APPENDIX A

MATLAB Script And Function Files

```
% COLUMN FLOTATION MODEL

% Initialization routine:

% - Variables initial values
% - Matrices dimensions

% -------------------------------------------------------------------------------

% Constants

% G: gravity acceleration
% Ul: water viscosity
% SGw: water density

global G Ul SGw;
G=980;
Ul=0.01;
SGw=1;

% -----------------------------------------------------------------------

% Input Parameters:

% Nb:        number of bubble size classes
% Np:        number of particle size classes
% Nc:        number of particle composition classes
% Tfinal:    time simulation stops
% Ntimes:    number of time steps taken in simulation
% dt:        time interval used = total time/number of steps
% nZlp:      number of zones in lower part of pulp
% nZup:      number of zones in upper part of pulp
% nZsf:      number of zones in stabilized froth
% nZdf:      number of zones in draining froth
% A:         column cross section
% Cd:        column diameter
% L:         total column length
% Lp:        length column below interface
% Lf:        length column above interface
% FP:        distance of the feed port measured from column top
% Lt:        length of the transition zones
% Lww:       length of column section above wash water addition point
% nZ:        total number of zones in column
% Vz:        zone volume
% allVz:     vector of all zones volumes
% alllg:     vector of all zone heights

global Nb Nc Np Cd A dt Nz
Nb=4;
Np=2;
Nc=3;
dt=0.08;
Cd=5;
A=pi*Cd*Cd/4;
nZlp=6;
```

```
nZup=3;
nZsf=6;
nZdf=2;
allnZ=[1;nZlp;1;nZup;1;nZsf;1;nZdf];
Nz=sum(allnZ);
Lt=2;
L=200; Lp=150; Lf=L-Lp; Lww=10; Lg=10; FP=70;
Lzdf=(Lww-0.5*Lt)/nZdf;
Lzsf=(Lf-Lww-0.5*Lt)/nZsf;
Lzup=(FP-Lf-0.5*Lt)/nZup;
Lzlp=(L-FP-0.5*Lt)/nZlp;
allVz=zeros(1,Nz);
alllg=zeros(1,Nz);


% ============================================================================

% Operating Conditions:

% Qg:         volume of air per unit time entering the column
% Vg:         superficial velocity of air entering the column
% Qw:         wash water volumetric rate
% Qfeed:      slurry volume per unit time entering the column at feed port
% Mfeed:      solids mass per unit time entering the column at feed port
% Db:         vector of bubble sizes
% Dp:         vector of particle sizes
% SGp:        vector of particle densities for each composition type
% SGb:        vector of bubble densities for each bubble size class at time t
%             in any zone
% SGsl:       slurry density at time t in any zone
% Sgsusp:     density of the three-phase suspension at time t in any zone
% Blk:        vector of bubble loading fractions for each bubble size class at
%             time t in any zone
% Blsck:      matrix of bubble loading fractions corresponding to each particle
%             size and composition, for each bubble size class, at time t in
%             any zone
% maxBlk:     maximum fractional bubble coverage
% Mfeedsc:    matrix of mass rates for each particle size and composition
%             entering the column in the feed slurry
% Qt:         tailings rate
% Qb:         bias rate
% Qp:         product liquid rate
% Qsl:        slurry volumetric rate in any zone
% Vsl:        slurry superficial velocity in any zone
% Usl:        slurry viscosity for any zone at time t
% kasck:      matrix of flotation rate constants between each bubble size class
%             and each particle size and composition classes
% fnd:        vector of discrete number distribution for each bubble size class
%             entering the aeration zone
% fvd:        vector of discrete volume distribution for each bubble size class
%             entering the aeration zone
% Cg:         proportionality constant in empirical relation between gas
%             velocity and generated bubble size
% Pcollis:    matrix of size (Nb*Np,Nc) whose elements are the probabilities of
%             collision between bubbles in each size class k and particles of
%             type i,j.
% Pattach:    matrix of size (Nb*Np,Nc) whose elements are the probabilities of
%             attachment between bubbles in each size class k and particles of
%             type i,j.
% Pcsck:      matrix of the probabilities of collection of particles in class
%             (i,j) by bubbles in size class k
% Feedsols:   feed mass percent solids
% Feeddens:   feed slurry specific gravity
% Tcfeed:     total concentration of solids in the feed (mass of solids per
```

```
%             unit of feed volumetric flow)
% feedsizd:  discrete size distribution of feed particles
% feedfloatd:       discrete composition distribution of the feed particles
% Cfeed:     matrix of the concentration of feed solids belonging to each pair
%             of size and composition classes.
% sflambda:  matrix of coalescence efficiency rate parameters for the
%             stabilized froth region
% wwlambda:  matrix of coalescence efficiency rate parameters for the wash
%             water addition zone
% dflambda:  matrix of coalescence efficiency rate parameters for the draining
%             froth region


global Vg Qg Qw Qfeed Qt Qb Qp Cfeed Db Dp SGp Vsl kasck fvd maxBlk

Vg=2.0; Qg=Vg*A;
Vw=0.3; Qw=Vw*A;
Vfeed=0.4; Qfeed=Vfeed*A;
Qt=(0.4/Vfeed)*Qfeed;
Qb=Qt-Qfeed;
Qp=Qw-Qb;

Cg=0.0947;
Dbaven=Cg*Vg^(1/4);
fnd=[0.5470;0.4525;0.0003;0.0002];

Db=zeros(Nb,1);
denom=0;
for i=1:Nb
denom=fnd(i)*2^((i-1)/2)+denom;
end
for i=1:Nb
Db(i)=(Dbaven*2^((i-1)/2))/denom;
end

fvd=(fnd.*(Db.^3))./sum(fnd.*(Db.^3))

Dp=[0.0056 0.008]';
SGp=[1.2 2.0 3.0];
SGb=zeros(Nb,1);

n=2; Re=2; Pcollis=zeros(Np*Nb,Nc); Pattach=zeros(Np*Nb,Nc);
for i=1:Np
      for j=1:Nc
            for k=1:Nb
                  Pcollis(Np*(k-1)+i,j)=(1.5+(4*Re^0.72)/15)*
                  (Dp(i)./Db(k))^n;
            end
      end
end
Pattach=     [1.0 0.6 0;0.9 0.54 0;0.9 0.54 0;0.8 0.48 0;0.8 0.48 0;
             0.7 0.42 0;0.7 0.42 0;0.6 0.36 0];
Pcsck=Pcollis.*Pattach;

Feedsols=20;
Feeddens=1.8;
TCfeed=Feedsols*Feeddens/100;
feedsizd=[0.7;0.3];
feedfloatd=[0.6;0.2;0.2];
Cfeed=zeros(Np,Nc);
for i=1:Np
 for j=1:Nc
      Cfeed(i,j)=TCfeed*feedsizd(i).*feedfloatd(j);
 end
end

sflambda=zeros(Nb,Nb);
```

```
Ks=2e-4;
for i=1:Nb
 for j=1:Nb-i
      sflambda(i,j)=Ks*(Db(i)+Db(j))^(-2)
 end
end

wwlambda=zeros(Nb,Nb);
Kw=4e-2;
for i=1:Nb
 for j=1:Nb-i
      wwlambda(i,j)=Kw*exp(-((Db(i)+Db(j))^(2)))
 end
end

dflambda=zeros(Nb,Nb);
Kd=3e-2;
for i=1:Nb
 for j=1:Nb-i
      dflambda(i,j)=Kd*exp(-((Db(i)+Db(j))^(2)))
 end
end

kasck=zeros(Nb*Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
      kasck(Np*(k-1)+i,j)=1.5*Vg*Pcsck(Np*(k-1)+i,j)/Db(k)
    end
  end
end
maxBlk=0.5;

% ==========================================================================================

% Other Variables Definitions:

% Eik:       vector of air fraction values for each bubble size class at time
%            t-1 in any zone
% Ek:        vector of air fraction values for each bubble size class
%            calculated for time step t in any zone
% dEk:       vector of the changes in air fraction values for each bubble size
%            class in any zone
% Vgs:       vector of slip velocities for each bubble size class at time t
% Vgspz:     vector of slip velocities for each bubble size class at time t in
%            the(i-1)-th zone below
% Vp:        vector of settling velocities for the ith zone corresponding to
%            each particle size and composition classes at time t
% Vpnz:      vector of settling velocities for the i+1-th zone corresponding
%            to each particle size and composition classes at time t
% EnZ:       vector of air fraction values for the n-th bubble size class for
%            all zones at time t
% EnZT:      matrix of air fraction values for the n-th bubble size class for
%            all zones at time t
% EZ:        vector of total air fractions for all zones at time t
% EZT:       matrix of total air fractions from t=0...tfinal for all zones
% Mfisc:     matrix of masses of free particles for each particle size and
%            composition classes at time t-1 in the i-th zone
% Cfisc:     matrix of mass concentrations of free particles for each particle
%            size and composition classes at time t-1 in the i-th zone
% Mfsc:      matrix of masses of free particles for each particle size and
%            composition classes at time t in the i-th zone
% Cfsc:      matrix of mass concentrations of free particles for each particle
%            size and composition classes at time t in the i-th zone
% Mfscnt:    matrix of masses of free particles for each particle size and
%            composition classes at time t in all zones
```

```
% Cfscnt:    matrix of mass concentrations of free particles for each particle
%            size and composition classes at time t in all zones
% Mfiscpz:   matrix of masses of free particles for each particle size and
%            composition classes at time t-1 in the (i-1)-th zone
% Cfiscpz:   matrix of mass concentrations of free particles for each particle
%            size and composition classes at time t-1 in the (i-1)-th zone
% Mfiscnz:   matrix of masses of free particles for each particle size and
%            composition classes at time t-1 in the (i+1)-th zone
% Cfiscnz:   matrix of mass concentrations of free particles for each particle
%            size and composition classes at time t-1 in the (i+1)-th zone
% dMfsc:     matrix of the changes in the masses of free particles for each
%            particle size and composition classes at time t in any zone
% dCfsc:     matrix of the changes in the mass concentrations of free
%            particles for each particle size and composition classes at time
%            t in any zone
% MfncZ:     vector of masses of free particles belonging to the n-th particle
%            size class, regardless of composition, at time t in all zones
% MfncZT:    matrix of masses of free particles belonging to the n-th particle
%            size class, regardless of composition, from t=0...tfinal in all
%            zones
