Simulation of water and nutrient uptake by plant roots

Pierre-Henri Tournier

Laboratoire Jacques Louis Lions

December 7, 2012

Fourth Tutorial and Workshop on FreeFem++

Goals

- Simulate water movement in soil and water uptake by plant roots, together with the transport and uptake of nutrients.
- Explicitly take into account the geometry of a root system.
- Study how water and nutrient uptake is affected by the type and shape of the root system.

Outline

1. Richards equation
2. Water flow: uptake and transport in the root system
3. 1D representation of the root system
4. The sink term in the 3D model
5. Coupling of the 1D and 3D systems
6. Some numerical results
7. Nutrient uptake
8. Perspectives

Richards equation

The Richards equation represents the movement of water in unsaturated soils. It is obtained by combining Darcy's law with the continuity equation:

$$\begin{cases} \dfrac{\partial \theta(h)}{\partial t} = -\nabla . \vec{q} + S \\ \vec{q} = -K(h)\nabla(z + h). \end{cases}$$

- h is the matric head.
- $\vec{q}$ is the Darcy flux.
- $\theta(h)$ is the volumetric water content.
- $K(h)$ is the hydraulic conductivity.

The $\theta(h)$ and $K(h)$ relationships are given by empirical models whose parameters depend on the soil physical properties. We use the Brooks-Corey model:

$$\Theta(h) := \frac{\theta(h) - \theta_m}{\theta_M - \theta_m} = \left[\frac{h}{h_b}\right]^{-\lambda} := \begin{cases} \left(\frac{h}{h_b}\right)^{-\lambda} & \text{for} \quad h \leq h_b \\ 1 & \text{for} \quad h \geq h_b. \end{cases}$$

$$K(h) = K_s \left[\frac{h}{h_b}\right]^{-\lambda e(\lambda)} \quad \text{with } e(\lambda) := 3 + \frac{2}{\lambda}.$$

- θ_M is the saturated water content.
- θ_m is the residual water content.
- K_s is the saturated hydraulic conductivity.
- h_b is the bubbling pressure head.
- λ is the pore size distribution index.

We introduce the Kirchhoff transformation κ which enables us to eliminate the nonlinearity in the diffusion term:

$$\kappa : h \rightarrow u := \int_0^h K(p)dp.$$

The water content as a function of u is denoted by

$$M(u) := \theta(\kappa^{-1}(u)).$$

Using the chain rule, we have

$$\nabla u = K(h)\nabla h.$$

Thus, the Richards equation reads:

$$\frac{\partial M(u)}{\partial t} = \nabla.(\nabla u + K(\kappa^{-1}(u))\nabla z) + S.$$

In order to solve the Richards equation numerically

$$\frac{\partial M(u)}{\partial t} = \nabla.(\nabla u + K(\kappa^{-1}(u))\nabla z)$$

we use backward Euler together with the first-order Taylor series approximation of $M(u)$. This leads to the following Newton-like iteration:

$$\begin{aligned} & M'(u^m_{t_{i+1}})\frac{(u^{m+1}_{t_{i+1}} - u^m_{t_{i+1}})}{\Delta t} + \frac{M(u^m_{t_{i+1}}) - M(u_{t_i})}{\Delta t} \\ = \quad & \nabla.(\nabla u^{m+1}_{t_{i+1}} + K(\kappa^{-1}(u^m_{t_{i+1}}))\nabla z). \end{aligned}$$

The Brooks-Corey model gives us $M(.), M'(.)$ and $K(\kappa^{-1}(.))$ in explicit form.

For a cylindrical root segment and according to the Ohm's law analogy, the volumetric radial water flow between the soil-root interface and the root xylem can be written as

$$J_r = K_r s_r (h_s - h_r).$$

- K_r is the radial conductivity.
- $s_r = 2\pi r l$ is the root-soil interface area.
- h_s is the soil water potential at the root surface.
- h_r is the xylem water potential.

Water moves through the root system due to the transpiration-cohesion-tension mechanism.
The longitudinal water flow in the xylem is defined as:

$$J_x = -K_x \frac{dh_r}{dl},$$

where K_x is the xylem conductance.

1D representation of the root system

The root system geometry is represented as a series of interconnected nodes, forming a network of root segments.

