Texas Tech University

Department of Civil Engineering

Water Resources Center

Alireza Asadi and Vahid Salahi

November 2017

Assignment 6

Summary:

This exercise evaluates the allocation of areal distribution of pumping based on total demands for consumption and availability of water for the certain aquifer. The model was created with linear programming in aquifer management to minimize the cost of the pumping and conveyance to meet the water requirement. Constraints employed for this problem designed to prevent lake water from encroaching into the aquifer. The output of the optimized model indicated that there is no unique answer for location of the pumps and amount of pumping for each one. The cost can be minimized for different choices.

Introduction:

Globally, groundwater is considered as a sustainable source of fresh water supply and plays an important role in the development and management of water resources. However, expansion of water withdrawals from aquifers to provide the different uses can lead to deterioration of water quality, especially for those neighboring lakes. To minimize the effects of human on its environment, the optimization problems seem to be more pragmatic in the sense of making decision in order to achieve specified objectives, subjected to specified constraints. Linear programming (LP) formulation as a tool to find the solution for decision making problems are commonly used in variety of fields such as water resource management systems not only for the problem which there is a linear relation among its constraints and goal but also for nonlinear problems too since its outputs are more tangible from a point of data interpretation. In this case study it is tried to use the LP solver tools to minimize the cost of the pumping plus transmission of the water from one cell to a certain location in the aquifer in account of satisfy its constrains (Both constraints and objective are linear).

Problem Statement:

A 10 km ×10 km square shape (25 cells of 2km × 2km size) aquifer is shown in Figure 1. Aquifer has impervious boundaries on three sides and a lake is on the forth side. Since the lake water quality is low, keeping the water level of the aquifer in a certain level should be taken into consideration to prevent water encroaching from the lake into the aquifer. Therefore, water level at the distance of 1 km and 3 km from the lake should be kept as +0.64 m and +0.95 m respectively.

Precipitation with a rate of N=100 mm/yr is replenishing the aquifer uniformly over the entire area. The entire replenishment drains through the aquifer into the lake. Aquifer is homogenous and has transmissibility of T=1000 m^2/day.

Solution:

This project contains three parts: 1. to calculate the initial value of the head in each cell when there is no pump included to the system, 2. considering a pump for each cell individually and find the influence coefficients for all the cells corresponding to cell containing the pump. Finally, evaluate the objective function with its constraints to find the best allocation of pumps to provide water requirement for the investigating aquifer.

First, before to start doing the step 1, the multiple cell model of aquifer is produced to determine its response to activities and make comparisons among the deferent plan proposed for the area.

Multi cell approach is the numerical scheme that divides the lump aquifer to small cells and uses the partial differential equation to water balance model to correlate each cell to other one. The formulation of the water balance for rectangular cell i,j for balance period between t and t+Δt can be as the following way:

$$\Delta t\left(Q_x|_{i-\frac{1}{2},j} - Q_x|_{i+\frac{1}{2},j} + Q_y\big|_{i,j-\frac{1}{2}} - Q_y\big|_{i,j+\frac{1}{2}} + R_{i,j} - P_{i,j} + N_{i,j}\right) = S_{i,j}\Delta x_i \Delta y_j\left(\emptyset_{i,j}^{t+\Delta t} - \emptyset_{i,j}^{t}\right) \quad (1)$$

Where

Q_x and Q_y are the total rates of flow at time t through cell boundaries

$R_{i,j} + N_{i,j}$ is the total recharge rate (artificial and natural, respectively) in cell i,j during Δt

$P_{i,j}$ is the total pumping rate in cell i,j during Δt

$S_{i,j}$ is the average aquifer storability in cell i,j and

$\Phi^{t}_{i,j}$ is the piezometric head in cell i,j at time t.