% MfsnZ:     vector of masses of free particles belonging to the n-th
%            composition class, regardless of particle size, at time t in all
%            zones
% MfsnZT:    matrix of masses of free particles belonging to the n-th
%            composition class, regardless of particle size, from t=0...tfinal
%            in all zones
% MfnmZ:     matrix of masses of free particles in the n-th particle size
%            class and the m-th composition class at time t for all zones
% MfZ:       vector of the total mass of free particles at time t in all zones
% MfZT:      matrix of of the total mass of free particles from t=0 to
%            t=tfinal in all zones
% CfZT:      matrix of of the total mass concentration of free particles from
%            t=0 to t=tfinal in all zones
% Maisck:    matrix of masses of particles attached to bubbles in size class
%            k, for each particle size and composition classes, at time t-1 in
%            the i-th zone
% Caisck:    matrix of mass concentrations of particles attached to bubbles in
%            size class k, for each particle size and composition classes, at
%            time t-1 in the i-th zone
% Masck:     matrix of masses of particles attached to bubbles in size class
%            k, for each particle size and composition classes, at time t in
%            the i-th zone
% Casck:     matrix of mass concentrations of particles attached to bubbles in
%            size class k, for each particle size and composition classes, at
%            time t in the i-th zone
% Mascknt:   matrix of masses of particles attached to bubbles in size class
%            k, for each particle size and composition classes, at time t in
%            all zones
% Cascknt:   matrix of mass concentrations of particles attached to bubbles in
%            size class k, for each particle size and composition classes, at
%            time t in all zones
% Maisckpz:  matrix of masses of particles attached to bubbles in size class
%            k, for each particle size and composition classes, at time t-1 in
%            the (i-1)-th zone
% Caisckpz:  matrix of mass concentrations of particles attached to bubbles in
%            size class k, for each particle size and composition classes, at
%            time t-1 in the (i-1)-th zone
% Maischnz:  matrix of masses of particles attached to bubbles in size class
%            k, for each particle size and composition classes, at time t-1 in
%            the (i+1)-th zone
% Caischnz:  matrix of mass concentrations of particles attached to bubbles in
%            size class k, for each particle size and composition classes, at
%            time t-1 in the(i+1)-th zone
% dMasck:    matrix of the changes in the masses of particles attached to
%            bubbles in size class k, for each particle size and composition
%            classes, at time t in any zone
% dCasck:    matrix of the changes in the mass concentrations of particles
```

```
%              attached to bubbles in size class k, for each particle size and
%              composition classes, at time t in any zone
% MancZ:       vector of masses of attached particles belonging to the n-th
%              particle size class regardless of composition, at time t in all
%              zones
% MancZT:      matrix of masses of attached particles belonging to the n-th
%              particle size class regardless of composition, from t=0...tfinal
%              in all zones
% MasnZ:       vector of masses of attached particles belonging to the n-th
%              composition class regardless of particle size, at time t in all
%              zones
% MasnZT:      matrix of masses of attached particles belonging to the n-th
%              composition class regardless of particle size, from t=0...tfinal
%              in all zones
% ManmZ:       matrix of masses of attached particles belonging to the n-th
%              particle size class and the m-th composition class, at time t for
%              all zones
% ManmpZ:      matrix of masses of particles in the n-th particle size class and
%              the m-th composition class, attached to bubbles in size class p,
%              at time t for all zones
% MaZ:         vector of the total mass of attached particles at time t in all
%              zones
% MaZT:        matrix of the total mass of attached particles from t=0 to
%              t=tfinal in all zones
% CaZT:        matrix of the total mass concentration of attached particles from
%              t=0 to t=tfinal in all zones
% Yent:        matrix of solids flowrates of each size and composition classes
%              carried by entrainment from the pulp to the interface
% Yfloat:      matrix of solids flowrates of each size and composition classes
%              carried by attachment to bubbles in each size class from the pulp
%              to the interface
% Yret:        matrix of solids flowrates of each size and composition classes
%              returned from the froth to the pulp
% Ytail:       matrix of solids flowrates of each size and composition classes
%              carried with the tailings
% YencP:       solids flowrate for size class n (all particle compositions)
%              carried by entrainment from the pulp to the interface
% YesnP:       solids flowrate for composition class n (all particle sizes)
%              carried by entrainment from the pulp to the interface
% YfncP:       solids flowrate for size class n (all particle compositions)
%              carried by flotation from the pulp to the interface
% YfsnP:       solids flowrate for composition class n (all particle sizes)
%              carried by flotation from the pulp to the interface
% YrncP:       solids flowrate for size class n (all particle compositions)
%              returned from the froth to the pulp
% YrsnP:       solids flowrate for composition class n (all particle sizes)
%              returned from the froth to the pulp
% YtncP:       solids flowrate for size class n (all particle compositions)
%              carried with the tailings
% YtsnP:       solids flowrate for composition class n (all particle sizes)
%              carried with the tailings
% RtncP:       pulp recovery of solids in size class n (all particle
%              compositions) carried with the tailings
% RtsnP:       pulp recovery of solids in composition class n (all particle
%              sizes) carried with the tailings
% RncP:        total pulp recovery of solids in size class n (all particle
%              compositions) from the pulp to the interface
% RsnP:        total pulp recovery of solids in composition class n (all
%              particle sizes)from the pulp to the interface
% RP:          total recovery of solids from the pulp to the interface by all
%              mechanisms
% Rt:          total recovery of solids in the tailings
% frRncP:      total froth recovery of solids in size class n (all particle
%              compositions) that cross the interface
% frRsnP:      total froth recovery of solids in composition class n (all
%              particle sizes)that cross the interface
% RF:          total recovery of solids in the froth regions
```

```
% Gsn:       concentrate grade at time t in terms of mineral species n.
% ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

% Assign matrix space:

Vgs=zeros(Nb,1);          Vgspz=zeros(Nb,1);
Vp=zeros(Np,Nc);          Vpnz=zeros(Np,Nc);
SGb=zeros(Nb,1);

Eknt=[0.10/Nb*ones(Nb,1+nZlp+1+nZup) 0.70/Nb*ones(Nb,1+nZsf+1+nZdf)];

Eik=zeros(Nb,1);          Eiknz=zeros(Nb,1);
Eikpz=zeros(Nb,1);        dEk=zeros(Nb,1);
Ek=zeros(Nb,1);           EZ=zeros(1,Nz);
EZT=[];
for i=1:Nb
      eval(['E' int2str(i) 'Z=zeros(1,Nz);']);
      eval(['E' int2str(i) 'ZT=[];']);
end

Mfisc=zeros(Np,Nc);       Mfiscnz=zeros(Np,Nc);
Mfiscpz=zeros(Np,Nc);     Mfscnt=zeros(Np,Nc*Nz);
dMfsc=zeros(Np,Nc);       Mfsc=zeros(Np,Nc);
Cfisc=zeros(Np,Nc);       Cfiscnz=zeros(Np,Nc);
Cfiscpz=zeros(Np,Nc);     Cfscnt=zeros(Np,Nc*Nz);
dCfsc=zeros(Np,Nc);       Cfsc=zeros(Np,Nc);

Maisck=zeros(Nb*Np,Nc);   Mascknt=zeros(Nb*Np,Nc*Nz);
Masck=zeros(Nb*Np,Nc);    Maiscknz=zeros(Nb*Np,Nc);
dMasck=zeros(Nb*Np,Nc);   Maisckpz=zeros(Nb*Np,Nc);
Caisck=zeros(Nb*Np,Nc);   Cascknt=zeros(Nb*Np,Nc*Nz);
Casck=zeros(Nb*Np,Nc);    Caiscknz=zeros(Nb*Np,Nc);
dCasck=zeros(Nb*Np,Nc);   Caisckpz=zeros(Nb*Np,Nc);

MfZ=zeros(1,Nz);          MfZT=[];
MaZ=zeros(1,Nz);          MaZT=[];
for i=1:Np
      eval(['Mf' int2str(i) 'cZ=zeros(1,Nz);']);
      eval(['Ma' int2str(i) 'cZ=zeros(1,Nz);']);
      eval(['Mf' int2str(i) 'cZT=[];']);
      eval(['Ma' int2str(i) 'cZT=[];']);
      eval(['Yf' int2str(i) 'cP = [];']);
      eval(['Ye' int2str(i) 'cP = [];']);
      eval(['Yr' int2str(i) 'cP = [];']);
      eval(['Yt' int2str(i) 'cP = [];']);
      eval(['Rt' int2str(i) 'cP = [];']);
      eval(['R' int2str(i) 'cP = [];']);
      eval(['frYf' int2str(i) 'cP = [];']);
      eval(['frYe' int2str(i) 'cP = [];']);
      eval(['frR' int2str(i) 'cP = [];']);
      for j=1:Nc
            eval(['Mf' int2str(i) int2str(j) 'Z=zeros(1,Nz);']);
            eval(['Ma' int2str(i) int2str(j) 'Z=zeros(1,Nz);']);
            for k=1:Nb
                  eval(['Ma' int2str(i) int2str(j) int2str(k) 'Z =
                  zeros(1,Nz);']);
            end
      end
end
for j=1:Nc
      eval(['Mfs' int2str(j) 'Z=zeros(1,Nz);']);
      eval(['Mas' int2str(j) 'Z=zeros(1,Nz);']);
      eval(['Mfs' int2str(j) 'ZT=[];']);
      eval(['Mas' int2str(j) 'ZT=[];']);
      eval(['Yfs' int2str(j) 'P = [];']);
```

```
      eval(['Yes' int2str(j) 'P = [];']);
      eval(['Yrs' int2str(j) 'P = [];']);
      eval(['Yts' int2str(j) 'P = [];']);
      eval(['Rts' int2str(j) 'P = [];']);
      eval(['Rs' int2str(j) 'P = [];']);
      eval(['frYfs' int2str(j) 'P = [];']);
      eval(['frYes' int2str(j) 'P = [];']);
      eval(['frRs' int2str(j) 'P = [];']);

end

Rt=[]; RP=[]; RF=[];


% ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

% Main Procedure:

mainproc;

% ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

% **COLUMN FLOTATION MODEL**

% **Main Procedure**