In this example, a code developed at BOKU is used that simulates the growth of the root system of a 20-days-old maize plant and outputs the corresponding geometry.

The radial and longitudinal flows equations

$$J_r = K_r s_r (h_s - h_r), \quad J_x = -K_x \frac{dh_r}{dl}$$

can be used to define the following water mass balance equation for a given root node i of parent node $p(i)$ in the tree-like structure:

$$-K_{x,i:p(i)} \frac{h_{r,p(i)} - h_{r,i}}{l_{i:p(i)}} = - \sum_{j \in childs(i)} K_{x,i:j} \frac{h_{r,i} - h_{r,j}}{l_{i:j}}$$

$$+K_{r,i:p(i)} 2\pi r_{i:p(i)} l_{i:p(i)} \frac{(h_{s,i} - h_{r,i}) + (h_{s,p(i)} - h_{r,p(i)})}{2},$$

The xylem water potential vector $(h_{r,i})_i$ is then solution of a linear system, with the right-hand side containing the soil factors represented by the $h_{s,i}$.

1D root system

FreeFem++ code

```
real[int,int] A(nnumpt,nnumpt);
real[int] b(nnumpt);
real[int] sol(nnumpt);
...
b[0] = -0.00015; // transpiration rate
for (int i=0; i < nnumpt; i++) // for each node i
{
    int npere = pere(i,0);
    // the soil factors are in the right-hand side:
    b[i] = radialcond[i]*surf[i]*(pratnodetab(0,i)+pratnodetab(0,npere))/2.;
    if (i == 0) // root collar
        ...
    else
    {
        // xylem flows:
        A(i,npere) += -xylemcond(i,npere)/length(i,npere);
        A(i,i) += xylemcond(i,npere)/length(i,npere);
        for (int j = 0; j<nbfils[i]; j++)
        {
            int nfils = fils(i,j);
            A(i,nfils) += -xylemcond(i,nfils)/length(i,nfils);
            A(i,i) += xylemcond(i,nfils)/length(i,nfils);
        }
        // radial uptake flow for segment (i,p(i)):
        A(i,npere) += radialcond[i]*surf[i]/2.;
        A(i,i) += radialcond[i]*surf[i]/2.;
    }
}
matrix M = A;
set (M,solver = sparsesolver); // M is sparse
sol = M^-1*b; // solve the linear system
```

The sink term in the 3D model

In order to take the radial water uptake flows into account in the 3D model, a sink term in Richards equation is defined in the domain.
Since we want the sink term to precisely match the volumetric water uptake flows in the 3D space, we define a characteristic function of the root system, representative of its geometry.
Then, we use this characteristic function to define the sink term.

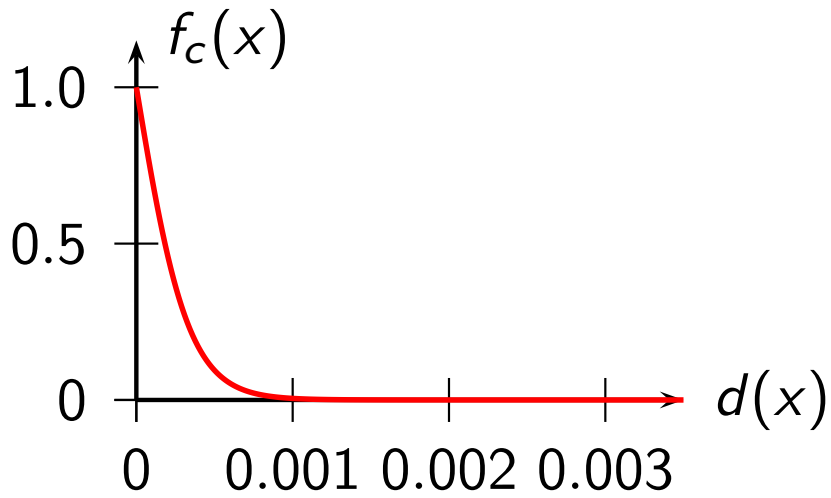
The function f_c representative of the geometry of the root system in the domain is constructed as follows:

- For a point x in the domain the distance d from x to the root is computed:

$$d(x) = \min_{s \in \Sigma} d_s(x)$$

with Σ the set of root segments in the 1D network.

