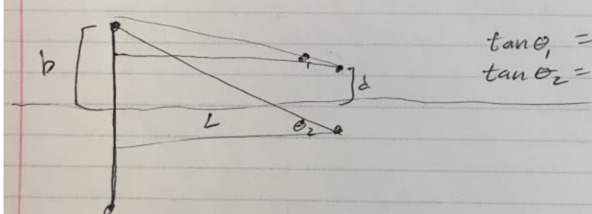
1) conditions: One stagnation point, all contaminant captures

for one stagnation point $d = \frac{Q}{2\pi Q_{xo}}$

To capture all contaminant, $\Psi = 0$ at $-L \pm ib$



$\tan\theta_1 = b-d/L$
$\tan\theta_2 = b+d/L$

$$\Psi = -Q_{xo} b + \frac{Q}{2\pi}\left(2\pi - \arctan\frac{b-d}{L} - \arctan\frac{b+d}{L}\right) = 0$$

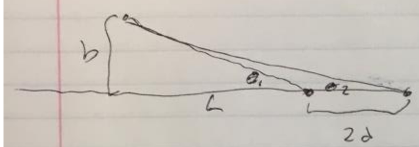
$Q = 2\pi Q_{xo} d$, solve for d using matlab

* more detailed derivation in original submission *

2)

condition: all contaminant captured

$\Psi = 0 = -L \pm i d$



$$\cancel{\Psi = -Q_{x0}b - \frac{Q}{2\pi}\left(2\pi - \arctan\frac{b}{L} - \right.}$$

$$\cancel{\Psi = -Q_{x0}b + \frac{Q_{x0}}{2\pi}\left(2\pi - \arctan\frac{b}{L} - \arctan\frac{b}{L+\dots}\right.}$$

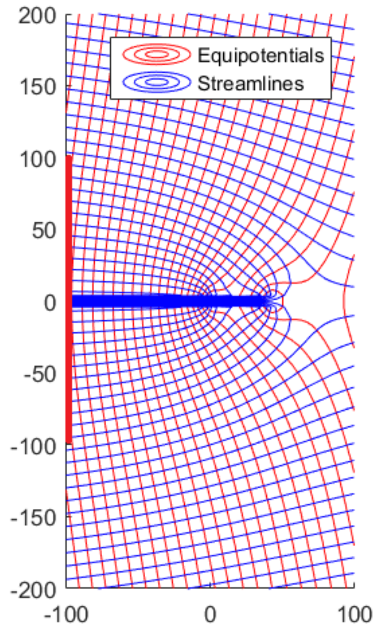
$$\Psi = 0 = -Q_{x0}b + \frac{Q}{2\pi}\left(2\pi - \arctan\frac{b}{L} - \arctan\frac{b}{L+2d}\right)$$

$$\Rightarrow Q = Q_{x0}b\, 2\pi\left(2\pi - \arctan\frac{b}{L} - \arctan\frac{b}{L+2d}\right)^{-1}$$

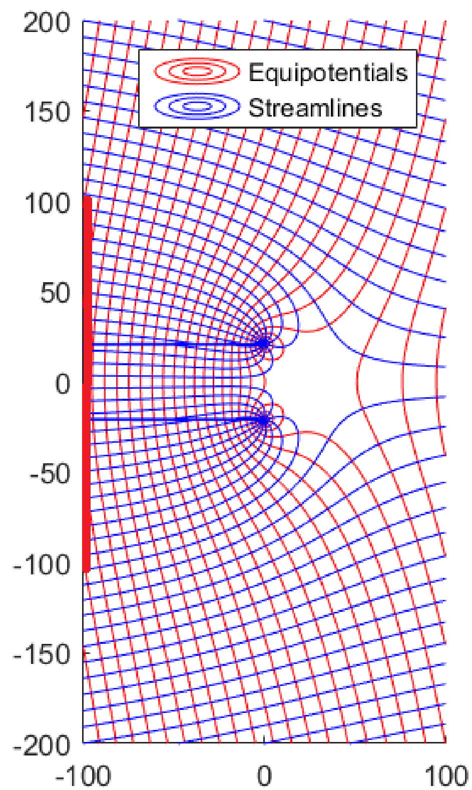
3. The in-line variation is superior because there is no possibility of contamination escaping the system. In the system with wells aligned on the y axis, contamination could escape between the wells.

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:



Equipotentials
Streamlines
200
150
100
50
0
-50
-100
-150
-200
-100
0
100



Equipotentials
Streamlines
200
150
100
50
0
-50
-100
-150
-200
-100
0
100

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);
```