abhardw3_Assignment5

Aarush Bhardwaj

09/11/2021

```
# Setting Working Directory

 setwd("D:/A_Sem_2/Quantative Management Modeling/Assignments/Assignment 5/Quant_Assignment5")
```

Installing the required packages.

```
library(Benchmarking)
```

```
## Warning: package 'Benchmarking' was built under R version 4.0.5
```

```
## Loading required package: lpSolveAPI
```

```
## Loading required package: ucminf
```

```
## Loading required package: quadprog
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.4
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.6     v dplyr   1.0.4
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

COMPUTING FORMULATION

```
# creating the vectors

input  <-  matrix(c(150,400,320,520,350,  320,  200,  700,  1200,  2000,  1200,  700),ncol  =  2)
output <- matrix(c(14000,14000,42000,28000,19000,14000,3500,21000,10500,42000,25000, 15000),ncol = 2)

# Assigning names to columns

colnames(output) <- c("staff_daily_hours","supplies_daily")
colnames(input) <- c("daily_reimbursed_patient", "daily_privately_paid_patient")

# values of input & output

input
```

```
##      daily_reimbursed_patient daily_privately_paid_patient
## [1,]                      150                          200
## [2,]                      400                          700
## [3,]                      320                         1200
## [4,]                      520                         2000
## [5,]                      350                         1200
## [6,]                      320                          700
```

```
output
```

```
##      staff_daily_hours supplies_daily
## [1,]             14000           3500
## [2,]             14000          21000
## [3,]             42000          10500
## [4,]             28000          42000
## [5,]             19000          25000
## [6,]             14000          15000
```

It is clear from the table above that the results are similar to the performance data table from the Hope Valley Health Care Association's 6 nursing facilities.

Now, we will use a tool called "DEA" that can help organizations to identify and allocate their resources to enhance their efficiency.

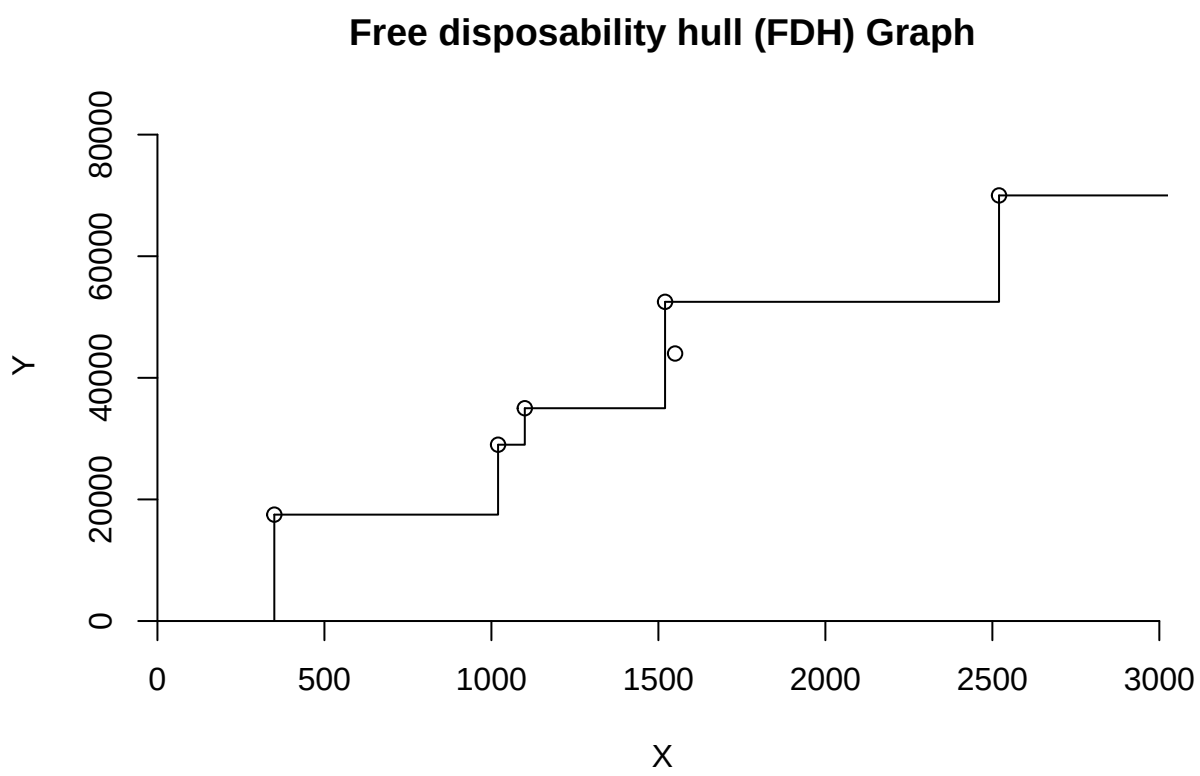
DEA Analysis (Using FDH)