- $f_c(x) = f_d(d(x))$
$= 1 - \tanh\left(\frac{3d(x)}{\epsilon}\right)$
with $\epsilon = 0.001$ m.



The sink term in the 3D model

FreeFem++ code

```
macro fd(dist,reps) (1.-tanh(3.*dist/reps))//
...
func real ddist(int i, real reps)  // compute fc at vertex i of the mesh
{
   real x1 = Th(i).x, y1 = Th(i).y, z1 = Th(i).z;  // coordinates of vertex i
   real res  = -1.e+30;  //current value of fc at vertex i
   ...
   for (int k=0;k<nbr;k++)  // loop on the root segments
   {
      xa = px[k]; ya = py[k]; za = pz[k];  // coordinates of the nodes (p,q) of segment k
      xb = qx[k]; yb = qy[k]; zb = qz[k];
      vx = xb-xa; vy = yb-ya; vz = zb-za;
      //compute the squared distance of vertex i from root segment k
      hypera = ((xb-xa)*(x1-xa)+(yb-ya)*(y1-ya)+(zb-za)*(z1-za) < 0)*((x1-xa)^ 2+(y1-ya)^ 2
      hyperb = ((xa-xb)*(x1-xb)+(ya-yb)*(y1-yb)+(za-zb)*(z1-zb) < 0)*((x1-xb)^ 2+(y1-yb)^ 2
      og = (hypera > hyperb ? hypera : hyperb);

      pvx = (y1-ya)*vz-(z1-za)*vy;
      pvy = (z1-za)*vx-vz*(x1-xa);
      pvz = (x1-xa)*vy-vx*(y1-ya);
      og2 = (pvx^ 2+pvy^ 2+pvz^ 2)/(vx^ 2+vy^ 2+vz^ 2);

      og = max(og,og2);
      og = fd(sqrt(og),reps);
      res = max(res,og); // update the value of fc
   }
return res;
}
```

The sink term in the 3D model

FreeFem++ code

```
fespace Uh(Th,P1);
Uh fc;
...
// compute fc on the current mesh
int nb = floor(Th.nv*1./mpisize)+1;
// simple example of loop parallelization
for (int i=nb*mpirank;i<min(Th.nv-1,nb*(mpirank+1)-1)+1;i++)
  fc[][i] = ddist(i,reps); // compute fc at vertex i
// a simple gather on processor 0
mpiRequest[int] rq (mpisize-1);
if (mpirank == 0)
{
  for (int i=1;i<mpisize;i++)
  {
    real[int] tmp(min(Th.nv-1,nb*(i+1)-1)-nb*i+1);
    Irecv(processor(i,rq[i-1]), tmp);
    mpiWait(rq[i-1]);
    for (int j = 0;j < min(Th.nv-1,nb*(i+1)-1)-nb*i+1;j++)
      fc[][nb*i+j] = tmp[j];
  }
}
else
{
  Isend(processor(0,rq[mpirank-1]), fc[](nb*mpirank:min(Th.nv-1,nb*(mpirank+1)-1)));
  mpiWait(rq[mpirank-1]);
}
...
```

The sink term in the 3D model

FreeFem++ code

```
//mesh adaptation

//in 2D:
Th=adaptmesh(Th,fc,hmax=hmax,err=etol,iso=0);

//in 3D:
real[int] MSHdoptions(4);
int[int] MSHloptions(7);
MSHdoptions(0)=hmin;
...
real[int] M = mshmet(Th,fc,aniso=1,loptions=MSHloptions,doptions=MSHdoptions);
Th = mmg3d(Th,metric=M,opt="-O 1");
```

The sink term in the 3D model

Let us consider the case of a cylindrical root segment, formed by the nodes i and j. The corresponding radial flow is

$$J_r = K_r 2\pi r_r l_r \frac{(h_{s,i} - h_{r,i}) + (h_{s,j} - h_{r,j})}{2}.$$

Moreover, we have

$$\int_\Omega f_c r dr d\theta dl = 2\pi l_r \int_0^{+\infty} r f_d(r) dr = 2\pi l_r T_r.$$

We then define the corresponding sink term as

$$S = f_c \frac{K_r r_r}{T_r} h_l,$$

where h_l only depends on l and linearly interpolates $h_s - h_r$ along the segment.