If we use Darcy law for porous media into the equation the equation may be rewritten:

$$\Delta t\left(T_{i-\frac{1}{2},j}\Delta y_j \frac{\emptyset_{i-1,j}^{t} - \emptyset_{i,j}^{t}}{(\Delta x_i - \Delta x_{i-1})/2} - T_{i+\frac{1}{2},j}\Delta y_j \frac{\emptyset_{i+1,j}^{t} - \emptyset_{i,j}^{t}}{(\Delta x_{i+1} - \Delta x_i)/2} + T_{i,j-1/2}\Delta x_i \frac{\emptyset_{i,j-1}^{t} - \emptyset_{i,j}^{t}}{(\Delta y_j - \Delta y_{j-1})/2} - \right.$$
$$\left. T_{i,j+1/2}\Delta x_i \frac{\emptyset_{i,j+1}^{t} - \emptyset_{i,j}^{t}}{(\Delta y_{j+1} - \Delta y_j)/2}\right) + R_{i,j} - P_{i,j} + N_{i,j} = S_{i,j}\Delta x_i \Delta y_j\left(\emptyset_{i,j}^{t+\Delta t} - \emptyset_{i,j}^{t}\right) \quad (2)$$

Where $T_{i,j}$ is the average aquifer transmissivity.

Because the cells in this problem are identical squares, the flow is steady flow (there is no depth or velocity variation in respect to time) and aquifer is homogeneous, equation above simplified to:

$$T\left(\Delta y \frac{\emptyset_{i-1,j}^{t} - \emptyset_{i,j}^{t}}{\Delta x} - \Delta y \frac{\emptyset_{i+1,j}^{t} - \emptyset_{i,j}^{t}}{\Delta x} + \Delta x \frac{\emptyset_{i,j-1}^{t} - \emptyset_{i,j}^{t}}{\Delta y} - \Delta x \frac{\emptyset_{i,j+1}^{t} - \emptyset_{i,j}^{t}}{\Delta y}\right) = P_{i,j} - N_{i,j} \quad (3)$$

This equation shows how energy head for each cell can connect to cells of its neighborhoods. Also, to number the cells, they were labeled as shown in the figure 1. The equation for central cells defined:

$$T(\phi_n - 4 * \phi_n + \phi_n) = (P_n - N_n)$$

The value of the head in three impervious aquifers boundary obtains from following equation:

$$T(\phi_n - 3 * \phi_n + \phi_n) = (P_n - N_n)$$

And for a lake side of the aquifer we have:

$\Delta x_{lake\text{-}side} = \Delta x/2$

Then if a cell near to impervious boundary:

$$T(\phi_n - 3 * \phi_n + \phi_n) = (P_n - N_n)$$

Otherwise:

$$T(\phi_n - 4 * \phi_n + \phi_n) = (P_n - N_n)$$

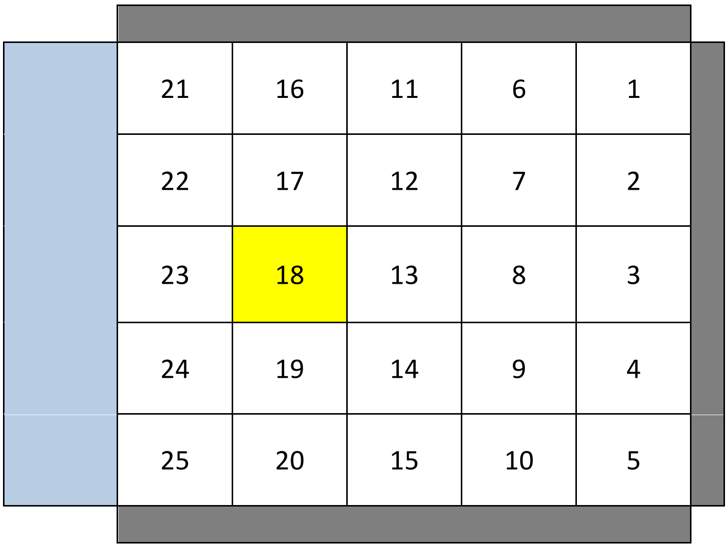

Figure 1. Layout of a 25 cells aquifer

So we have 25 equations for 25 cells. The final matrix of coefficient for ϕ is shown in figure 2.