```
analysis_fdh<-  dea(input,output,RTS  =  "fdh")

eff_fdh <- as.data.frame(analysis_fdh$eff)

colnames(eff_fdh)  <-  c("efficiency_fdh")

peer_fdh  <-  peers(analysis_fdh)

colnames(peer_fdh) <- c("peer1_fdh")
```

```
lambda_fdh <- lambda(analysis_fdh)

colnames(lambda_fdh) <- c("L1_fdh", "L2_fdh", "L3_fdh", "L4_fdh", "L5_fdh", "L6_fdh")

peer_lamb_eff_fdh  <-  cbind(peer_fdh,  lambda_fdh,  eff_fdh)

peer_lamb_eff_fdh
```

```
##   peer1_fdh L1_fdh L2_fdh L3_fdh L4_fdh L5_fdh L6_fdh efficiency_fdh
## 1         1      1      0      0      0      0      0              1
## 2         2      0      1      0      0      0      0              1
## 3         3      0      0      1      0      0      0              1
## 4         4      0      0      0      1      0      0              1
## 5         5      0      0      0      0      1      0              1
## 6         6      0      0      0      0      0      1              1
```

```
# Plotting results
dea.plot(input,output,RTS="fdh", main="Free disposability hull (FDH) Graph")
```

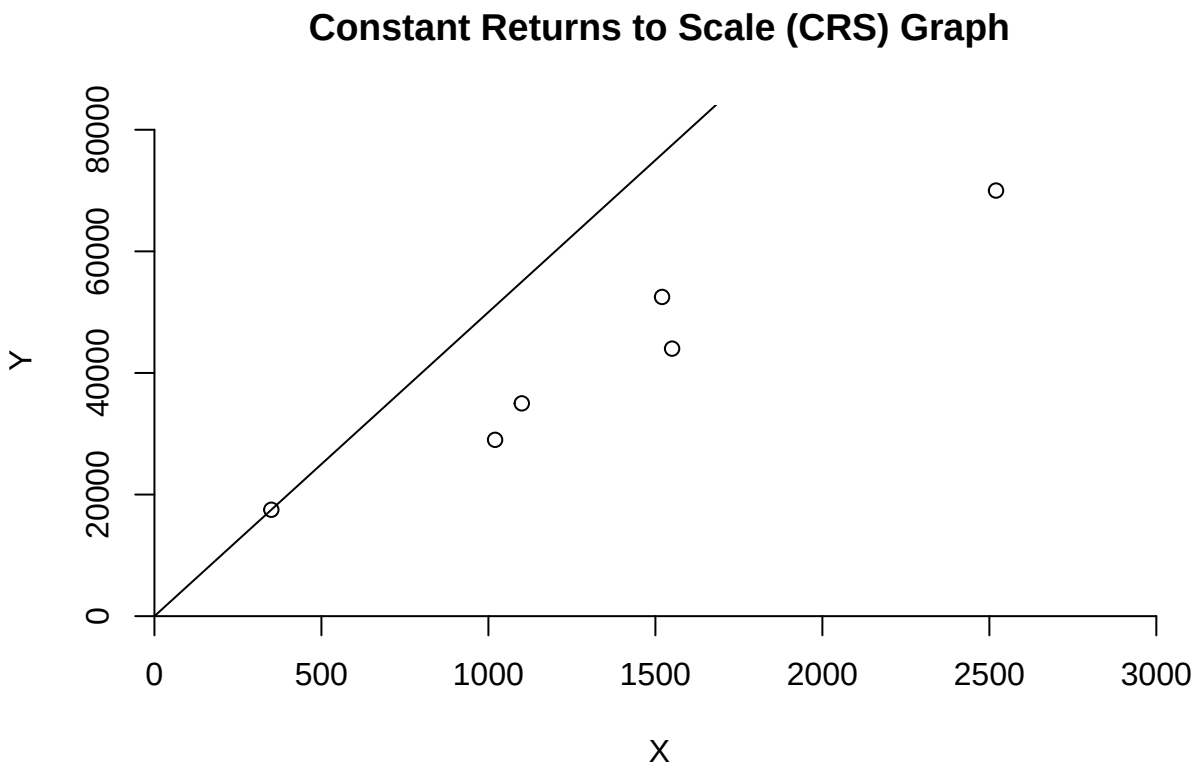


DEA Analysis (Using CRS)

```r
analysis_crs <- dea(input,output,RTS = "crs")

eff_crs <- as.data.frame(analysis_crs$eff)

colnames(eff_crs) <- c("efficiency_crs")

peer_crs <- peers(analysis_crs)

colnames(peer_crs) <- c("peer1_crs", "peer2_crs", "peer3_crs")

lambda_crs <- lambda(analysis_crs)

colnames(lambda_crs) <- c("L1_crs", "L2_crs", "L3_crs", "L4_crs")

peer_lamb_eff_crs <- cbind(peer_crs, lambda_crs, eff_crs)

peer_lamb_eff_crs
```

```
##   peer1_crs peer2_crs peer3_crs    L1_crs     L2_crs L3_crs    L4_crs
## 1         1        NA        NA 1.0000000 0.00000000      0 0.0000000
## 2         2        NA        NA 0.0000000 1.00000000      0 0.0000000
## 3         3        NA        NA 0.0000000 0.00000000      1 0.0000000
## 4         4        NA        NA 0.0000000 0.00000000      0 1.0000000
## 5         1         2         4 0.2000000 0.08048142      0 0.5383307
## 6         1         2         4 0.3428571 0.39499264      0 0.1310751
##   efficiency_crs
## 1      1.0000000
## 2      1.0000000
## 3      1.0000000
## 4      1.0000000
## 5      0.9774987
## 6      0.8674521
```

