Geometrical parameters %%%
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%

R2 = 0.01;                                  % Diameter of the exit die orifice (m)
R1 = 0.04;                                  % Diameter of the extruder barrel (m)
L1 = 0.1;                                    % Lenght of the barrel (m)
L2 = 0.03 ;                                  % Conical height (dead zone in paper) (m)
S = pi*R1*R1;

 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Computational parameters %%%
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

coef = 10000;                                 % Coefficient created to define number of discretized elements
nz = coef*L1 ;                               % Number of spatially discretized elements.
nt = nz ;                                    % Number of temporally discretized elements
zmax = L1 ;                                  % Position z maximum corresponding to the length of plug zone.
Dz = L1/nz ;                                 % Thickness of a discretized volume.
```

```
z=[Dz/2:Dz:L1-Dz/2] ;                          % Axe des x discrétisé

Ttot = 400 ;                                   % Total time of the extrusion process
Dt = Ttot/nt ;                                 % Value of delta t.

 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Starts of the algorithm solving the flow at at each iteration %%%
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Fistore=ones(nz,nt);                          % Allocation of flux matrix size
Tn=Dt/(Dz*rhow);                               % Definition of Tn
Store=ones(nz,nt);                             % Allocation of store matrix size permiting the update of void
ratio after each steps

%e=ones(nz,nt);                                % Allocation matrix size

for y=1:nz                                     % allocation of void ratio matrix size
e(y,1)=e0;                                     % Problem initialization, homogeneized void ratio over syringe
length
end

sigmaz(1,2) = 37;                               % Initially applied stress t=0 position L1
F(1)=sigmaz(1,1);                               % Initiation of the appplied stress for compaction law
%sigmaz=ones(nz,nt);                            % Allocation of stress matrix size
%U=ones(nz,nt);                                 % Allocation of velocity matrix size
%Voidstress=ones(1:nt);
%K=ones(nz,nt);
Storefi(:,1)=ones(nz,1);


for j=2:1:(nt-3);                                % Temporal loop for each time we compute the stress, void,
permeability distribution over the syringe length

    MatQuadriQUICK

    Fistore([1:1:(nz+2-j)],j)= A\(Tn*Storefi([1:1:(nz+2-j)],j-1));     % Flux matrix calculation, based on
previous flux state and A matrix
```

```
 for x=1:1:(nt+2-j)

    Store(x,j) = e(x,j-1)+ (Fistore(x,j)*S);           % Calculation of the "virtual" void ratio at the end
of step j

    Storefi(x,j)=Fistore(x,j);

    Exit(j)=Store(nz,j);                               % Storage of the void ratio quantity of the extruded
slice at t=j*Dt

    e(x,j)=Store(x,j);                                 % Value of the new e distribution vector at steps j+1
= value at the end of the process at time j displaced to i+1

 end

end
```

MatQuadriQUICK

```
format long;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 %%% Definition of the stress distribution in the syringe as a function of the void ratio%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%% Here we compute the entire stress profile along the plug zone using a simple
%% relation deriving the stress at position z+dz from the one acting at position z
%% taking in to account the frictionnal plastic stress.
%%
%%
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



    for i=1:1:(nz+2-j)

          %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
          %%%
          %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

          phid(i,j)=1/(1+e(i,j-1)) ;                                   % Solid volume fraction at
position i in the syringe and time j
          tau0(i,j)=a*e(i,j-1)^b;                                      % Yield stress at position i in
the syringe and time j (depends on the void ratio e(i,j)
          phi(i,j)=phimax*((emin-e(i,j-1))/(emin-e0));                 % Friction angle at position i in
the syringe and time j (depends on the void ratio e(i,j)
          K(i,j)=(10^((e(i,j-1)-e0)/Ck))+K0;                           % Evolution of permeability at
position i in the syringe and time j(depends on the void ratio e(i,j)

          %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
          %%%
          %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

          sigmaz(i+1,j)=sigmaz(i,j)-((4*Dz/m*R1)*(tau0(i,j)+tan(phi(i,j))*(1-sin(phi(i,j)))*sigmaz(i,j))); %
Iteration process giving stress distribution in the plug zone
```

```
    end

          sigmaz(nz+1,j)=sigmaz(nz,j); %

    for y=1:1:(nz+2-j)

         DSigm(y)=(sigmaz(y,j)-sigmaz(y+1,j))/Dz;   % Pore pressure differenciation

    end

          sigmaz(1,j+1)=sum(DSigm(y));
          F(j) = sum(DSigm(y));                                        % Calculation of the total force
acting on the ram at each time steps ( used in the calculation of Voidstress)

          Voidstress(j)= max(e0-Cc*log(F(j)),0.44);                    % Calculation of Voidstress as a
function of the applied stress to the ram
          K(nz+1,j)=K(nz,j);
          %e(nz+1,j)=e(nz,j);