number of cells	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	-2	1			1																				
2	1	-3	1			1																			
3		1	-3	1			1																		
4			1	-3	1			1																	
5	1			1	-2	0			1																
6		1			0	-3	1			1															
7			1			1	-4	1			1														
8				1			1	-4	1			1													
9					1			1	-4	1			1												
10						1			1	-3	0			1											
11							1			0	-3	1			1										
12								1			1	-4	1			1									
13									1			1	-4	1			1								
14										1			1	-4	1			1							
15											1			1	-3	0			1						
16												1			0	-3	1			1					
17													1			1	-4	1			1				
18														1			1	-4	1			1			
19															1			1	-4	1			1		
20																1			1	-3	0			1	
21																	1			0	-3	1			1
22																		1			1	-4	1		
23																			1			1	-4	1	
24																				1			1	-4	1
25																					1			1	-3

Figure 2. Transmissibility matrix

The R programming software is used to drive the initial head vector ϕ_0 (when there is no pump in the system). The values of ϕ_{i0} are computed as following:

ϕ_{i0} = 13.70 m i =1, …,5

ϕ_{i0} = 12.60 m i =6, …,10

ϕ_{i0} = 10.41 m i =11, …,15

ϕ_{i0} = 7.12 m i =16, …,20

ϕ_{i0} = 2.74 m i =21, …,25

Next step is finding the influence of the pump on each cell's head. As we know, there is a linear relation between pumping and head:

$\Phi_k = a_{i,k} P_i$

which $a_{i,k}$ can be defined as pump coefficient for P_i and the index 'k' is the number of the cell and index 'i' is the number of the pump.

To find the components of the matrix of $a_{i,k}$, the effect of each pump on the head is considered separately and then these components are calculated by using the Superposition Theorem.

Finally, the value of pumping is considered as objective decision variables. Since there is just one consumer located in cell 18, the total cost for each cell is associated with the cost of the pumping and transmission of water to cell 18:

$$min\ f(C, P) = C_6P_6 + C_7P_7 + \cdots + C_{20}P_{20}$$

Constraints are defined based on 1) the restrictions at distances of 1 km and 3 km to prevent encroachment of lake water into the aquifer and 2) the total demand. Table 1 presents the framework of the optimization model.

Table 1. Constraint matrix coefficient

Pumps coefficients															Constraint		
P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	Type	Level	designation
1.0759	0.938	0.7893	0.6808	0.6256	1.295	0.9733	0.7306	0.5877	0.5229	1.8358	0.9296	0.5722	0.4166	0.3554	≤	6.27	Level 16
0.938	0.9272	0.8295	0.7341	0.6808	0.9733	1.0523	0.8304	0.6658	0.5877	0.9296	1.4785	0.774	0.511	0.4166	≤	6.27	Level 17
0.7893	0.8295	0.872	0.8295	0.7893	0.7306	0.8304	0.9875	0.8304	0.7306	0.5722	0.774	1.4172	0.774	0.5722	≤	6.27	Level 18
0.6808	0.7341	0.8295	0.9272	0.938	0.5877	0.6658	0.8304	1.0523	0.9733	0.4166	0.511	0.774	1.4785	0.9296	≤	6.27	Level 19
0.6256	0.6808	0.7893	0.938	1.0759	0.5229	0.5877	0.7306	0.9733	1.295	0.3554	0.4166	0.5722	0.9296	1.8358	≤	6.27	Level 20
0.3465	0.3102	0.2664	0.2323	0.2145	0.4053	0.3262	0.2522	0.2044	0.1818	0.5432	0.337	0.212	0.151	0.1266	≤	2.15	Level 21
0.3102	0.3027	0.2761	0.2485	0.2323	0.3262	0.3314	0.2783	0.2297	0.2044	0.337	0.4183	0.276	0.1876	0.151	≤	2.15	Level 22
0.2664	0.2761	0.2849	0.2761	0.2664	0.2522	0.2783	0.3088	0.2783	0.2522	0.212	0.276	0.3938	0.276	0.212	≤	2.15	Level 23
0.2323	0.2485	0.2761	0.3027	0.3102	0.2044	0.2297	0.2783	0.3314	0.3262	0.151	0.1876	0.276	0.4183	0.337	≤	2.15	Level 24
0.2145	0.2323	0.2664	0.3102	0.3465	0.1818	0.2044	0.2522	0.3262	0.4053	0.1266	0.151	0.212	0.337	0.5432	≤	2.15	Level 25
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	≥	7	Demand
3.9	3.2	3	3.2	3.9	3.2	2.4	2	2.4	3.2	3	2	1	2	3	-	min	Cost