```r
# Plotting results
dea.plot(input,output,RTS="crs", main="Constant Returns to Scale (CRS) Graph")
```



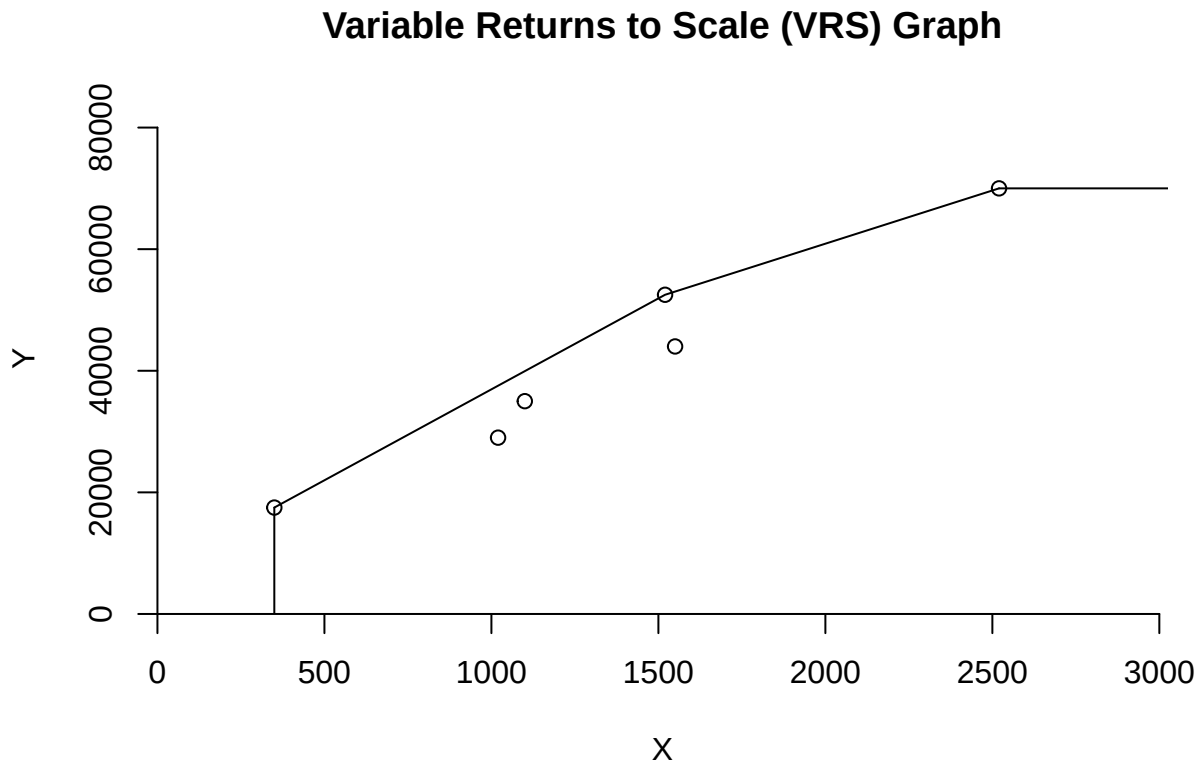
DEA Analysis (Using VRS)

```
analysis_vrs  <-  dea(input,output,RTS  =  "vrs")

eff_vrs <- as.data.frame(analysis_vrs$eff)

colnames(eff_vrs)  <-  c("efficiency_vrs")

peer_vrs <- peers(analysis_vrs)

colnames(peer_vrs) <- c("peer1_vrs", "peer2_vrs", "peer3_vrs")

lambda_vrs <- lambda(analysis_vrs)

colnames(lambda_vrs) <- c("L1_vrs", "L2_vrs", "L3_vrs", "L4_vrs", "L5_vrs")

peer_lamb_eff_vrs <- cbind(peer_vrs, lambda_vrs, eff_vrs)

peer_lamb_eff_vrs
```

```
##   peer1_vrs peer2_vrs peer3_vrs    L1_vrs    L2_vrs L3_vrs L4_vrs    L5_vrs
## 1         1        NA        NA 1.0000000 0.0000000      0      0 0.0000000
## 2         2        NA        NA 0.0000000 1.0000000      0      0 0.0000000
```

```
## 3         3        NA        NA 0.0000000 0.0000000      1      0 0.0000000
## 4         4        NA        NA 0.0000000 0.0000000      0      1 0.0000000
## 5         5        NA        NA 0.0000000 0.0000000      0      0 1.0000000
## 6         1         2         5 0.4014399 0.3422606      0      0 0.2562995
##   efficiency_vrs
## 1      1.0000000
## 2      1.0000000
## 3      1.0000000
## 4      1.0000000
## 5      1.0000000
## 6      0.8963283
```