          %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
          %%%
          %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

          for o=2:1:(nz+1-j)

              U(o,j)=(1/S)*(K(o,j)/rhow*g)*((sigmaz(o+1,j)*(1-(e0-e(o+1,j-1))/(e0-Voidstress(j))))-
(2*sigmaz(o,j)*(1-(e0-e(o,j-1))/(e0-Voidstress(j))))+(sigmaz(o-1,j)*(1-(e0-e(o-1,j-1))/(e0-
Voidstress(j)))))/((Dz)^2));

          end

          U(1,j)=(1/S)*(K(1,j)/rhow*g)*(((sigmaz(2,j)*(1-(e0-e(2,j-1))/(e0-Voidstress(j))))-
(2*sigmaz(1,j)*(1-(e0-e(1,j-1))/(e0-Voidstress(j))))))/((Dz)^2));
          U(nz+2-j,j)=(1/S)*(K(nz,j)/rhow*g)*((2*sigmaz(nz,j)*(1-(e0-e(nz+2-j,j-1))/(e0-
Voidstress(j))))+(sigmaz(nz-1)*(1-(e0-e(nz+1-j,j-1))/(e0-Voidstress(j)))))/((Dz)^2));;


          U(nz+1,j)=U(nz,j);                                           %
          %U(nz+2,j)=U(nz+1,j);                                            %


%size(sigmaz)
%size(K)
%size(U)



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  %%%%% Construction of the quadridiagonal matrix solving the flux problem%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%% The QUICK scheme is the oldest upper order numerical methods, it has been introduced
%% by Leonard in 1979.
%%
%% The QUICK scheme use a quadratic interpolation on three weighted points and is solved
%% implicitly. It is based on the use of finite differences on first and second order
%% and taylor developpment of the void flux function
%%
%% The description of the flux is given by the construction of a quadridiagonal
%% matrix
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                                                                           % Allocation of memory for matrix
A
De = 0.3 ;                                                                 % Diffusion coefficient east
Dw = 0.3 ;                                                                 % Diffusion coefficient west
Da = 0.3 ;                                                                 % Diffusion coefficient for
mirror condition ( Dirichlet condition )
Ua = 0.003 ;                                                               % Mirror conditon for node 1
(Dirichlet condition)                                                               %

apo = ( Dz / Dt ) * rhow ;                                                 %


for d=3:1:(nz+2-j)

    V1(1,j)=   (7/8) * U(2,j) + De + 3 * Da + apo ;
    V1(2,j)=(6/8) * U(3,j) - (3/8) * U(1,j) + De + Dw+ apo ;
    V1(d,j) = (6/8) * U(d+1,j) + (-3/8) * U(d-1,j) + De + Dw + apo;          % Definition of the vector V1
defining the principal diagonal of the quadridiagonal matrix
    V1(nz+2-j,j)= -(3/8) * U((nz+2-j),j) + Dw + (9/8)*U((nz+3-j),j) + apo;

end

for f=3:1:(nz+1-j)

    V2(1,j)=(3/8) * U(2,j) - De ;
    V2(2,j)= -(3/8) * U(3,j) - De ;
    V2(f,j)= (6/8) * U(f+1,j) -De;                                          % Definition of the vector V2
defining the +1 (east) of the quadridiagonal matrix

end

for h=3:1:(nz+1-j)

    V3(1,j) = -Ua - (8/3) * Da ;
    V3(2,j) = (1/8) * (U(3,j) - 7 * U(2,j)) - Dw ;
    V3(h,j)= (1/8) * (-U(h+1,j)-6*U(h-1,j)) - Dw;                           % Definition of the vector V3
defining the -1 (west) diagonal of the quadridiagonal matrix
    V3(nz+1-j,j) = -(6/8) * U((nz+1-j),j) - Dw - (1/8)*U((nz+2-j),j) ;

end

for l=3:1:(nz-j)

    V4(1,j) = 0 ;                                                          % Modidied value of aww taking
into account BC first slice (in contact with ram)
    V4(2,j) = 0 ;                                                            % Modidied value of aww taking
into account BC second slice
    V4(l,j)= U(l-1,j)/8;                                                    % Definition of the vector V4
defining the -2  (west west) diagonal of the quadridiagonal matrix
    V4(nz-j,j) = U((nz-j),j)/8 ;                                               % Modidied value of aww
taking into account z=0 BC  i=nz last slice before extrusio

end
```

```
size(V1);
size(V2);
size(V3);
size(V4);


A = diag(V1([1:1:(nz+2-j)],j),0)+ diag(V2([1:1:(nz+1-j)],j),1) + diag(V3([1:1:(nz+1-j)],j),-1)+
diag(V4([1:1:(nz-j)],j),-2);                     % Building of A matrix


A;
```