```
% ------------------------------------------------------------------------------
```

% Initial Conditions:

```
Tfinal=900;
Ntimes=Tfinal/dt;
Eikz=Eknt;
Mfiscz=Mfscnt;
Cfiscz=Cfscnt;
Maisckz=Mascknt;
Caisckz=Cascknt;
EZT=sum(Eikz);
for i=1:Nb
      eval(['E' int2str(i) 'ZT' '=Eikz(i,:);']);
end

for z=1:Nz
      for i=1:Np
            total1=0; total2=0;
            for j=1:Nc
                  total1=total1+Mfiscz(i,Nc*(z-1)+j);
                  total3=0;
                  for k=1:Nb
                        total3=total3 + Maisckz(Np*(k-1)+i,Nc*(z-1)+j);
                        total2=total2 + Maisckz(Np*(k-1)+i,Nc*(z-1)+j);
                  end
                  eval(['Ma' int2str(i) int2str(j) 'Z(z)=total3;']);
            end
            eval(['Mf' int2str(i) 'cZ(z)=total1;']);
            eval(['Mf' int2str(i) 'cZT=total1;']);
            eval(['Ma' int2str(i) 'cZ(z)=total2;']);
            eval(['Ma' int2str(i) 'cZT=total2;']);
      end
end

for z=1:Nz
      sum1=0; sum2=0;
      for j=1:Nc
            tot1=0; tot2=0;
            for i=1:Np
```

```
            tot1=tot1+Mfiscz(i,Nc*(z-1)+j);
            eval(['tot2=tot2 + Ma' int2str(i) int2str(j) 'Z(z);']);
        end
        eval(['Mfs' int2str(j) 'Z(z)=tot1;']);
        eval(['Mfs' int2str(j) 'ZT=tot1;']);
        eval(['Mas' int2str(j) 'Z(z)=tot2;']);
        eval(['Mas' int2str(j) 'ZT=tot2;']);
        eval(['sum1=Mfs' int2str(j) 'Z(z)+sum1;']);
        eval(['sum2=Mas' int2str(j) 'Z(z)+sum2;']);
        eval(['Gs' int2str(j) '= [];']);
    end
    MfZ(z)=sum1;
    MaZ(z)=sum2;
end
MfZT=MfZ;
MaZT=MaZ;
eprof=[];caprof=[];cfprof=[];
```

```
% ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

% **Time Iteration**

```
% -----------------------------------------------------------------------------------

for t=1:Ntimes,
```

% **Aeration zone**:

```
Vz=Lt*A;
z=1;
allVz(1,z)=Vz;
alllg(1,z)=Vz/A;
[Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caisckz,Blk,Vgs,Vgsnz,
Vp,Vpnz]=auxcalc(Eikz,Cfiscz,Caisckz,z,Vz);
[Ek,Mfsc,Masck,Ytail]=gaszone(Eik,Eiknz,Cfisc,Cfiscnz,Caisck,Caisckz,Blk,Vgs,
Vgsnz,Vp,Vpnz,Vz);
Eknt(:,z)=Ek;
Mfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc;
Cfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc/Vz;
Mascknt(:,(z-1)*Nc+1:z*Nc)=Masck;
Cascknt(:,(z-1)*Nc+1:z*Nc)=Masck/Vz;

for k=1:Nb
    eval(['E' int2str(k) 'Z(z)=Ek(k);']);
end
EZ(z)=sum(Ek);

for i=1:Np
    total1=0; total2=0;
    for j=1:Nc
        total1=total1+Mfsc(i,j);
        total3=0;
        for k=1:Nb
            total3=total3 + Masck(Np*(k-1)+i,j);
            total2=total2 + Masck(Np*(k-1)+i,j);
        end
        eval(['Ma' int2str(i) int2str(j) 'Z(z)=total3;']);
    end
    eval(['Mf' int2str(i) 'cZ(z)=total1;']);
    eval(['Ma' int2str(i) 'cZ(z)=total2;']);
end

sum1=0; sum2=0;
for j=1:Nc
    tot1=0; tot2=0;
```

```
        for i=1:Np
                tot1=tot1+Mfsc(i,j);
                eval(['tot2=tot2 + Ma' int2str(i) int2str(j) 'Z(z);']);
        end
        eval(['Mfs' int2str(j) 'Z(z)=tot1;']);
        eval(['Mas' int2str(j) 'Z(z)=tot2;']);
        eval(['sum1=Mfs' int2str(j) 'Z(z)+sum1;']);
        eval(['sum2=Mas' int2str(j) 'Z(z)+sum2;']);
end
MfZ(z)=sum1;
MaZ(z)=sum2;

Vgspz=Vgs;



% Pulp zones below feed port

counter=z;
Vz=Lzlp*A;
for z=counter+1:counter+nZlp,
allVz(1,z)=Vz;
alllg(1,z)=Vz/A;
[Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caisckn z,Blk,Vgs,Vgsnz,
Vp,Vpnz]=auxcalc(Eikz,Cfiscz,Caisckz,z,Vz);
[Ek,Mfsc,Masck]=lpzones(Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,Caisckpz,
Caisckn z,Blk,Vgs,Vgspz,Vgsnz,Vp,Vpnz,Vz);
Eknt(:,z)=Ek;Mfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc;Mascknt(:,(z-l)*Nc+1:z*Nc)=Masck;
Cfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc/Vz; Cascknt(:,(z-1)*Nc+1:z*Nc)=Masck/Vz;
for k=1:Nb
        eval(['E' int2str(k) 'Z(z)=Ek(k);']);
end
EZ(z)=sum(Ek);

for i=1:Np
        total1=0; total2=0;
        for j=1:Nc
                total1=total1+Mfsc(i,j);
                total3=0;
                for k=1:Nb
                        total3=total3 + Masck(Np*(k-1)+i,j);
                        total2=total2 + Masck(Np*(k-1)+i,j);
                end
                eval(['Ma' int2str(i) int2str(j) 'Z(z)=total3;']);
        end
        eval(['Mf' int2str(i) 'cZ(z)=total1;']);
        eval(['Ma' int2str(i) 'cZ(z)=total2;']);
end

sum1=0; sum2=0;
for j=1:Nc
        tot1=0; tot2=0;
        for i=1:Np
                tot1=tot1+Mfsc(i,j);
                eval(['tot2=tot2 + Ma' int2str(i) int2str(j) 'Z(z);']);
        end
        eval(['Mfs' int2str(j) 'Z(z)=tot1;']);
        eval(['Mas' int2str(j) 'Z(z)=tot2;']);
        eval(['sum1=Mfs' int2str(j) 'Z(z)+sum1;']);
        eval(['sum2=Mas' int2str(j) 'Z(z)+sum2;']);
end
MfZ(z)=sum1;
MaZ(z)=sum2;

Vgspz=Vgs;

end
```

% **Feed zone**:

```
Vz=Lt*A;
z=z+1;
allVz(1,z)=Vz;
alllg(1,z)=Vz/A;
[Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caiscknz,Blk,Vgs,Vgsnz,
Vp,Vpnz]=auxcalc(Eikz,Cfiscz,Caisckz,z,Vz);
[Ek,Mfsc,Masck]=feedzone(Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,
Caisckpz,Caiscknz,Blk,Vgs,Vgspz,Vgsnz,Vp,Vpnz,Vz);
Eknt(:,z)=Ek;Mfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc;Mascknt(:,(z-1)*Nc+1:z*Nc)=Masck;
Cfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc/Vz; Cascknt(:,(z-1)*Nc+1:z*Nc)=Masck/Vz;
for k=1:Nb
      eval(['E' int2str(k) 'Z(z)=Ek(k);']);
end
EZ(z)=sum(Ek);

for i=1:Np
      total1=0; total2=0;
      for j=1:Nc
            total1=total1+Mfsc(i,j);
            total3=0;
            for k=1:Nb
                  total3=total3 + Masck(Np*(k-1)+i,j);
                  total2=total2 + Masck(Np*(k-1)+i,j);
            end
            eval(['Ma' int2str(i) int2str(j) 'Z(z)=total3;']);
      end
      eval(['Mf' int2str(i) 'cZ(z)=total1;']);
      eval(['Ma' int2str(i) 'cZ(z)=total2;']);
end

sum1=0; sum2=0;
for j=1:Nc
      tot1=0; tot2=0;
      for i=1:Np
            tot1=tot1+Mfsc(i,j);
            eval(['tot2=tot2 + Ma' int2str(i) int2str(j) 'Z(z);']);
      end
      eval(['Mfs' int2str(j) 'Z(z)=tot1;']);
      eval(['Mas' int2str(j) 'Z(z)=tot2;']);
      eval(['sum1=Mfs' int2str(j) 'Z(z)+sum1;']);
      eval(['sum2=Mas' int2str(j) 'Z(z)+sum2;']);
end
MfZ(z)=sum1;
MaZ(z)=sum2;

Vgspz=Vgs;
```

