Narrative:

In this assignment the effect of inhomogeneities on wells was investigated. The quantity of interest in this investigation was the maximum discharge of a well in an inhomogeneity. The maximum discharge of a well was calculated by setting the potential at the well's screen equal to zero, indicating maximal drawdown.

The complex potential for a well in an inhomogeneity is

$$\Omega = \frac{Q}{2\pi}\ln(z-z_w) + \frac{k_1-k}{k_1+k}\frac{Q}{2\pi}\ln\left[\left(z-\frac{R^2}{\bar{z}_w}\right)\frac{\bar{z}_w}{-R}\right] + \frac{k_1}{k}C \qquad z\bar{z} \leq R^2$$

At maximum discharge, the potential at the wells screen is zero, so the real part of the above equation, evaluated at z = zw + rw (position of the well + radius of the well), can be solved from Q_max. The same method will apply in the case that there is uniform flow, but the complex potential term must include the expression for uniform flow.

Q_max depends on C, the constant. C is found by using the far field condition (a known potential at a faraway point) and the complex potential for a well in an inhomogeneity, outside the inhomogeneity:

$$\Omega = \frac{2k}{k_1+k}\frac{Q}{2\pi}\ln(z-z_w) + \frac{k_1-k}{k_1+k}\frac{Q}{2\pi}\ln\frac{z}{R} + C \qquad z\bar{z} > R^2$$

This equation is evaluated with z = z far field, and the real part is equal to the far field potential. As described, the two equations have 2 unknowns, Q_max and C, which can be solved for algebraically, giving the maximum discharge of a well in an inhomogeneity.

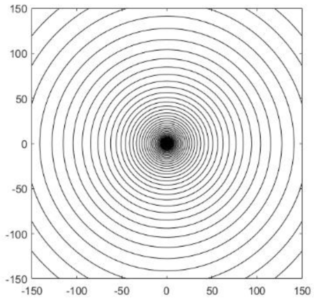
Using this method, the maximum discharge of various cases was investigated and the detailed results are shown below. Generally, wells in inhomogeneities of high hydraulic conductivity drastically increased the maximum well discharge relative to a well in a homogenous aquifer. Wells in low hydraulic conductivity zones had dramatically lower maximum discharge. The closer a well is to the center of an inhomogeneity, the greater the observed effect of the inhomogeneity on the maximum pumping rate. The addition of Uniform flow had little influence on the maximum pumping rate at these wells.

Case 1, no uniform flow

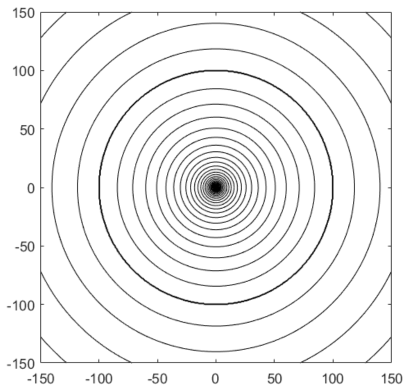
	Q_max, m/d
K = k1	1.27 * 10^3
K1 = 10* k	5.31* 10^4
K= 10* k1	72

Potential contours:

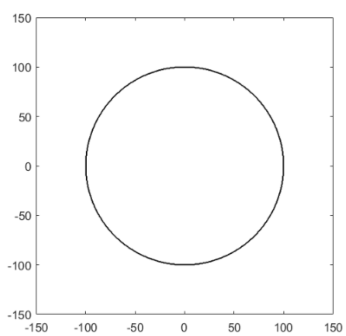
K=k1



K1>k



k>k1



2.

For k1>>k

	Q max, m/d
Zw=0	5.31* 10^4
Zw=50	4.84 * 10^3
Zw=75	4.14 * 10^4

For k1 << k

	Q max, m/d
Zw=0	72
Zw=50	73
Zw=75	75

The maximum discharge decreases slightly as the well is placed further from the center of the inhomogeity.

3.

For k1>k

Radius of gravel pack, m	Qmax, m/d
.5	2.19 * 10^4
1	2.37 * 10^4
1.5	2.49 * 10^4
3	2.73 * 10^4
5	2.94 * 10^4

The maximum discharge of the well increases somewhat as the size of the gravel pack around it increases.

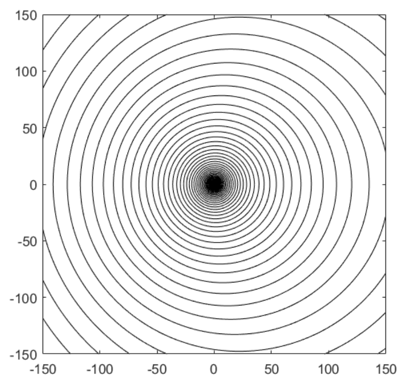
Case 2, uniform flow left to right

	Q_max, m/d
K = k1	1.27 * 10^3

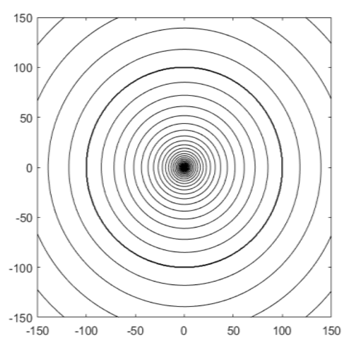
K1 = 10* k	5.79 * 10^4
K= 10* k1	6.79

Head contours:

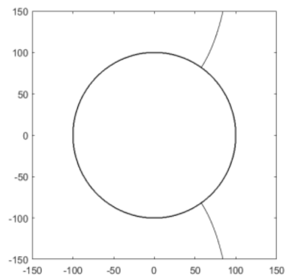
K=k1



K1>k



k>k1



2.