```
# Plotting results
dea.plot(input,output,RTS="vrs", main="Variable Returns to Scale (VRS) Graph")
```



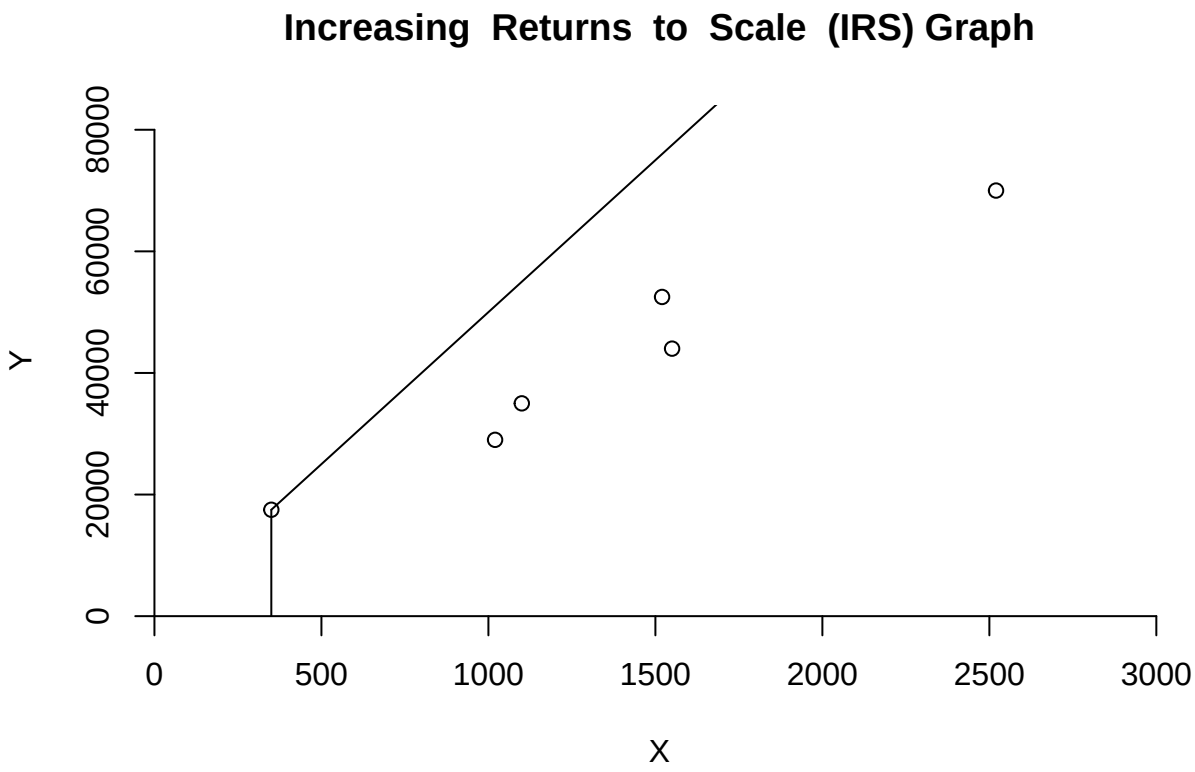
DEA Analysis (Using IRS)

```
analysis_irs  <-  dea(input,output,RTS  =  "irs")

eff_irs <- as.data.frame(analysis_irs$eff)

colnames(eff_irs)  <-  c("efficiency_irs")
```

```r
peer_irs <- peers(analysis_irs)

colnames(peer_irs)  <-  c("peer1_irs",  "peer2_irs",  "peer3_irs")

lambda_irs  <-  lambda(analysis_irs)

colnames(lambda_irs) <- c("L1_irs", "L2_irs", "L3_irs", "L4_irs", "L5_irs")

peer_lamb_eff_irs <- cbind(peer_irs, lambda_irs, eff_irs)

peer_lamb_eff_irs
```

```
##   peer1_irs peer2_irs peer3_irs    L1_irs    L2_irs L3_irs L4_irs    L5_irs
## 1         1        NA        NA 1.0000000 0.0000000      0      0 0.0000000
## 2         2        NA        NA 0.0000000 1.0000000      0      0 0.0000000
## 3         3        NA        NA 0.0000000 0.0000000      1      0 0.0000000
## 4         4        NA        NA 0.0000000 0.0000000      0      1 0.0000000
## 5         5        NA        NA 0.0000000 0.0000000      0      0 1.0000000
## 6         1         2         5 0.4014399 0.3422606      0      0 0.2562995
##   efficiency_irs
## 1      1.0000000
## 2      1.0000000
## 3      1.0000000
## 4      1.0000000
## 5      1.0000000
## 6      0.8963283
```