Thus, we have

$$\int_\Omega S = \int_\Omega f_c \frac{K_r r_r}{T_r} h_l = K_r 2\pi r_r \int h_l dl = J_r.$$

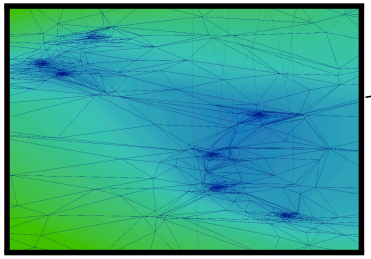
Coupling of the two models

The coupling of the 1D and 3D models consists in iteratively solving the two problems until convergence. Let $h_s^{t_i}$ be the soil matric potential distribution at time t_i, h_s^k and h_r^k the soil and xylem matric potentials at inner iteration k and time t_{i+1}.

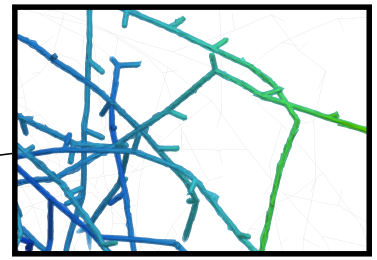
1. $h_s^0 = h_s^{t_i}$.
2. Solve the 1D linear system with soil factors h_s^k, obtain h_r^k.
3. Compute S using h_s^k and h_r^k.
4. Perform an inner iteration of Richards equation, obtain h_s.
5. $h_s^{k+1} = h_s^k + \alpha_k(h_s - h_s^k)$, where $0 < \alpha_k \leq 1$ is an under-relaxation parameter that ensures convergence of the system.
6. If $||h_s - h_s^k|| > \epsilon$, go to 2 with $k := k + 1$.

Overview

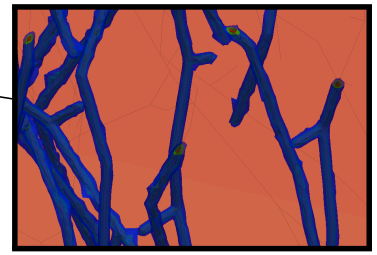
xylem matric potential h_r defined on the 1D network



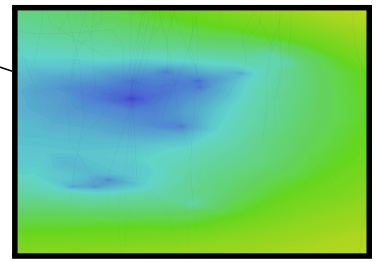
slice through the mesh, showing adaptive refinement relative to the characteristic function



sink term S in the domain, defined on the characteristic function



isosurfaces 0.3, 0.5 and 0.7 of the characteristic function f_c



slice of the solution h_s to Richards equation in the domain

Numerical results

- 1D root system from RootBox: ~10000 segments.
- Finite element mesh: ~3M vertices, ~13M tetrahedra.
- Adaptive mesh refinement relative to the characteristic function with MMG3D.
- Schwarz domain decomposition method: partition of the domain into 16 subdomains.

Domain decomposition

FreeFem++ code

```
//Build the overlapping partition:
BuildPartitioning(sizeoverlaps,mesh3,Th,Thi,aThij,RAS,pii,jpart,comm,vdebug)
BuildTransferMat(ipart,mesh3,Pk,1,[0],Thi,Whi,Whij,Thij,aThij,Usend,Vrecv,jpart,vdebug)
...
//Build the linear system corresponding to
//an inner iteration for Richards equation on Thi:
Whi muprimeciti = muprime(citi)/delta;
Whi Muciti = Mu(citi)/delta;
Whi Muu0i = Mu(u0i)/delta;
Whi krmuciti = Ks*krmu(citi);
Whi scm = muprimeciti*citi-Muciti+Muu0i-upti;

varf vPb(U,V)= int3d(Thi)(muprimeciti*U*V)
+int3d(Thi)(Ks*(dx(U)*dx(V)+dy(U)*dy(V)+dz(U)*dz(V)))
+on(10,U=0);

varf vPbb(U,V) = int3d(Thi)(scm*V)
-int3d(Thi)(krmuciti*dz(V))
+on(10,U=0);

matrix Ai = vPb(Whi,Whi,solver=sparsesolver);
real[int] Bi=vPbb(0,Whi);
...
DMMDeffuncAndGlobals(Richards,comm,jpart,Whi,VhC,1,Ai,vPbC,onG10,Pii,Usend,Vrecv,[0])
//Call the DDM solver: gmres with a coarse grid preconditioner
RichardsDDMSolver(Bi,uii,vii,3,ddmeps,vdebug)
```