% **Pulp zones above feed port:**

```
counter=z;
Vz=Lzup*A;
for z=counter+1:counter+nZup,
allVz(1,z)=Vz;
alllg(1,z)=Vz/A;
[Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caiscknz,Blk,Vgs,Vgsnz,
Vp,Vpnz]=auxcalc(Eikz,Cfiscz,Caisckz,z,Vz);
[Ek,Mfsc,Masck,Yfloat,Yent,Yret,temp2]=upzones(Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,
Cfiscnz,Caisck,Caisckpz,Caiscknz,Blk,Vgs,Vgspz,Vgsnz,Vp,Vpnz,Vz);
Eknt(:,z)=Ek;Mfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc;Mascknt(:,(z-1)*Nc+1:z*Nc)=Masck;
Cfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc/Vz; Cascknt(:,(z-1)*Nc+1:z*Nc)=Masck/Vz;
for k=1:Nb
      eval(['E' int2str(k) 'Z(z)=Ek(k);']);
```

```
end
EZ(z)=sum(Ek);

for i=1:Np
      total1=0; total2=0;
      for j=1:Nc
            total1=total1+Mfsc(i,j);
            total3=0;
            for k=1:Nb
                  total3=total3 + Masck(Np*(k-1)+i,j);
                  total2=total2 + Masck(Np*(k-1)+i,j);
            end
            eval(['Ma' int2str(i) int2str(j) 'Z(z)=total3;']);
      end
      eval(['Mf' int2str(i) 'cZ(z)=total1;']);
      eval(['Ma' int2str(i) 'cZ(z)=total2;']);
end

sum1=0; sum2=0;
for j=1:Nc
      tot1=0; tot2=0;
      for i=1:Np
            tot1=tot1+Mfsc(i,j);
            eval(['tot2=tot2 + Ma' int2str(i) int2str(j) 'Z(z);']);
      end
      eval(['Mfs' int2str(j) 'Z(z)=tot1;']);
      eval(['Mas' int2str(j) 'Z(z)=tot2;']);
      eval(['sum1=Mfs' int2str(j) 'Z(z)+sum1;']);
      eval(['sum2=Mas' int2str(j) 'Z(z)+sum2;']);
end
MfZ(z)=sum1;
MaZ(z)=sum2;

Vgspz=Vgs;

end


for i=1:Np
      total=0;total1=0;total1b=0;total2=0;total3=0;total4=0
      for j=1:Nc
            total=total+Yfloat(i,j);
            total1=total1+Yent(i,j);
            total1b=total1b+Yret(i,j);
            total2=total2+Ytail(i,j);
            total3=total3+Ytail(i,j)./(Qfeed*sum(Cfeed(i,:)));
            total4=total4+(Yent(i,j)+Yfloat(i,j))./
            (Qfeed*sum(Cfeed(i,:))+sum(Yret(i,:)));
      end
      eval(['Yf' int2str(i) 'cP = [Yf' int2str(i) 'cP;total];']);
      eval(['Ye' int2str(i) 'cP = [Ye' int2str(i) 'cP;total1];']);
      eval(['Yt' int2str(i) 'cP = [Yt' int2str(i) 'cP;total2];']);
      eval(['Rt' int2str(i) 'cP = [Rt' int2str(i) 'cP;total3];']);
      eval(['R' int2str(i) 'cP = [R' int2str(i) 'cP;total4];']);
end

allt=0;allte=0;alltf=0;alltt=0;
for j=1:Nc
      tot=0;tot1=0;tot2=0;tot3=0;tot4=0;
      for i=1:Np
            tot=tot+Yfloat(i,j);
            tot1=tot1+Yent(i,j);
            tot2=tot2+Ytail(i,j);
            tot3=tot3+Ytail(i,j)./(Qfeed*sum(Cfeed(:,j)));
            tot4=tot4+(Yent(i,j)+Yfloat(i,j))./
            (Qfeed*sum(Cfeed(:,j))+sum(Yret(i,:)));
            allt=allt+Yfloat(i,j)+Yent(i,j);
```

```
        alltt=alltt+Ytail(i,j);
    end
    eval(['Yfs' int2str(j) 'P = [Yfs' int2str(j) 'P;tot];']);
    eval(['Yes' int2str(j) 'P = [Yes' int2str(j) 'P;tot1];']);
    eval(['Yts' int2str(j) 'P = [Yts' int2str(j) 'P;tot2];']);
    eval(['Rts' int2str(j) 'P = [Rts' int2str(j) 'P;tot3];']);
    eval(['Rs' int2str(j) 'P = [Rs' int2str(j) 'P;tot4];']);
end

nrt=alltt./(Qfeed*sum(sum(Cfeed)));
nr=allt./(Qfeed*sum(sum(Cfeed))+sum(sum(Yret)));
Rt=[Rt;nrt];
RP=[RP;nr];
frfeed=Yfloat + Yent;


% Interface zone

Vz=Lt*A;
z=z+1;
allVz(1,z)=Vz;
alllg(1,z)=Vz/A;
[Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caisckn z,Blk,Vgs,
Vgsnz,Vp,Vpnz]=auxcalc(Eikz,Cfiscz,Caisckz,z,Vz);
frEi=Eik;
[Ek,Mfsc,Masck]=intface(Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,Caisckpz,
Caisckn z,Blk,Vgs,Vgspz,Vgsnz,Vp,Vpnz,Vz,temp2);
Eknt(:,z)=Ek;Mfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc;Mascknt(:,(z-1)*Nc+1:z*Nc)=Masck;
Cfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc/Vz; Cascknt(:,(z-1)*Nc+1:z*Nc)=Masck/Vz;
for k=1:Nb
    eval(['E' int2str(k) 'Z(z)=Ek(k);']);
end
EZ(z)=sum(Ek);

for i=1:Np
    total1=0; total2=0;
    for j=1:Nc
        total1=total1+Mfsc(i,j);
        total3=0;
        for k=1:Nb
            total3=total3 + Masck(Np*(k-1)+i,j);
            total2=total2 + Masck(Np*(k-1)+i,j);
        end
        eval(['Ma' int2str(i) int2str(j) 'Z(z)=total3;']);
    end
    eval(['Mf' int2str(i) 'cZ(z)=total1;']);
    eval(['Ma' int2str(i) 'cZ(z)=total2;']);
end

sum1=0; sum2=0;
for j=1:Nc
    tot1=0; tot2=0;
    for i=1:Np
        tot1=tot1+Mfsc(i,j);
        eval(['tot2=tot2 + Ma' int2str(i) int2str(j) 'Z(z);']);
    end
    eval(['Mfs' int2str(j) 'Z(z)=tot1;']);
    eval(['Mas' int2str(j) 'Z(z)=tot2;']);
    eval(['sum1=Mfs' int2str(j) 'Z(z)+sum1;']);
    eval(['sum2=Mas' int2str(j) 'Z(z)+sum2;']);
end
MfZ(z)=sum1;
MaZ(z)=sum2;

Vgspz=Vgs;
```

```
% Stabilized froth zones:

counter=z;
Vz=Lzsf*A;
for z=counter+1:counter+nZsf,
allVz(1,z)=Vz;
alllg(1,z)=Vz/A;
[Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caisckn z,Blk,Vgs,Vgsnz,
Vp,Vpnz]=auxcalc(Eikz,Cfiscz,Caisckz,z,Vz);
[Ek,Mfsc,Masck,frothYent,frothYfloat]=sfzones(Eik,Eikpz,Eiknz,frEi,Cfisc,
Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caisckn z,Blk,Vgs,Vgspz,Vgsnz,Vp,Vpnz,Vz,Lzsf,
sflambda);
Eknt(:,z)=Ek;Mfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc;Mascknt(:,(z-1)*Nc+1:z*Nc)=Masck;
Cfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc/Vz; Cascknt(:,(z-1)*Nc+1:z*Nc)=Masck/Vz;
for k=1:Nb
      eval(['E' int2str(k) 'Z(z)=Ek(k);']);
end
EZ(z)=sum(Ek);

for i=1:Np
      total1=0; total2=0;
      for j=1:Nc
            total1=total1+Mfsc(i,j);
            total3=0;
            for k=1:Nb
                  total3=total3 + Masck(Np*(k-1)+i,j);
                  total2=total2 + Masck(Np*(k-1)+i,j);
            end
            eval(['Ma' int2str(i) int2str(j) 'Z(z)=total3;']);
      end
      eval(['Mf' int2str(i) 'cZ(z)=total1;']);
      eval(['Ma' int2str(i) 'cZ(z)=total2;']);
end

sum1=0; sum2=0;
for j=1:Nc
      tot1=0; tot2=0;
      for i=1:Np
            tot1=tot1+Mfsc(i,j);
            eval(['tot2=tot2 + Ma' int2str(i) int2str(j) 'Z(z);']);
      end
      eval(['Mfs' int2str(j) 'Z(z)=tot1;']);
      eval(['Mas' int2str(j) 'Z(z)=tot2;']);
      eval(['sum1=Mfs' int2str(j) 'Z(z)+sum1;']);
      eval(['sum2=Mas' int2str(j) 'Z(z)+sum2;']);
end
MfZ(z)=sum1;
MaZ(z)=sum2;

Vgspz=Vgs;

end


% Wash water addition zone:

Vz=Lt*A;
z=z+1;
allVz(1,z)=Vz;
alllg(1,z)=Vz/A;
[Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caisckn z,Blk,Vgs,Vgsnz,
Vp,Vpnz]=auxcalc(Eikz,Cfiscz,Caisckz,z,Vz);
[Ek,Mfsc,Masck,frothYent,frothYfloat]=wwzone(Eik,Eikpz,Eiknz,frEi,Cfisc,
Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caisckn z,Blk,Vgs,Vgspz,Vgsnz,Vp,Vpnz,Vz,Lt,
wwlambda);
Eknt(:,z)=Ek;Mfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc;Mascknt(:,(z-1)*Nc+1:z*Nc)=Masck;
Cfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc/Vz; Cascknt(:,(z-1)*Nc+1:z*Nc)=Masck/Vz;
```

```
for k=1:Nb
      eval(['E' int2str(k) 'Z(z)=Ek(k);']);
end
EZ(z)=sum(Ek);

for i=1:Np
      total1=0; total2=0;
      for j=1:Nc
            total1=total1+Mfsc(i,j);
            total3=0;
            for k=1:Nb
                  total3=total3 + Masck(Np*(k-1)+i,j);
                  total2=total2 + Masck(Np*(k-1)+i,j);
            end
            eval(['Ma' int2str(i) int2str(j) 'Z(z)=total3;']);
      end
      eval(['Mf' int2str(i) 'cZ(z)=total1;']);
      eval(['Ma' int2str(i) 'cZ(z)=total2;']);
end

sum1=0; sum2=0;
for j=1:Nc
      tot1=0; tot2=0;
      for i=1:Np
            tot1=tot1+Mfsc(i,j);
            eval(['tot2=tot2 + Ma' int2str(i) int2str(j) 'Z(z);']);
      end
      eval(['Mfs' int2str(j) 'Z(z)=tot1;']);
      eval(['Mas' int2str(j) 'Z(z)=tot2;']);
      eval(['sum1=Mfs' int2str(j) 'Z(z)+sum1;']);
      eval(['sum2=Mas' int2str(j) 'Z(z)+sum2;']);
end
MfZ(z)=sum1;
MaZ(z)=sum2;

Vgspz=Vgs;


% Draining froth zones:

counter=z;
Vz=Lzdf*A;
for z=counter+1:counter+nZdf,
allVz(1,z)=Vz;
alllg(1,z)=Vz/A;
[Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caisckn z,Blk,Vgs,Vgsnz,
Vp,Vpnz]=auxcalc(Eikz,Cfiscz,Caisckz,z,Vz);
[Ek,Mfsc,Masck,frothYent,frothYfloat]=dfzones(Eik,Eikpz,Eiknz,frEi,Cfisc,
Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caisckn z,Blk,Vgs,Vgspz,Vgsnz,Vp,Vpnz,Vz,Lzdf,
dflambda);
Eknt(:,z)=Ek;Mfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc;Mascknt(:,(z-1)*Nc+1:z*Nc)=Masck;
Cfscnt(:,(z-1)*Nc+1:z*Nc)=Mfsc/Vz; Cascknt(:,(z-1)*Nc+1:z*Nc)=Masck/Vz;
for k=1:Nb
      eval(['E' int2str(k) 'Z(z)=Ek(k);']);
end
EZ(z)=sum(Ek);

for i=1:Np
      total1=0; total2=0;
      for j=1:Nc
            total1=total1+Mfsc(i,j);
            total3=0;
            for k=1:Nb
                  total3=total3 + Masck(Np*(k-1)+i,j);
                  total2=total2 + Masck(Np*(k-1)+i,j);
            end
            eval(['Ma' int2str(i) int2str(j) 'Z(z)=total3;']);
```

```
        end
        eval(['Mf' int2str(i) 'cZ(z)=total1;']);
        eval(['Ma' int2str(i) 'cZ(z)=total2;']);
end

sum1=0; sum2=0;
for j=1:Nc
        tot1=0; tot2=0;
        for i=1:Np
                tot1=tot1+Mfsc(i,j);
                eval(['tot2=tot2 + Ma' int2str(i) int2str(j) 'Z(z);']);
        end
        eval(['Mfs' int2str(j) 'Z(z)=tot1;']);
        eval(['Mas' int2str(j) 'Z(z)=tot2;']);
        eval(['sum1=Mfs' int2str(j) 'Z(z)+sum1;']);
        eval(['sum2=Mas' int2str(j) 'Z(z)+sum2;']);
end
MfZ(z)=sum1;
MaZ(z)=sum2;

Vgspz=Vgs;

end


for i=1:Np
total=0;total1=0;total2=0;
for j=1:Nc
total=total+frothYfloat(i,j);
total1=total1+frothYent(i,j);
 if sum(frfeed(i,:)) > 0
        total2=total2+(frothYent(i,j)+frothYfloat(i,j))./sum(frfeed(i,:));
 end
end
eval(['frYf' int2str(i) 'cP = [frYf' int2str(i) 'cP;total];']);
eval(['frYe' int2str(i) 'cP = [frYe' int2str(i) 'cP;total1];']);
eval(['frR' int2str(i) 'cP = [frR' int2str(i) 'cP;total2];']);
end


allt=0;
for j=1:Nc
        tot=0;tot1=0;tot2=0;tot3=0;tot4=0;
        for i=1:Np
                tot=tot+frothYfloat(i,j);
                tot1=tot1+frothYent(i,j);
                if sum(frfeed(:,j)) > 0
                        tot2=tot2+(frothYent(i,j)+frothYfloat(i,j))./
                        sum(frfeed(:,j));
                end
                allt=allt+frothYfloat(i,j)+frothYent(i,j);
        end
        eval(['frYfs' int2str(j) 'P = [frYfs' int2str(j) 'P;tot];']);
        eval(['frYes' int2str(j) 'P = [frYes' int2str(j) 'P;tot1];']);
        eval(['frRs' int2str(j) 'P = [frRs' int2str(j) 'P;tot2];']);
end
if allt > 0
        for j=1:Nc
                eval(['Gs' int2str(j) '= [Gs' int2str(j) ';Ys' int2str(j)
                '/allt];']);
        end
end

nfrr=0;
if sum(sum(frfeed)) > 0
        nfrr=allt./sum(sum(frfeed));
end
```

```
RF=[RF;nfrr];

%^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

Eikz=Eknt;
Cfiscz=Cfscnt;
Mfiscz=Mfscnt;
Caisckz=Cascknt;
Maisckz=Mascknt;

EZT=[EZT;EZ];
CfZ=MfZ./allVz;
CaZ=MaZ./allVz;
MfZT=[MfZT;MfZ];
MaZT=[MaZT;MaZ];
for k=1:Nb
      eval(['E' int2str(k) 'ZT' '=[E' int2str(k) 'ZT;E' int2str(k) 'Z];']);
end
for i=1:Np
      eval(['Mf' int2str(i) 'cZT' '=[Mf' int2str(i) 'cZT;Mf' int2str(i)
      'cZ];']);
      eval(['Ma' int2str(i) 'cZT' '=[Ma' int2str(i) 'cZT;Ma' int2str(i)
      'cZ];']);
end

for j=1:Nc
      eval(['Mfs' int2str(j) 'ZT' '=[Mfs' int2str(j) 'ZT;Mfs' int2str(j)
      'Z];']);
      eval(['Mas' int2str(j) 'ZT' '=[Mas' int2str(j) 'ZT;Mas' int2str(j)
      'Z];']);
end


% -------------------------------------------------------------------------------------------
end
for z=1:Nz
      zlg=alllg(z);
      i=0;
      while i <= zlg
            eprof=[eprof;EZ(z)];
            cfprof=[cfprof;MfZ(z)/allVz(z)];
            caprof=[caprof;MaZ(z)/allVz(z)];
            i=i+0.1;
      end
end
% ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

% **COLUMN FLOTATION MODEL**

% **Defined Functions:**

% ---

% 1. Function that calls the ones that determine the bubble rise velocities, particle settling velocities, slurry densities, bubble densities and bubble coverage at each time step.