```r
# Plotting results
dea.plot(input,output,RTS="irs",  main="Increasing  Returns  to  Scale  (IRS) Graph")
```



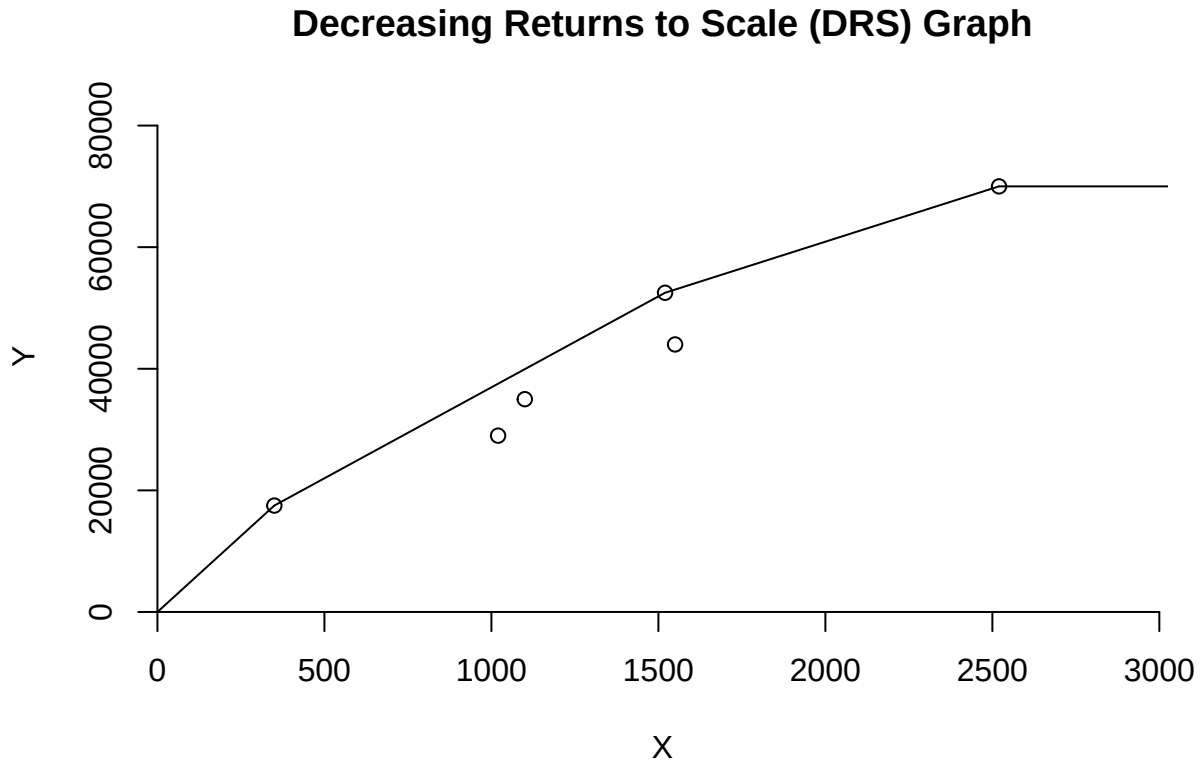
DEA Analysis (Using DRS)

```
analysis_drs  <-  dea(input,output,RTS  =  "drs")

eff_drs <- as.data.frame(analysis_drs$eff)

colnames(eff_drs)  <-  c("efficiency_drs")

peer_drs <- peers(analysis_drs)

colnames(peer_drs) <- c("peer1_drs", "peer2_drs", "peer3_drs")

lambda_drs <- lambda(analysis_drs)

colnames(lambda_drs) <- c("L1_drs", "L2_drs", "L3_drs", "L4_drs")

peer_lamb_eff_drs <- cbind(peer_drs, lambda_drs, eff_drs)

peer_lamb_eff_drs
```

```
##   peer1_drs peer2_drs peer3_drs    L1_drs     L2_drs L3_drs    L4_drs
## 1         1        NA        NA 1.0000000 0.00000000      0 0.0000000
## 2         2        NA        NA 0.0000000 1.00000000      0 0.0000000
```

```
## 3         3        NA        NA 0.0000000 0.00000000       1 0.0000000
## 4         4        NA        NA 0.0000000 0.00000000       0 1.0000000
## 5         1         2         4 0.2000000 0.08048142       0 0.5383307
## 6         1         2         4 0.3428571 0.39499264       0 0.1310751
##   efficiency_drs
## 1      1.0000000
## 2      1.0000000
## 3      1.0000000
## 4      1.0000000
## 5      0.9774987
## 6      0.8674521
```

