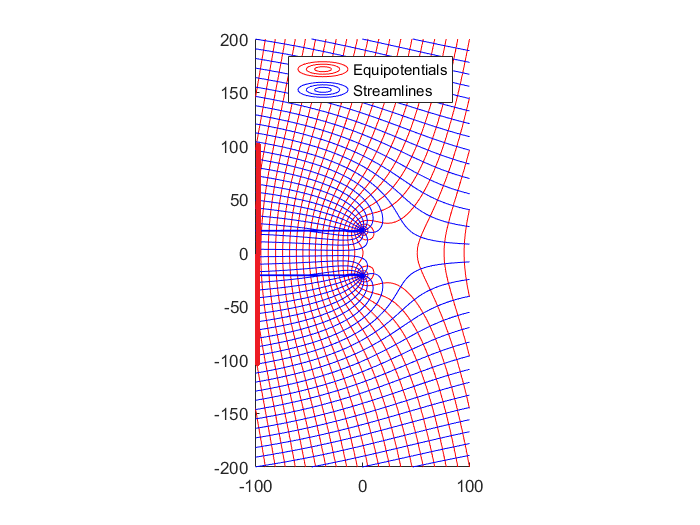
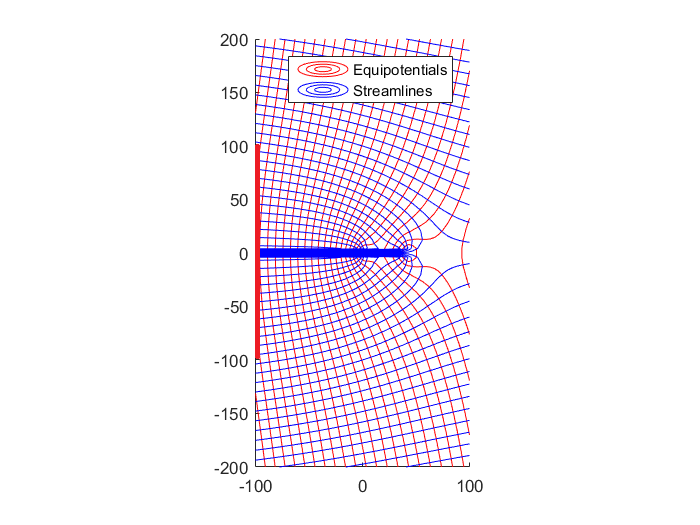


3. The in-line variation is superior because there is no possibility of contamination escaping the system. In the system with wells aligned on the y axis, contamination could escape between the wells.

4.Discussion: It requires less pumping to capture all of the contaminant when the in-line system is used. It appears that the streamline at –l + ib and –l – ib are not captured, but this is a result of the contouring routine’s resolution.

Figures:



Code:

ContoutMe\_flownet.m:

function [Grid] = ContourMe\_flow\_net(xfrom, xto, Nx, yfrom, yto, Ny, func,nint)

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

% ContourMe\_I(xfrom, xto, Nx, yfrom, yto, Ny, func) (01.23.09)

%

% Contour the imaginary part of the specified complex function.

%

% Arguments:

%

% xfrom starting x-value for the domain

% xto ending x-value for the domain

% Nx number of grid columns

%

% yfrom starting y-value for the domain

% yto ending y-value for the domain

% Ny number of grid rows

%

% func function to contour; must take one complex argument.

%

% Returns:

%

% Grid Ny x Nx matrix of values of func at the rid nodes.

%

% Example Usage:

%

% G = ContourMe\_I(1,2,11,1,2,11,@(z)Omega(1,-1,z));

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

Grid = zeros(Ny,Nx);

X = linspace(xfrom, xto, Nx);

Y = linspace(yfrom, yto, Ny);

for row = 1:Ny

for col = 1:Nx

Grid(row,col) = func( complex( X(col), Y(row) ) );

end

end

Bmax=max(imag(Grid));

Bmin=min(imag(Grid));

Cmax=max(Bmax);

Cmin=min(Bmin);

D=Cmax-Cmin;

del=D/nint;

Bmax=max(real(Grid));

Bmin=min(real(Grid));

Cmax=max(Bmax);

Cmin=min(Bmin);

D=Cmax-Cmin;

nintr=round(D/del);

figure;

hold on

contour(X, Y,real(Grid),nintr,'r');

contour(X, Y,imag(Grid),nint,'b');

legend('Equipotentials','Streamlines')

axis square

axis equal

%hold on

%contour(X, Y,real(Grid),nintr);

%contour(X, Y,imag(Grid),nint);

%axis equal

findRoots.m

f = @(Qx0,b,l,d) - Qx0\*b + Qx0\*d\*(2\*3.1415- atan((b-d)/l) - atan((d+b)/l)) ;

Qx0=6;

b=100;

l=100;

fun = @(d) f(Qx0,b,l,d);

xmin = 0;

xmax = 100;

fplot(fun, [xmin, xmax] )

d= fzero(fun,1)

Omeg\_flow.m:

function [ Omega ] = Omega\_Uniformflow (W0,z)

Omega = -W0\*z/2

end

Omega\_total.m:

function [ Omega ] = Omega\_total( z,Qx0,Q, z1,z2,rw)

Omega= -Qx0 \* z + Omega\_well(z,z1,rw,Q) +Omega\_well(z,z2,rw,Q);

end

Omega\_well.m:

function [ Omega ] = Omega\_well(z,z0,rw,Q)

rsq=(z-z0)\*conj(z-z0);

if rsq>rw^2

Omega=Q/(2\*pi)\*log(z-z0);

else

Omega = 0;

end

wells\_perp\_ruinfile.m:

;

Qx0 = 1 ;

d=;

Q= d\*2\*3.14\* Qx0;

l = 10;

rw = 0.2; %m

b= 10 ;

z1= i\*d;

z2= i\*-d;

ContourMe\_flow\_net(-l,5,500,-(b+5),(b+5),500,@(z)Omega\_total(z,Qx0,Q,z1,z2,rw),30);

Wells\_inline\_runfile.m:

Q = 100;

Qx0 = 1 ;

d= Q /(pi\*Qx0);

l = 100;

rw = 0.2; %m

a= .6;

b= 50 ;

z1= 0;

z2= 2\*d;

ContourMe\_flow\_net(-l,l,50,-(b+5),(b+5),50,@(z)Omega\_total(z,Qx0,Q,z1,z2,rw),30);