For k1>>k

	Q max, m/d
Zw=0	5.79 * 10^4
Zw=50	5.25 * 10^4
Zw=75	4.48 * 10^4

For k1 << k

	Q max, m/d
Zw=0	6.79
Zw=50	6.19
Zw=75	6.0

3.

For k1>k

Radius of gravel pack, m	Qmax, m/d
.5	2.39 * 10^4
1	2.59 * 10^4
1.5	2.72 * 10^4
3	2.92* 10^4
5	3.212 * 10^4

Code:

Main.m:

```
%case 1: no uniform flow

    %Parameters
    k = 10;
    k1= 10; %m/d
    zw =0;
    rw = 0.05;
    R = 100; %m
    Rinf = 10*R;
    Qx0 = 0; %No uniform flow
    PhiInf = .5 * k * 20*20;
    z = zw+rw;
    %calculate maximum discharge
    Q_max = ((Qx0*(z))*(2*k1/(k1+k))+ (-Qx0* (Rinf -(k1-
k)*R*R/((k1+k)*Rinf)))- (k1/k)*real(PhiInf))/
real((1/(2*pi))*log(z-zw)+ ((k1-k)/(k1+k))*(1/(2*pi)) *
log((conj(zw)*(z)/-R) + R)  - (2*k/(k1+k))* (1/(2*pi))*log(Rinf
- zw) -((k1-k)/(k1+k))*(1/(2*pi))*log(Rinf/R));
    %calculate constant
    c = real(PhiInf +  (2*k/(k1+k))* (Q/(2*pi))*log(Rinf - zw)
+((k1-k)/(k1+k))*(Q/(2*pi))*log(Rinf/R)+ Qx0* (Rinf -(k1-
k)*R*R/((k1+k)*Rinf)));


    %Calculate Q max if there was no inhomogeneity
    Q_noInhomogeneity = -PhiInf  /real(   (1/(2*pi))*(log(zw+rw-
zw) -log(Rinf - zw))  );
```

```
    %Contour the real potential
    ContourMe_R_int(-150,150,500, -150,150,500,
@(z)real(Omega_total(Qx0, z, k1,k,R, c,Q_max,zw)),60);




%Case 2, uniform flow



    %Parameters
    k = 10;
    k1= 100; %m/d
    zw =0;
    rw = 0.05;
    R = 5; %m
    Rinf = -10*R;
    Qx0= .5*k*(21*21 - 19*19)/(2 * abs(Rinf)) ;%with uniform
flow
    PhiInf = .5 * k * 21*21;
    z = zw+rw;

    Q_max = ((Qx0*(z))*(2*k1/(k1+k))+ (-Qx0* (Rinf -(k1-
k)*R*R/((k1+k)*Rinf)))- (k1/k)*real(PhiInf))/
real((1/(2*pi))*log(z-zw)+ ((k1-k)/(k1+k))*(1/(2*pi)) *
log((conj(zw)*(z)/-R) + R)  - (2*k/(k1+k))* (1/(2*pi))*log(Rinf
- zw) -((k1-k)/(k1+k))*(1/(2*pi))*log(Rinf/R))

    c = real(PhiInf +  (2*k/(k1+k))* (Q/(2*pi))*log(Rinf - zw)
+((k1-k)/(k1+k))*(Q/(2*pi))*log(Rinf/R)+ Qx0* (Rinf -(k1-
k)*R*R/((k1+k)*Rinf)));




    ContourMe_R_int(-150,150,500, -150,150,500,
@(z)real(Omega_total(Qx0, z, k1,k,R, c,Q_max,zw)),60);
```

```
function [ Omega ] = Omega_total(Qx0, z, k1,k,R,C,Q,zw )
%UNTITLED4 Summary of this function goes here
%   Detailed explanation goes here

rsq=(z)*conj(z);
if rsq>R^2
    Omega = Omega_outside(Qx0, z, k1,k,R, C,Q,zw);
else
    Omega = Omega_inside(Qx0, z, k1,k,R, C,Q,zw);

end
```

```
function [ Omega ] = Omega_outside(Qx0, z, k1,k,R,C,Q ,zw)
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here
Omega = -Qx0*(z-((k1-k)/(k1 +k))*(R*R)/z) +
(2*k/(k1+k))*(Q/(2*pi))*log(z-zw) + ((k1-
k)/(k1+k))*(Q/(2*pi))*log(z/R) + real(C);
end
```

```
function [ Omega ] = Omega_inside(Qx0, z, k1,k,R,C,Q,zw )
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here

Omega =( -2*k1/(k1 + k))*Qx0*z +(Q/(2*pi))*log(z-zw)+ ((k1-
k)/(k1+k))*(Q/(2*pi)) * log(R - z* conj(zw)/R)
+(k1/k)*real(C);
end
```

ContourMe_R_int.m

```
function [Grid] = ContourMe_R_int(xfrom, xto, Nx, yfrom, yto,
Ny, func,nint)
%============================================================
===========
% ContourMe(xfrom, xto, Nx, yfrom, yto, Ny, func)
(01.23.09)
%
%   Contour the real 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(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
contour(X, Y, real(Grid),nint, 'k');
axis equal
```