```
function
[Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caiscknz,Blk,Vgs,Vgsnz,
Vp,Vpnz]=auxcalc(Eikz,Cfiscz,Caisckz,z,Vz)
```

```
global Ul Nb Np Nc Nz SGw

Blsck=zeros(Np*Nb,Nc); Blk=zeros(Nb,1);
Eik=Eikz(:,z);Cfisc=Cfiscz(:,(z-1)*Nc+1:z*Nc);
Caisck=Caisckz(:,(z-1)*Nc+1:z*Nc);

[Blsck,Blk]=bubload(Caisck,Eik);
SGb=bubdens(Blsck);
SGsl=slurdens(Cfisc,Eik);
SGsusp=suspdens(Eik,SGb,SGsl);
Usl=slurvis(Cfisc,Eik);
Vgs=bubslip(Eik,SGb,SGsl,SGsusp,Usl);
Vp=partslip(Eik,Cfisc,SGsl);

if (z > 1)
      Eikpz=Eikz(:,z-1); Cfiscpz=Cfiscz(:,(z-2)*Nc+1:(z-1)*Nc);
      Caisckpz=Caisckz(:,(z-2)*Nc+1:(z-1)*Nc);
end

if (z < Nz)
      Eiknz=Eikz(:,z+1); Cfiscnz=Cfiscz(:,z*Nc+1:(z+1)*Nc);
      Caiscknz=Caisckz(:,z*Nc+1:(z+1)*Nc); Blscknz=bubload(Caiscknz,Eiknz);
      SGbnz=bubdens(Blscknz); SGslnz=slurdens(Cfiscnz,Eiknz);
      SGsuspnz=suspdens(Eiknz,SGbnz,SGslnz);
      Vgsnz=bubslip(Eiknz,SGbnz,SGslnz,SGsuspnz,Usl);
      Vpnz=partslip(Eiknz,Cfiscnz,SGslnz);
end

if z==Nz
      Eiknz=ones(Nb,1); Cfiscnz=zeros(Np,Nc);
      Caiscknz=zeros(Np*Nb,Nc); Blscknz=zeros(Np*Nb,Nc); SGbnz=zeros(Nb,1);
      Sgslnz=0; SGsuspnz=0;
      Vgsnz=zeros(Nb,1); Vpnz=zeros(Np,Nc);
end

%==========================================================================================

% 2. Function to calculate the density of bubbles loaded with particles

function densb=bubdens(BLsck)
%
%
global Nb Nc Np Db Dp SGp;

densb=zeros(Nb,1);
for k=1:Nb,
      total=0;
      for j=1:Nc,
            for i=1:Np,
                  total=total + Dp(i)*SGp(j)*BLsck(Np*(k-1)+i, j);
            end
      end
      densb(k)=4*total/Db(k);
end

%==========================================================================================


% 3. Function to calculate the loading of bubbles of size class k.


function [blsck,blk]=bubload(CAsck,E)
global Db Dp SGp Nb Np Nc;

blsck=zeros(Np*Nb,Nc); blk=zeros(Nb,1);
```

```
for i=1:Np,
      for j=1:Nc,
            for k=1:Nb,
              blsck(Np*(k-1)+i, j)=CAsck(Np*(k-1)+i, j)*Db(k)/
              (4*SGp(j)*Dp(i)*E(k));
              blk(k)=blk(k)+blsck(Np*(k-1)+i, j);
            end
      end
end

%=============================================================================

% 4. Function to calculate the hindered rise velocity of bubbles in the pulp,
for db<2mm

function [Ugs,m]=bubslip(E,SGb,SGsl,SGsusp,Usl)
%
%
% Ugs=((Db)^2 * (bubble dens - susp. dens) * g * F(1-air fraction))/
% (18*mu*(1 + 0.15*(Reb)^0.687))
%
% F --> Richardson and Zaki: (1-e)^m-2
%
% m=(4.45+(18* (Db/Dc)^2)) * (Re)^(-0.1) for 1 < Re < 200
% m=4.45* (Re)^(-0.1) for 200 < Re < 500
% m=2.39              for Re > 500
%
% Re=(Db* slurry dens * Ut)/mu
% Terminal velocity Ut = (Db)^2 * (bubble dens - water dens) * g /
% (18*mu*(1 + 0.15*(Re)^0.687))
%
% Procedure:
%
%     Iterate for Ut, Re
%     Calculate m
%     Iterate for Ugs, Res
%
global G  Db Ul Nb SGw A

Ugs=zeros(Nb,1);
Dc=sqrt(4*A/pi);

for k=1:Nb,

Re=200; pRe=0;
while abs(pRe-Re) > 0.1
      Ut=G*Db(k)*Db(k)*abs(SGb(k)-SGw)/(18*Ul*(1+0.15*(Re^0.687)));
      pRe=Re;
      Re=Db(k)*Ut*SGw/Ul;
end
if Re < 200,
      m=(4.45 + 18*(Db(k)/Dc)^2)*(Re^(-0.1));
elseif Re < 500,
      m=4.45*(Re^(-0.1));
else
      m=2.39;
end

Res=200; pRes=0;
while abs(pRes-Res) > 0.1
  Us=G*Db(k)*Db(k)*abs(SGb(k)-SGsusp)*((1-sum(E))^(m-2))/
  (18*Usl*(1+0.15*(Res^0.687)));
  pRes=Res;
  Res=Db(k)*Us*SGsl*(1-sum(E))/Usl;
end
```

```
Ugs(k)=Us;

end

% ==========================================================================================
```

% <u>5. Functions to calculate the solids detached in the froth zones due to the reduction in bubble surface area caused by coalescence.</u>

% a) Using

% $\Delta S_k^{'} = \sum_{j}^{Nb-k} f\left(\frac{D_{k,j}\beta_k}{Db_k} - \frac{A_{k,j}\beta_{\max}}{Db_{k+j}}\right)$

```
-----------------------------------------------------------------------------------------------
 function [cdetach,tcdetach]=detment(Caisck,Eik,lambda,D,Ap)

global Np Nc Nb SGp Db Dp maxBlk

cdetach=zeros(Nb*Np,Nc);
tcdetach=zeros(Np,Nc);
[Blsck,Blk]=bubload(Caisck,Eik);