```
# Plotting results
dea.plot(input,output,RTS="drs", main="Decreasing Returns to Scale (DRS) Graph")
```



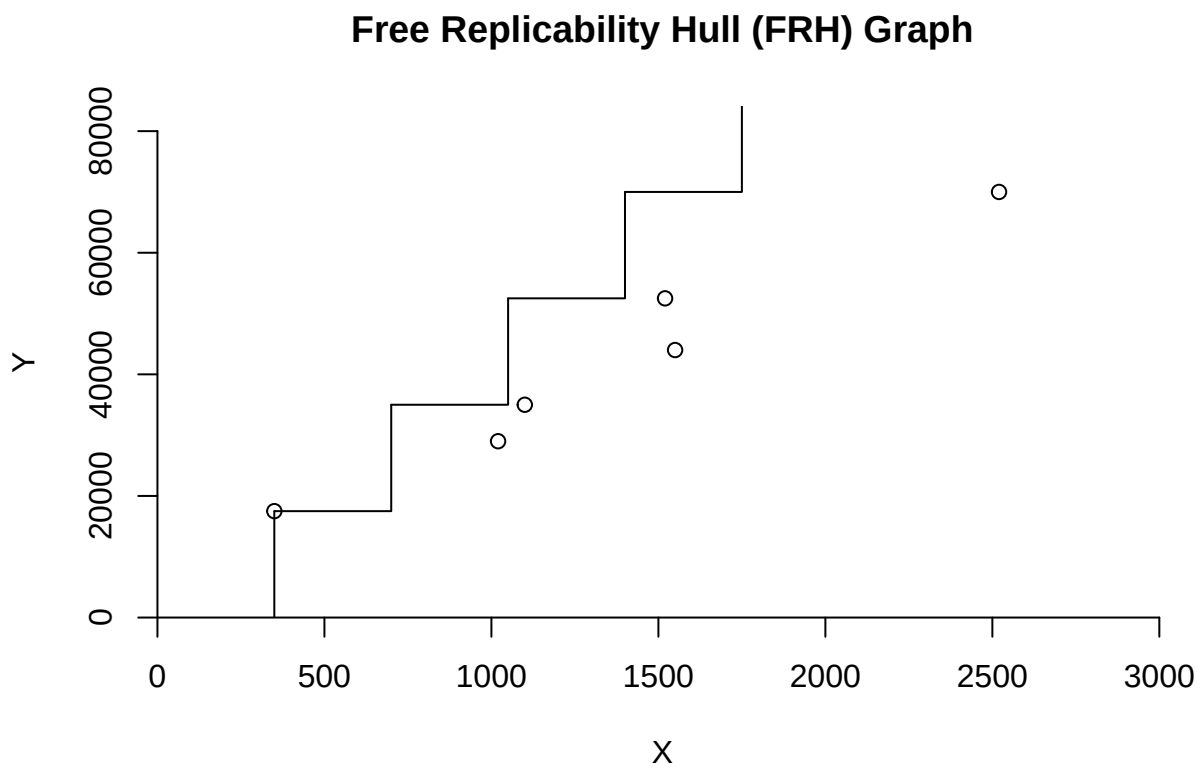
DEA Analysis (Using FRH)

```
analysis_frh  <-  dea(input,output,RTS  =  "add")

eff_frh <- as.data.frame(analysis_frh$eff)

colnames(eff_frh) <- c("efficiency_frh")
```

```r
peer_frh <- peers(analysis_frh)

colnames(peer_frh) <- c("peer1_frh")

lambda_frh <- lambda(analysis_frh)

colnames(lambda_frh) <- c("L1_frh", "L2_frh", "L3_frh", "L4_frh", "L5_frh", "L6_frh")

peer_lamb_eff_frh <- cbind(peer_frh, lambda_frh, eff_frh)

peer_lamb_eff_frh
```

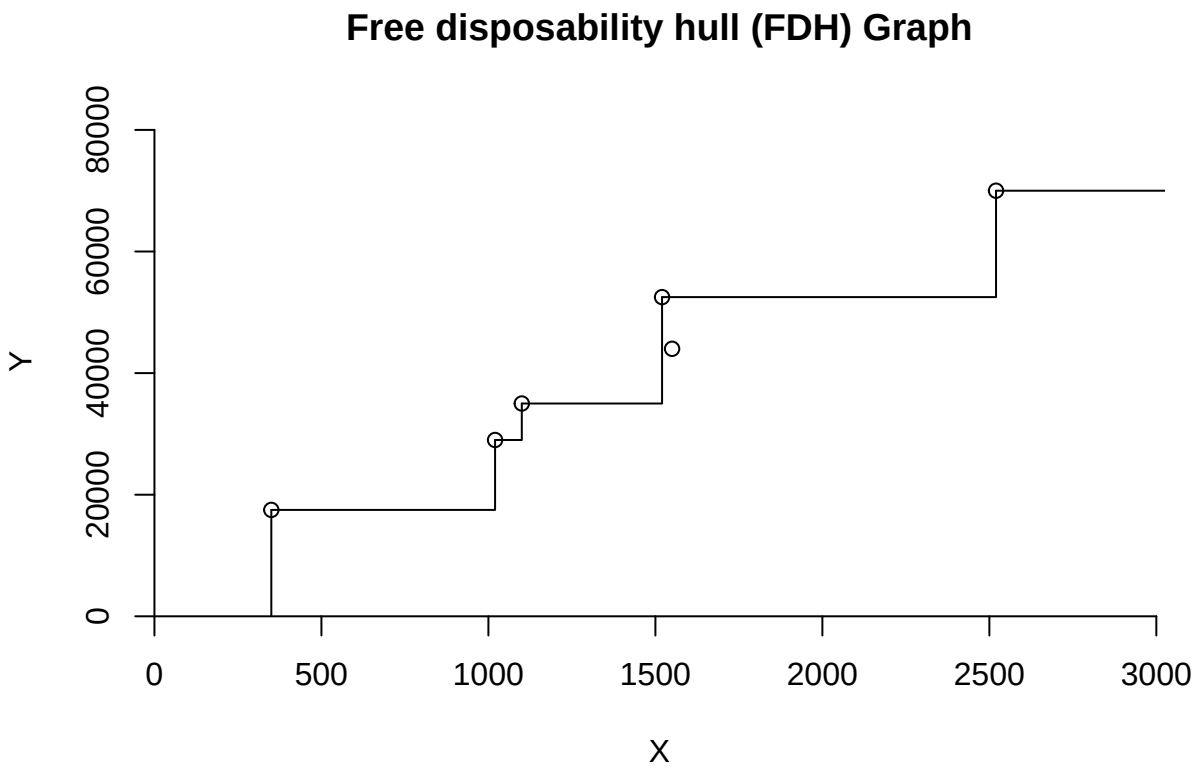
```
##   peer1_frh L1_frh L2_frh L3_frh L4_frh L5_frh L6_frh efficiency_frh
## 1         1      1      0      0      0      0      0              1
## 2         2      0      1      0      0      0      0              1
## 3         3      0      0      1      0      0      0              1
## 4         4      0      0      0      1      0      0              1
## 5         5      0      0      0      0      1      0              1
## 6         6      0      0      0      0      0      1              1
```