'`lpsolve`' solver tool in **R** was used to model the linear programing model. The objective function (cost function) is built based on distance calculation from cell 18 for pumping from each cell. The constraints were modeled by defining the constraint matrix, the right-hand side and finally the sense of constraints. This written programing code is presented in the Appendix.

Discussion and Conclusion:

Since the aquifer is square in shape and the user is in cell 18, we expect symmetrical results for amount of pumping for cells surrounding the user. For instance, if we are pumping from cell 19, we expect to have the same amount of pumping from cell 17 because they are in same distance from the user (cell 18) and both are in the same column. The initial results from the Linear Programming does not show the expected symmetry in the results. This is because there is more than one answer for this LP. To tackle this issue, 6 more set of constrains were added to the constraint matrix to force the pumping be symmetric.

After applying this modification, the pumping results are as below:

$P_i = 0$ for i=6, 7, …, 15

$P_{16} = P_{20} = 0.679$ MGD

$P_{17} = P_{19} = 2.021$ MGD

$P_{18} = 1.600$ MGD

Appendix

```
library(lpSolve)
library(gridExtra)
library(grid)

###############################################################
### Reservoir Dimentions ###
###############################################################

x <- 10000 # m
y <- 10000 # m
irow <- 5
jcol <- 5

deltax <- x/irow # m
deltay <- y/jcol # m
dim <- irow*jcol

###############################################################
### Model Characteristics ###
###############################################################

Trans <- 1000*365/1000000 # Mm2/year
N <- 0.1 # m/year
Rech <- N*deltax*deltay/1000000 # Mm^3/year

###############################################################
### Building Coefficient Matrix ###
###############################################################

matT <- matrix(0,dim,dim)

# East abd West cells
for (i in 1:(dim-1)){
        matT[i,i+1] <- Trans
        matT[i+1,i] <- Trans
        if(i %% irow==0){
                matT[i,i+1] <- 0
                matT[i+1,i] <- 0
        }
}

# North and South cells
for (i in 1:(dim-irow)){
        matT[i,i+irow] <- Trans
        matT[i+irow,i] <- Trans
}

## Center (Main Diagonal)
for (i in 1:dim){
        if (i %% irow == 0 | i %% irow == 1 ){
                matT[i,i] <- -3*Trans
        }else{
                matT[i,i] <- -4*Trans
        }
}

for (i in 1:dim){
        if (i>=1 && i<=irow){
                if (i>=2 && i<=(irow-1)){
```

```
                matT[i,i] <- -3*Trans
            }else{
                matT[i,i] <- -2*Trans
            }
        }else if (i>=(dim-irow+1) && i<=dim){
            if (i>=(dim-irow+2) && i<=(dim-1)){
                matT[i,i] <- -5*Trans
            }else{
                matT[i,i] <- -4*Trans
            }
        }
}

###############################################################
### initial values of h (h0) ###
###############################################################

rhs <- rep(-Rech,dim)
h0<-solve(matT,rhs)

hh <- matrix(h0,irow,jcol,byrow = F)
hhh <- hh
for (i in 1:nrow(hh)){
        for (j in 1:ncol(hh)){
                hhh[i,j] <- hh[i,6-j]
        }
}

# Assign colunm and row names for the h0 matrix
colname <- c()
for (i in 1:ncol(hhh)){