Dij=zeros(Nb,Nb);

for i=1:Nb-1
	for j=1:Nb-i
		B=0;
		for l=1:Nb
			B=Eik(l).*Db(j).*Db(j).*Db(j)./(Db(l).*Db(l).*Db(l)) + B;
		end
		Dij(i,j)=lambda(i,j).*Eik(i).*Eik(j)./B;
	end
end

Apij=zeros(Nb,Nb);
for i=1:Nb-1
	for j=1:Nb-i
		B=0;
		for l=1:Nb
			B=Eik(l).*Db(j).*Db(j).*Db(j).*Db(i).*Db(i).*Db(i)./
			(Db(l).*Db(l).*Db(l).*Db(i+j).*Db(i+j).*Db(i+j)) + B;
		end
		Apij(i,j)=lambda(i,j).*Eik(j).*Eik(i)./B;
	end
end
Apij=0.5*Apij;

extrapart=zeros(Nb,1);

for k=1:Nb-1
	for j=1:Nb-k

	extrapart(k)=6*Dij(k,j)*Blk(k)/Db(k) - 6*Apij(k,j)*maxBlk/Db(k+j) +
	extrapart(k);

	end
end

for k=1:Nb
if extrapart(k) > 0
```

```
for j=Nc:-1:1
for i=Np:-1:1
      if (6*D(k)*Blsck(Np*(k-1)+i,j)/Db(k) - extrapart(k)) <= 0
            cdetach(Np*(k-1)+i,j)=4*(6*D(k)*Blsck(Np*(k-1)+i,j)/
            Db(k))*SGp(j)*Dp(i);
            extrapart(k)=extrapart(k)-6*D(k)*Blsck(Np*(k-1)+i,j)/Db(k);
            tcdetach(i,j)=cdetach(Np*(k-1)+i,j) + tcdetach(i,j);

      else

            cdetach(Np*(k-1)+i,j)=4*extrapart(k)*SGp(j)*Dp(i);
            extrapart(k)=0;
            tcdetach(i,j)=cdetach(Np*(k-1)+i,j) + tcdetach(i,j);

      end
end
end

end
end
```

% b) Using

% $detach_{s,c,k} = \frac{24 D_k \beta_{s,c,k} \rho_c Dp_s}{Db_k}$

```
function [cdetach,tcdetach]=detment(Caisck,Eik)

global dt Np Nc Nb SGp Db Dp maxBlk;

cdetach=zeros(Nb*Np,Nc);
tcdetach=zeros(Np,Nc);
[Blsck,Blk]=bubload(Caisck,Eik)

for k=1:Nb

      for i=1:Np
            for j=1:Nc
                  cdetach(Np*(k-1)+i,j)=24*Dp(i)*SGp(j)*D(k)*
                  Blsck(Np*(k-1)+i,j)/Db(k);
                  tcdetach(i,j)=24*Dp(i)*SGp(j)*D(k)*
                  Blsck(Np*(k-1)+i,j)/Db(k) + tcdetach(i,j);
            end
      end

end

% ==============================================================================
```

% 6. Function to calculate the settling velocity of particles.

```
function Ups=partslip(E,Cfsc,SGsl)
%
% Up=((Dp)^2 *(particle dens-suspension dens)*g*F(1-particle volume frac))/
% (18*mu*(1 + 0.15*(Rep)^0.687))
%
% F --> Richardson and Zaki: (1-e)^m
```

```
%
% m=(4.45+(18* (Db/Dc)^2)) * (Re)^(-0.1) for 1 < Re < 200
% m=4.45* (Re)^(-0.1) for 200 < Re < 500
% m=2.39              for Re > 500
%
% Re=(Dp* slurry dens * Ut)/mu
% Terminal velocity Ut = (Dp)^2 * (part. dens - water dens) * g /
% (18*mu*(1 + 0.15*(Re)^0.687))
%
% Procedure:
%
%     Iterate for Ut, Re
%     Calculate m
%     Iterate for Ups, Res
%
global G Ul Np Nc Dp SGp SGw A;

Ups=zeros(Np,Nc);
Dc=sqrt(4*A/pi);
total=0;
for j=1:Nc,
      for i=1:Np,
            total=total + (Cfsc(i,j)/(SGp(j)*(1-sum(E))));
      end
end

for j=1:Nc,
      for i=1:Np,

            Re=200; pRe=0;
            while abs(pRe-Re) > 0.1
                  Ut=G*Dp(i)*Dp(i)*abs(SGp(j)-SGw)/
                  (18*Ul*(1+0.15*(Re^0.687)));
                  pRe=Re;
                  Re=Dp(i)*Ut*SGw/Ul;
            end
            if Re < 200,
                  m=(4.45 + 18*(Dp(i)/Dc)^2)*(Re^(-0.1));
            elseif Re < 500,
                  m=4.45*(Re^(-0.1));
            else
                  m=2.39;
            end


            Res=200; pRes=0;
            while abs(pRes-Res) > 0.1
                  Us=G*Dp(i)*Dp(i)*abs(SGp(j)-SGsl)*((1-total)^(m-1))/
                  (18*Ul*(1+0.15*(Res^0.687)));
                  pRes=Res;
                  Res=Dp(i)*Us*SGw*(1-total)/Ul;
            end

            Ups(i,j)=Us;
      end
end

% ==========================================================================================
```

% 7. Function to calculate the slurry density from the liquid and solids densities and the volume fractions.

```
function denssl=slurdens(CFsc,E)
%
```

```
%
global Np Nc SGp SGw;

sum1=0;
sum2=0;

for j=1:Nc,
      for i=1:Np,
            sum1=sum1 + (CFsc(i,j)/((1-sum(E))));
            sum2=sum2 + (CFsc(i,j)/((1-sum(E))*SGp(j)));
      end
end

denssl=sum1 + SGw*(1-sum2);

%==========================================================================================
```

% 8. Function to calculate the suspension density from the bubbles and solids densities and the volume fractions.

```
function denssusp=suspdens(E,SGb,SGsl)
%
%
denssusp= sum(E.*SGb) + (1-sum(E))*SGsl;


%==========================================================================================
```

% 9. Function to calculate the changes in air fraction, free solids concentration and attached solids concentration at each time step in the aeration zone.

```
function [Ek,Mfsc,Masck,Ytail]=
gaszone(Eik,Eiknz,Cfisc,Cfiscnz,Caisck,Caiscknz,Blk,Vgs,Vgsnz,Vp,Vpnz,Vz)

global A Nb Np Nc kasck fvd dt maxBlk Qg Qt

Qsl=Qt; Qslnz=Qt;
attachm=zeros(Nb*Np,Nc);
tattachm=zeros(Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
      tattachm(i,j)=(kasck(Np*(k-1)+i,j).*(1-Blk(k)./maxBlk).*
      fvd(k).*Cfisc(i,j)) +     tattachm(i,j);
      attachm(Np*(k-1)+i,j)=(kasck(Np*(k-1)+i,j).*
      (1-Blk(k)./maxBlk).*fvd(k).*Cfisc(i,j));
    end
  end
end

VgtimesMa=zeros(Nb*Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
      VgtimesMa(Np*(k-1)+i,j)=Vgs(k).*Caisck(Np*(k-1)+i,j);
    end
  end
end

dEk=(Qg*fvd - (Qg-Qslnz-sum(Vgs.*Eik)*A)*Eik  - (Vgs.*Eik)*A)/Vz;
```

```
dCfsc=(-(Qg-Qslnz+sum(Vgs.*Eik)*A*(1-sum(Eik))/sum(Eik))*Cfisc +
(sum(Vgs.*Eik)*A/sum(Eik))*Cfiscnz +(Vpnz.*Cfiscnz)*A - Qsl*Cfisc-
(Vp.*Cfisc)*A)/Vz - tattachm;

dCasck=(-(Qg-Qslnz-sum(Vgs.*Eik)*A)*Caisck - VgtimesMa*A)/Vz + attachm;

Ek=Eik + dt*dEk
Cfsc=Cfisc + dt*dCfsc;
Mfsc=Cfsc*Vz;
Casck=Caisck + dt*dCasck;
Masck=Casck*Vz;

Ek(Ek<0)=zeros(size(Ek(Ek<0)));
Mfsc(Mfsc<0)=zeros(size(Mfsc(Mfsc<0)));
Masck(Masck<0)=zeros(size(Masck(Masck<0)));

Ytail=Qsl*Cfisc+(Vp.*Cfisc)*A;

%==========================================================================================


% 10. Function to calculate the changes in air fraction, free solids
concentration and
% attached solids concentration at each time step in each of the lower-
collection-region
% zones.


function [Ek,Mfsc,Masck]=lpzones(Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,Caisck,
Caisckpz,Caiscknz,Blk,Vgs,Vgspz, Vgsnz,Vp,Vpnz,Vz)

global A Nb Np Nc kasck fvd dt maxBlk Qg Qt

Qsl=Qt; Qslnz=Qt;
attachm=zeros(Nb*Np,Nc);
tattachm=zeros(Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
      tattachm(i,j)=(kasck(Np*(k-1)+i,j).*(1-Blk(k)./maxBlk).*
      fvd(k).*Cfisc(i,j)) +     tattachm(i,j);
      attachm(Np*(k-1)+i,j)=(kasck(Np*(k-1)+i,j).*
      (1-Blk(k)./maxBlk).*fvd(k).*Cfisc(i,j));
    end
  end
end

VgtimesMa=zeros(Nb*Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
      VgtimesMa(Np*(k-1)+i,j)=Vgs(k).*Caisck(Np*(k-1)+i,j);
    end
  end
end
VgtimesMapz=zeros(Nb*Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
      VgtimesMapz(Np*(k-1)+i,j)=Vgspz(k).*Caisckpz(Np*(k-1)+i,j);
    end
  end
end
dEk=(-(Qg-Qslnz-sum(Vgs.*Eik)*A)*Eik + (Qg-Qsl-sum(Vgspz.*Eikpz)*A)*Eikpz +
(Vgspz.*Eikpz)*A - (Vgs.*Eik)*A)/Vz;
```

```
dCfsc=(-(Qg-Qslnz+sum(Vgs.*Eik)*A*(1-sum(Eik))/sum(Eik))*Cfisc +
(sum(Vgs.*Eik)*A/sum(Eik))*Cfiscnz +(Vpnz.*Cfiscnz)*A +
(Qg-Qsl+sum(Vgspz.*Eikpz)*A*(1-sum(Eikpz))/sum(Eikpz))*Cfiscpz -
(sum(Vgspz.*Eikpz)*A/sum(Eikpz))*Cfisc -(Vp.*Cfisc)*A)/Vz - tattachm;

dCasck=(-(Qg-Qslnz-sum(Vgs.*Eik)*A)*Caisck - VgtimesMa*A +
(Qg-Qsl-sum(Vgspz.*Eikpz)*A)*Caisckpz + VgtimesMapz*A)/Vz + attachm;