```r
# Plotting results
dea.plot(input,output,RTS="add", main="Free Replicability Hull (FRH) Graph")
```

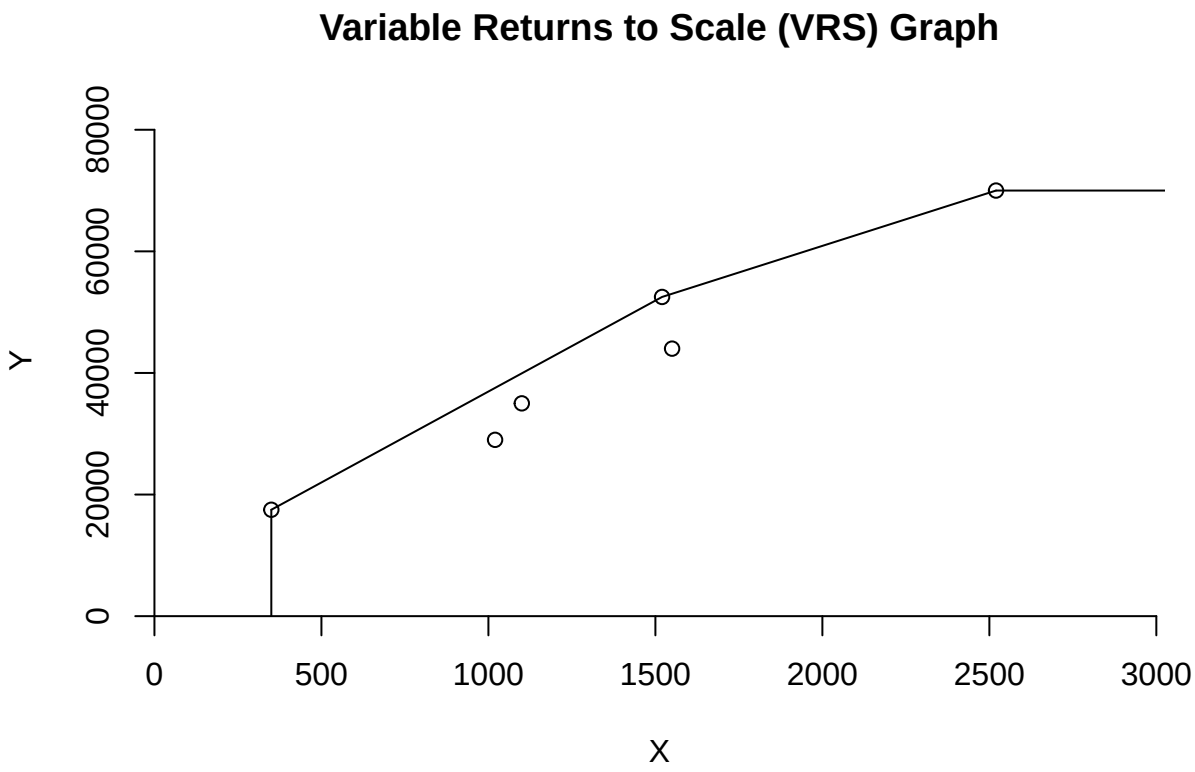


Comparing among different assumptions

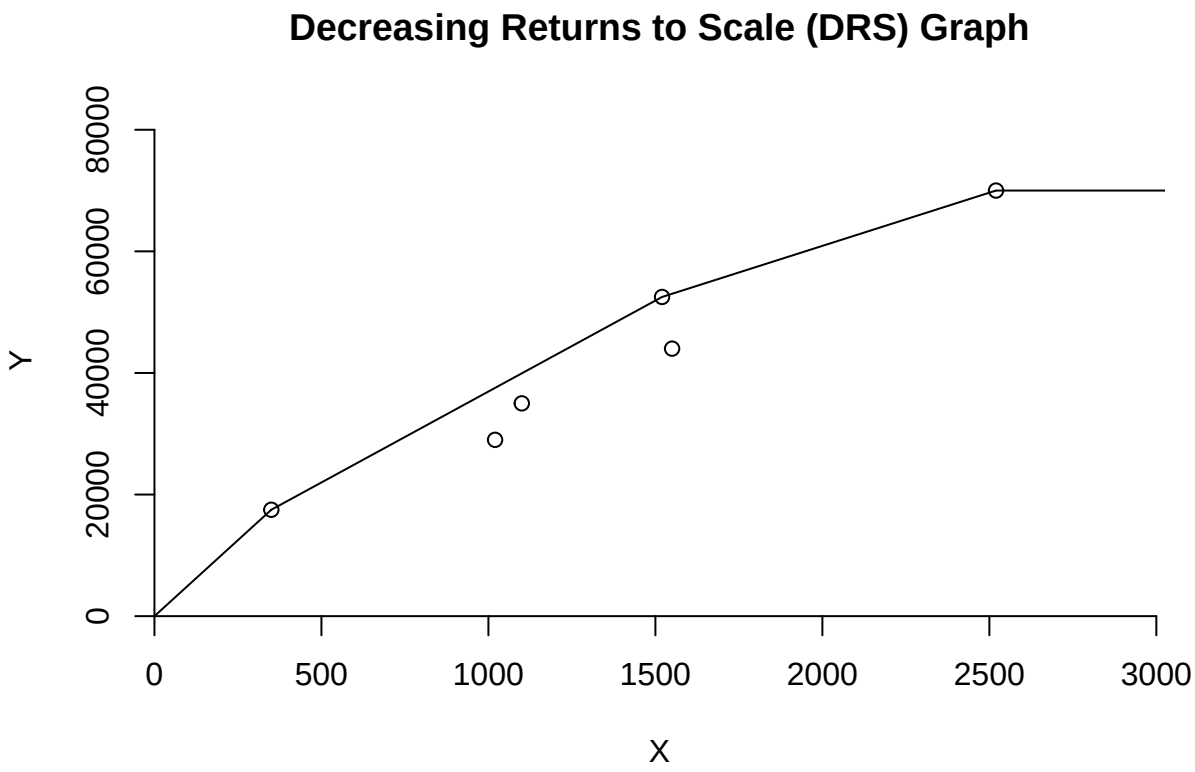