Nutrient uptake

Nutrient uptake is governed by the following mechanisms:

- Active uptake of nutrients by roots, described by Michaelis-Menten kinetics.
- Diffusion of nutrient ions in the soil solution.
  Dominant for phosphate.
- Transport of nutrients by mass flow.
  Dominant for nitrate.
- Adsorption of nutrient ions in the soil solid phase.
  Strong for phosphate, negligible for nitrate.

$$\frac{\partial}{\partial t}(\phi(c) + \theta c) = \nabla.(D\nabla c - \vec{q}c) + s_c.$$

- c is the nutrient concentration in the soil solution.
- $\phi(c)$ is an adsorption isotherm relating the amount of nutrient in the solid phase to the equilibrium concentration in the soil solution.
- θ is the volumetric water content.
- D is the diffusion coefficient of the nutrient in the soil solution.
- $\vec{q}$ is the Darcy flux.
- s_c represents sources/sinks. Here $s_c = h(c)$, corresponding to nonlinear Michaelis-Menten uptake kinetics.

Applying the backward Euler method, the convection-diffusion equation reads:

$$\frac{\phi(c_{t_{i+1}}) + \theta_{t_{i+1}} c_{t_{i+1}} - \phi(c_{t_i}) - \theta_{t_i} c_{t_i}}{\Delta t} = \nabla.(D\nabla c_{t_{i+1}} - \vec{q} c_{t_{i+1}}) + s_c.$$

Now, multiplying Richards equation by $c_{t_{i+1}}$ yields:

$$\frac{\theta_{t_{i+1}} c_{t_{i+1}} - \theta_{t_i} c_{t_{i+1}}}{\Delta t} = -c_{t_{i+1}} \nabla.\vec{q} + S c_{t_{i+1}}.$$

Substracting this equation from the previous one gives

$$\frac{\phi(c_{t_{i+1}}) + \theta_{t_i} c_{t_{i+1}} - \phi(c_{t_i}) - \theta_{t_i} c_{t_i}}{\Delta t} = \nabla.(D\nabla c_{t_{i+1}}) - \vec{q}.\nabla c_{t_{i+1}} - S c_{t_{i+1}} + s_c.$$

- Newton's method for nonlinear terms.
- Method of characteristics for the convective term.

How to simulate root growth with chemotropism: link the Matlab code RootBox with FreeFem++ by redirecting stdins and stdouts with pipes.

```
system("mkfifo pipe pipe2;exec 5<>pipe 6<>pipe2; FreeFem++ -v 0 pipe.edp < pipe > pipe2 &
matlab -nodesktop -nosplash -nodisplay -r  \"applygrowth\" < pipe2 > pipe");
```

FreeFem++ code: (pipe.edp)

```
...
Uh c; // concentration
...
while(xx != 1.e+30)
{
cin >> xx >> yy >> zz;
real cp = c(xx,yy,zz);
cout << cp << endl;
}
cout << "ok" << endl;
```

Matlab code: (applygrowth.m)

```
...
% R3 is the current 'test' direction
x_ = y.xT+dx*R3(1,1);
y_ = y.yT+dx*R3(2,1);
z_ = y.zT+dx*R3(3,1);

str = [num2str(x_), ' ', num2str(y_), ' '];
str = [str, num2str(z_), ' '];
s = input(str);
cost = -c*s + g*R3(3,1); % c, g are constants
...
ok = input('1.e+30 1.e+30 1.e+30','s');
quit;
```

Perspectives

- Simulate the growth of the root system as affected by different types of tropisms, chemotropism in particular.
- Represent the roots by their actual surface.
- Use the diffuse domain approach.
- Meshing.
- Sensitivity Analysis, shape optimization.