Ek=Eik + dt*dEk;
Cfsc=Cfisc + dt*dCfsc;
Mfsc=Cfsc*Vz;
Casck=Caisck + dt*dCasck;
Masck=Casck*Vz;

Ek(Ek<0)=zeros(size(Ek(Ek<0)));
Mfsc(Mfsc<0)=zeros(size(Mfsc(Mfsc<0)));
Masck(Masck<0)=zeros(size(Masck(Masck<0)));

%===========================================================================================
```

% <u>11. Function to calculate the changes in air fraction, free solids concentration and attached solids concentration at each time step in the feed zone.</u>

```
function [Ek,Mfsc,Masck]=feedzone(Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,Cfiscnz,
Caisck,Caisckpz,Caisscknz,Blk,Vgs,Vgspz,Vgsnz,Vp,Vpnz,Vz)

global A Nb Np Nc kasck fvd dt maxBlk Qg Qt Qb Qfeed Cfeed

Qsl=Qt; Qslnz=Qb;

attachm=zeros(Nb*Np,Nc);
tattachm=zeros(Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
      tattachm(i,j)=(kasck(Np*(k-1)+i,j).*(1-Blk(k)./maxBlk).*
      fvd(k).*Cfisc(i,j)) +     tattachm(i,j);
      attachm(Np*(k-1)+i,j)=(kasck(Np*(k-1)+i,j).*(1-Blk(k)./
      maxBlk).*fvd(k).*Cfisc(i,j));
    end
  end
end

VgtimesMa=zeros(Nb*Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
      VgtimesMa(Np*(k-1)+i,j)=Vgs(k).*Caisck(Np*(k-1)+i,j);
    end
  end
end
VgtimesMapz=zeros(Nb*Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
      VgtimesMapz(Np*(k-1)+i,j)=Vgspz(k).*Caisckpz(Np*(k-1)+i,j);
    end
  end
end

dEk=(-(Qg-Qslnz-sum(Vgs.*Eik)*A)*Eik + (Qg-Qsl-sum(Vgspz.*Eikpz)*A)*Eikpz +
(Vgspz.*Eikpz)*A - (Vgs.*Eik)*A)/Vz;

dCfsc= (Qfeed*Cfeed)/Vz + (-(Qg-Qslnz+sum(Vgs.*Eik)*A*
```

```
(1-sum(Eik))/sum(Eik))*Cfisc +(sum(Vgs.*Eik)*A/sum(Eik))*Cfiscnz +
(Vpnz.*Cfiscnz)*A + (Qg-Qsl+sum(Vgspz.*Eikpz)*A*(1-sum(Eikpz))/sum(Eikpz))*
Cfiscpz-(sum(Vgspz.*Eikpz)*A/sum(Eikpz))*Cfisc -(Vp.*Cfisc)*A)/Vz - tattachm;

dCasck=((Qg-Qsl-sum(Vgspz.*Eikpz)*A)*Caisckpz + VgtimesMapz*A -(Qg-Qslnz-
sum(Vgs.*Eik)*A)*Caisck - VgtimesMa*A)/Vz + attachm;

Ek=Eik + dt*dEk;
Cfsc=Cfisc + dt*dCfsc;
Mfsc=Cfsc*Vz;
Casck=Caisck + dt*dCasck;
Masck=Casck*Vz;

Ek(Ek<0)=zeros(size(Ek(Ek<0)));
Mfsc(Mfsc<0)=zeros(size(Mfsc(Mfsc<0)));
Masck(Masck<0)=zeros(size(Masck(Masck<0)));


%=============================================================================================


% 12. Function to calculate the changes in air fraction, free solids
concentration and attached solids concentration at each time step in each of
the upper-collection-region zones.


function [Ek,Mfsc,Masck,Yfloat,Yent,Yret,temp2]=upzones(Eik,Eikpz,Eiknz,Cfisc,
Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caiscknz,Blk,Vgs,Vgspz,Vgsnz,Vp,Vpnz,Vz)

global A Nb Np Nc kasck fvd dt maxBlk Qg Qb

Qsl=Qb; Qslnz=Qb;
attachm=zeros(Nb*Np,Nc);
tattachm=zeros(Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
       tattachm(i,j)=(kasck(Np*(k-1)+i,j).*(1-Blk(k)./maxBlk).*
       fvd(k).*Cfisc(i,j)) +     tattachm(i,j);
       attachm(Np*(k-1)+i,j)=(kasck(Np*(k-1)+i,j).*
       (1-Blk(k)./maxBlk).*fvd(k).*Cfisc(i,j));
    end
  end
end

VgtimesMa=zeros(Nb*Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
       VgtimesMa(Np*(k-1)+i,j)=Vgs(k).*Caisck(Np*(k-1)+i,j);
    end
  end
end
VgtimesMapz=zeros(Nb*Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
       VgtimesMapz(Np*(k-1)+i,j)=Vgspz(k).*Caisckpz(Np*(k-1)+i,j);
    end
  end
end

dEk=(-(Qg-Qslnz-sum(Vgs.*Eik)*A)*Eik + (Qg-Qsl-sum(Vgspz.*Eikpz)*A)*Eikpz +
(Vgspz.*Eikpz)*A - (Vgs.*Eik)*A)/Vz;

dCfsc=(-(Qg-Qslnz+sum(Vgs.*Eik)*A*(1-sum(Eik))/sum(Eik))*Cfisc +
(sum(Vgs.*Eik)*A/sum(Eik))*Cfiscnz +(Vpnz.*Cfiscnz)*A +
```

```
(Qg-Qsl+sum(Vgspz.*Eikpz)*A*(1-sum(Eikpz))/sum(Eikpz))*Cfiscpz -
(sum(Vgspz.*Eikpz)*A/sum(Eikpz))*Cfisc -(Vp.*Cfisc)*A)/Vz - tattachm;

dCasck=(-(Qg-Qslnz-sum(Vgs.*Eik)*A)*Caisck - VgtimesMa*A +
(Qg-Qsl-sum(Vgspz.*Eikpz)*A)*Caisckpz + VgtimesMapz*A)/Vz + attachm;

Ek=Eik + dt*dEk;
Cfsc=Cfisc + dt*dCfsc;
Mfsc=Cfsc*Vz;
Casck=Caisck + dt*dCasck;
Masck=Casck*Vz;

Ek(Ek<0)=zeros(size(Ek(Ek<0)));
Mfsc(Mfsc<0)=zeros(size(Mfsc(Mfsc<0)));
Masck(Masck<0)=zeros(size(Masck(Masck<0)));

Yent=zeros(Np,Nc); Yfloat=zeros(Np,Nc);
Yent=(Qg-Qslnz+sum(Vgs.*Eik)*A*(1-sum(Eik))/sum(Eik))*Cfisc;
Yret= (sum(Vgs.*Eik)*A/sum(Eik))*Cfiscnz + (Vpnz.*Cfiscnz)*A;
temp2=zeros(Np,Nc); V=Qg/(A*sum(Eik)); Vpz=Qg/(A*sum(Eikpz));
dV=-Qg/(A*(sum(Eik))^2);
for i=1:Np
 for j=1:Nc
       total=0;tot=0;
       for k=1:Nb
              total=(Qg-Qslnz-sum(Vgs.*Eik)*A)*Caisck(Np*(k-1)+i,j)+
              VgtimesMa(Np*(k-1)+i,j)*A + total;
              tot=V*A*Caisck(Np*(k-1)+i,j) + tot;
       end
       Yfloat(i,j)=total;
       temp2(i,j)=tot;
  end
end

% ==========================================================================================
```

% 13. Function to calculate the changes in air fraction, free solids concentration and attached solids concentration at each time step in the interface region.

```
function [Ek,Mfsc,Masck,Frothfeed]=intface(Eik,Eikpz,Eiknz,Cfisc,Cfiscpz,
Cfiscnz,Caisck,Caisckpz,Caiscknz,Blk,Vgs,Vgspz,Vgsnz,Vp,Vpnz,Vz,temp2)

global A G Nb Np Nc Db dt fvd Vg Vsl Qg Qb SGw Ul

Qsl=Qb; Qslnz=Qb;
Epz=sum(Eikpz);
e0=0.7;
Vsl=Qsl/A;
tempDb=Db;
tempNb=Nb;
Nb=1;
Db=(sum(Eikpz.*(tempDb.^(3/4)))/Epz)^(4/3);

Ektotal=fsolve('driftflx',e0);
Ek=(Ektotal/Epz).*Eikpz;
while Ek==[],
       Vsl=Vsl+1e-3;
       Ektotal=fsolve('driftflx',e0);
       Ek=(Ektotal/Epz).*Eikpz;
end

if Ektotal >= 1
       error('The conditions for two solutions to drift flux model have been
       exceeded');
```

```
end
e0=Ektotal;
Ek(Ek<0)=zeros(size(Ek(Ek<0)));
Db=tempDb;
Nb=tempNb;

VgtimesMa=zeros(Nb*Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
      VgtimesMa(Np*(k-1)+i,j)=Vgs(k).*Caisck(Np*(k-1)+i,j);
    end
  end
end

dCfsc=(-(Qg/sum(Eik))*Cfisc + (sum(Vgs.*Eik)*A/sum(Eik))*Cfiscnz +
(Vpnz.*Cfiscnz)*A + (Qg-Qsl+sum(Vgspz.*Eikpz)*A*
(1-sum(Eikpz))/sum(Eikpz))*Cfiscpz - (sum(Vgspz.*Eikpz)*A/sum(Eikpz))*Cfisc -
(Vp.*Cfisc)*A)/Vz;

Cfsc=Cfisc + dt*dCfsc;
Mfsc=Cfsc*Vz;

for i=1:Np
  for j=1:Nc
    for k=1:Nb
      Masck(Np*(k-1)+i,j)=(Ek(k).*Caisckpz(Np*(k-1)+i,j)*Vz)./(Eikpz(k));
    end
  end
end
Casck=Masck/Vz;

Mfsc(Mfsc<0)=zeros(size(Mfsc(Mfsc<0)));
Masck(Masck<0)=zeros(size(Masck(Masck<0)));

V=Vg/sum(Eik);
for i=1:Np
 for j=1:Nc
      tot=0;
      for k=1:Nb
             if Eik(k) > 0
                    tot= fvd(k)*Qg*Caisck(Np*(k-1)+i,j)./Eik(k) + tot;
             else
                    tot=0;
             end
      end
      temp2(i,j)=tot;
  end
end
Frothfeed=(Qg/sum(Eik))*Cfisc + temp2;