        colname <- c(colname,paste0("i=",6-i))
}

rowname <- c()
for (i in 1:(nrow(hhh))){
        rowname <- c(rowname,paste0("j=",i))
}

colnames(hhh) <- colname
row.names(hhh) <- rowname

# h0 resultant matrix
hhh

###############################################################
### Optimization - Creating the influence matrix ###
###############################################################

hmin1 <- 0.64 # m
hmin2 <- 0.95 # m
Ptot <- 7 # total demand of the consumer Mm^3/year

a <- c()
for (i in 6:20){
        rhs[i]<- -Rech+1
        h_temp<-solve(matT,rhs)
        a <- rbind(a,(h0-h_temp)/1)
        rhs <- rep(-Rech,dim)
}
```

```
a <- t(a[,16:25]) # Constraint for minimum heads in cells 16:25

a_final <- rbind(a,rep(1,15)) # Influence matrix including the last
constraint of total demand


a_final <- round(a_final,4)
a_sym2_1 <- c(1,0,0,0,-1,0,0,0,0,0,0,0,0,0,0)
a_sym2_2 <- c(0,1,0,-1,0,0,0,0,0,0,0,0,0,0,0)
a_sym3_1 <- c(0,0,0,0,0,1,0,0,0,-1,0,0,0,0,0)
a_sym3_2 <- c(0,0,0,0,0,0,1,0,-1,0,0,0,0,0,0)
a_sym4_1 <- c(0,0,0,0,0,0,0,0,0,0,1,0,0,0,-1)
a_sym4_2 <- c(0,0,0,0,0,0,0,0,0,0,0,1,0,-1,0)

a_final <-
rbind(a_final,a_sym2_1,a_sym2_2,a_sym3_1,a_sym3_2,a_sym4_1,a_sym4_2)

constr.dir <- c(rep("<=",10),rep("=",7))
rhs_constr <- c((h0[16:20]-hmin2),h0[21:25]-hmin1,Ptot,rep(0,6))


##############################################################
### Optimization - Cost Function ###
##############################################################

X <- 3 # cell 18th i value
Y <- 3 # cell 18th j value
Cost <- data.frame()
for (i in 1:3){
        for (j in 1:5){
                Cost[i,j]<-sqrt((i-X)^2+(j-Y)^2)
        }
}
obj.fun <- as.vector(t(Cost))+1 # coeficient of Cost Function MU/m^3

##############################################################
### Optimization - Linear Programing Solution ###
##############################################################

pupm.sol <- lp("min", obj.fun, a_final, constr.dir, rhs_constr,
compute.sens = TRUE)

##############################################################
### Generate meaningful output ###
### Get some constants to make readable output ###
##############################################################
HowManyDecision <- length(pupm.sol$solution)
P <- c()
message("Annual Operation Cost = $",pupm.sol$objval*1000) #objective
function value
for (i in 1:HowManyDecision) {
        message("Flow from Pipe ",i+5," = ",pupm.sol$solution[i])
        P[i] <- pupm.sol$solution[i]
}

#########################################################
### contour plot of head - note axes are rotated ###
##############################################################
PP <- c(rep(0,5),P,rep(0,5))
rhs <- rep(-Rech,dim)+PP
h <-solve(matT,rhs)
```

```r
h <- matrix(h,irow,jcol,byrow = T)

distancex <- seq(deltax/2,deltax*(irow-0.5),deltax)
distancey <- seq(deltay/2,deltay*(jcol-0.5),deltay)
contour(distancex,distancey,h, xlim = rev(range(distancex, finite = TRUE)),
        ylim = rev(range(distancey, finite = TRUE)),
        plot.title = title(main = "Head (Blue) Map",
                           xlab = "==== Meters (X axis)",
                           ylab = "==== Meters (Y axis)"),
        col="blue",lwd=3,nlevels=19,tck=1)


h_f <- matrix(h,irow,jcol,byrow = F)

hh_f <- h_f
for (i in 1:nrow(h_f)){
        for (j in 1:ncol(h_f)){
                hh_f[i,j] <- h_f[i,6-j]
        }
}
```