%==========================================================================================

% 14. Function to calculate the changes in air fraction, free solids
concentration and attached solids concentration at each time step along the
stabilized froth.

function [Ek,Mfsc,Masck]=sfzones(Eik,Eikpz,Eiknz,frE,Cfisc,Cfiscpz,
Cfiscnz,Caisck,Caisckpz,Caiscknz,Blk,Vgs,Vgspz,Vgsnz,Vp,Vpnz,Vz,Lzsf,sflambda)

global A Nb Np Nc Db kasck fvd dt maxBlk Vg Qg Qb

Qsl=Qb;Qslnz=Qb;
attachm=zeros(Nb*Np,Nc);
tattachm=zeros(Np,Nc);
```

```
D=zeros(Nb,1);
for k=1:Nb
 for j=1:Nb
  B=0;
  for l=1:Nb
   B=Eik(l).*Db(j).*Db(j).*Db(j)./(Db(l).*Db(l).*Db(l)) + B;
  end
  D(k)=sflambda(j,k).*Eik(k).*Eik(j)./B + D(k);
 end
end

Ap=zeros(Nb,1);
for k=2:Nb
 for j=1:k-1
  B=0;
  for l=1:Nb
       B=Eik(l).*Db(j).*Db(j).*Db(j).*Db(k-j).*Db(k-j).* Db(k-j)./
       (Db(l).*Db(l).*Db(l).*Db(k).*Db(k).*Db(k)) + B;
  end
  Ap(k)=sflambda(j,k-j).*Eik(j).*Eik(k-j)./B + Ap(k);
 end
end
Ap=0.5*Ap;

[detachm,tdetachm]=detment(Caisck,Eik,D,Ap);

Vpz=Vg/sum(Eikpz);
V=Vg/sum(Eik);
dVpz=-Vg/(sum(Eikpz)^2);
dV=-Vg/(sum(Eik)^2);

Ek=Eik + (dt/Lzsf)*(Eik.*dV + V*ones(Nb,1)).*(Eikpz-Eik) - dt*D + dt*Ap;

dCfsc=(-(Qg/sum(Eik))*Cfisc + (sum(Vgs.*Eik)*A/sum(Eik))*Cfiscnz +
(Vpnz.*Cfiscnz)*A + (Qg/sum(Eikpz))*Cfiscpz -
(sum(Vgspz.*Eikpz)*A/sum(Eikpz))*Cfisc -(Vp.*Cfisc)*A)/Vz - tattachm +
tdetachm;

dCasck=zeros(Nb*Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
       dCasck(Np*(k-1)+i,j)=(Caisck(Np*(k-1)+i,j).*dV.*(Eikpz(k)-Eik(k)) +
       V.*(Caisckpz(Np*(k-1)+i,j)-Caisck(Np*(k-1)+i,j)))/Lzsf +
       attachm(Np*(k-1)+i,j) - detachm(Np*(k-1)+i,j);
    end
  end
end

Cfsc=Cfisc + dt*dCfsc;
Casck=Caisck + dt*dCasck;
Mfsc=Cfsc*Vz;
Masck=Casck*Vz;

Ek(Ek<0)=zeros(size(Ek(Ek<0)));
Mfsc(Mfsc<0)=zeros(size(Mfsc(Mfsc<0)));
Masck(Masck<0)=zeros(size(Masck(Masck<0)));


%=========================================================================================

% 15. Function to calculate the changes in air fraction, free solids
concentration and attached solids concentration at each time step in the wash
water addition zone.
```

```
function [Ek,Mfsc,Masck]=wwzone(Eik,Eikpz,Eiknz,frE,Cfisc,Cfiscpz,Cfiscnz,
Caisck,Caisckpz,Caisclnz,Blk,Vgs,Vgspz,Vgsnz,Vp,Vpnz,Vz,Lt,wwlambda)

global A Nb Np Nc Db kasck fvd dt maxBlk Vg Qg Qb Qp

Qsl=Qb;Qslnz=Qp;

attachm=zeros(Nb*Np,Nc);
tattachm=zeros(Np,Nc);

D=zeros(Nb,1);
for k=1:Nb
 for j=1:Nb
  B=0;
  for l=1:Nb
   B=Eik(l).*Db(j).*Db(j).*Db(j)./(Db(l).*Db(l).*Db(l)) + B;
  end
  D(k)=wwlambda(j,k).*Eik(k).*Eik(j)./B + D(k);
 end
end

Ap=zeros(Nb,1);
for k=2:Nb
 for j=1:k-1
  B=0;
  for l=1:Nb
       B=Eik(l).*Db(j).*Db(j).*Db(j).*Db(k-j).*Db(k-j).* Db(k-j)./
       (Db(l).*Db(l).*Db(l).*Db(k).*Db(k).*Db(k)) + B;
  end
  Ap(k)=wwlambda(j,k-j).*Eik(j).*Eik(k-j)./B + Ap(k);
 end
end
Ap=0.5*Ap;

[detachm,tdetachm]=detment(Caisck,Eik,D,Ap);

Vpz=Vg/sum(Eikpz);
V=Vg/sum(Eik);
dVpz=-Vg/(sum(Eikpz)^2);
dV=-Vg/(sum(Eik)^2);

Ek=Eik + (dt/Lt)*(Eik.*dV + V*ones(Nb,1)).*(Eikpz-Eik) - dt*D + dt*Ap;

dCfsc=(-(Qg/sum(Eik))*Cfisc + (sum(Vgs.*Eik)*A/sum(Eik))*Cfiscnz +
(Vpnz.*Cfiscnz)*A + (Qg/sum(Eikpz))*Cfiscpz -
(sum(Vgspz.*Eikpz)*A/sum(Eikpz))*Cfisc -(Vp.*Cfisc)*A)/Vz - tattachm +
tdetachm;

dCasck=zeros(Nb*Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
       dCasck(Np*(k-1)+i,j)=(Caisck(Np*(k-1)+i,j).*dV.*
       (Eikpz(k)-Eik(k)) + V.*(Caisckpz(Np*(k-1)+i,j) -
       Caisck(Np*(k-1)+i,j)))/Lt + attachm(Np*(k-1)+i,j) -
       detachm(Np*(k-1)+i,j);
    end
  end
end

Cfsc=Cfisc + dt*dCfsc;
Casck=Caisck + dt*dCasck;
Mfsc=Cfsc*Vz;
Masck=Casck*Vz;

Ek(Ek<0)=zeros(size(Ek(Ek<0)));
Mfsc(Mfsc<0)=zeros(size(Mfsc(Mfsc<0)));
```

```
Masck(Masck<0)=zeros(size(Masck(Masck<0)));
```

% ==

% <u>16. Function to calculate the changes in air fraction, free solids concentration and attached solids concentration at each time step along the draining froth.</u>

```
function [Ek,Mfsc,Masck,frothYent,frothYfloat]=dfzones(Eik,Eikpz,Eiknz,frE,
Cfisc,Cfiscpz,Cfiscnz,Caisck,Caisckpz,Caisknz,Blk,Vgs,Vgspz,Vgsnz,Vp,
Vpnz,Vz,Lzdf,dflambda)

global A Nb Np Nc Db kasck fvd dt maxBlk Vg Qg Qp

Qsl=Qp;Qslnz=Qp;

attachm=zeros(Nb*Np,Nc);
tattachm=zeros(Np,Nc);

D=zeros(Nb,1);
for k=1:Nb
 for j=1:Nb
  B=0;
  for l=1:Nb
   B=Eik(l).*Db(j).*Db(j).*Db(j)./(Db(l).*Db(l).*Db(l)) + B;
  end
  D(k)=dflambda(j,k).*Eik(k).*Eik(j)./B + D(k);
 end
end

Ap=zeros(Nb,1);
for k=2:Nb
 for j=1:k-1
  B=0;
  for l=1:Nb
      B=Eik(l).*Db(j).*Db(j).*Db(j).*Db(k-j).*Db(k-j).*Db(k-j)./
      (Db(l).*Db(l).*Db(l).*Db(k).*Db(k).*Db(k))+B;
  end
  Ap(k)=dflambda(j,k-j).*Eik(j).*Eik(k-j)./B + Ap(k);
 end
end
Ap=0.5*Ap;

[detachm,tdetachm]=detment(Caisck,Eik,D,Ap)

Vpz=Vg/sum(Eikpz);
V=Vg/sum(Eik);
dVpz=-Vg/(sum(Eikpz)^2);
dV=-Vg/(sum(Eik)^2);

Ek=Eik + (dt/Lzdf)*(Eik.*dV + V*ones(Nb,1)).*(Eikpz-Eik) - dt*D + dt*Ap;

dCfsc=(-(Qg/sum(Eik))*Cfisc + (sum(Vgs.*Eik)*A/sum(Eik))*Cfiscnz +
(Vpnz.*Cfiscnz)*A + (Qg/sum(Eikpz))*Cfiscpz -
(sum(Vgspz.*Eikpz)*A/sum(Eikpz))*Cfisc -(Vp.*Cfisc)*A)/Vz - tattachm +
tdetachm;

dCasck=zeros(Nb*Np,Nc);
for i=1:Np
  for j=1:Nc
    for k=1:Nb
       dCasck(Np*(k-1)+i,j)=(Caisck(Np*(k-1)+i,j).*dV.*(Eikpz(k)-Eik(k)) +
       V.*(Caisckpz(Np*(k-1)+i,j)-Caisck(Np*(k-1)+i,j)))/Lzdf +
       attachm(Np*(k-1)+i,j) - detachm(Np*(k-1)+i,j);
```

```
    end
  end
end

Cfsc=Cfisc + dt*dCfsc;
Casck=Caisck + dt*dCasck;
Mfsc=Cfsc*Vz;
Masck=Casck*Vz;

Ek(Ek<0)=zeros(size(Ek(Ek<0)));
Mfsc(Mfsc<0)=zeros(size(Mfsc(Mfsc<0)));
Masck(Masck<0)=zeros(size(Masck(Masck<0)));

frothYent=(Qg/sum(Eik))*Cfisc;
frothYfloat=zeros(Np,Nc);
for i=1:Np
for j=1:Nc
total=0;
for k=1:Nb
 if frE(k) > 0
      total=(Qg*(sum(Eik)-Eik(k)+frE(k))*fvd(k)/frE(k)) *
      Caisck(Np*(k-1)+i,j)./sum(Eik)+total;
 else
      total=0;
 end
end
frothYfloat(i,j)=total;
end
end
```