```
dea.plot(input,output,RTS="fdh", main="Free disposability hull (FDH) Graph")
```



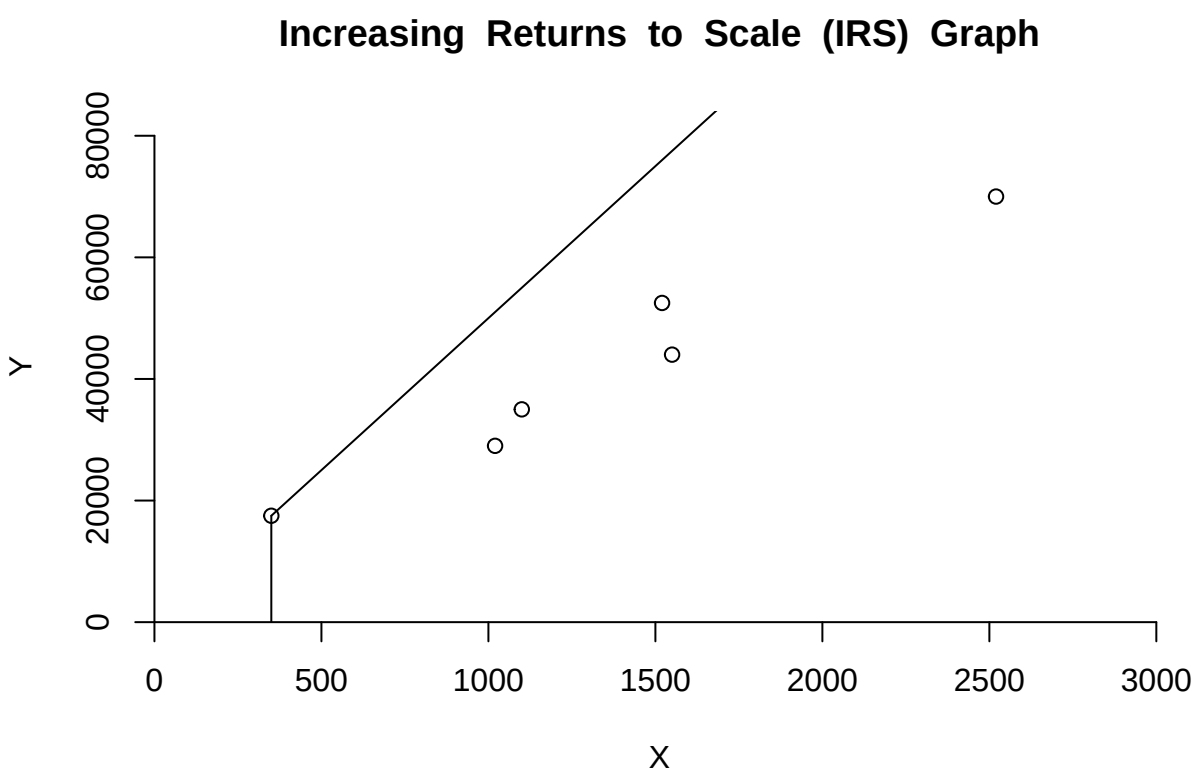

```
dea.plot(input,output,RTS="vrs", main="Variable Returns to Scale (VRS) Graph")
```



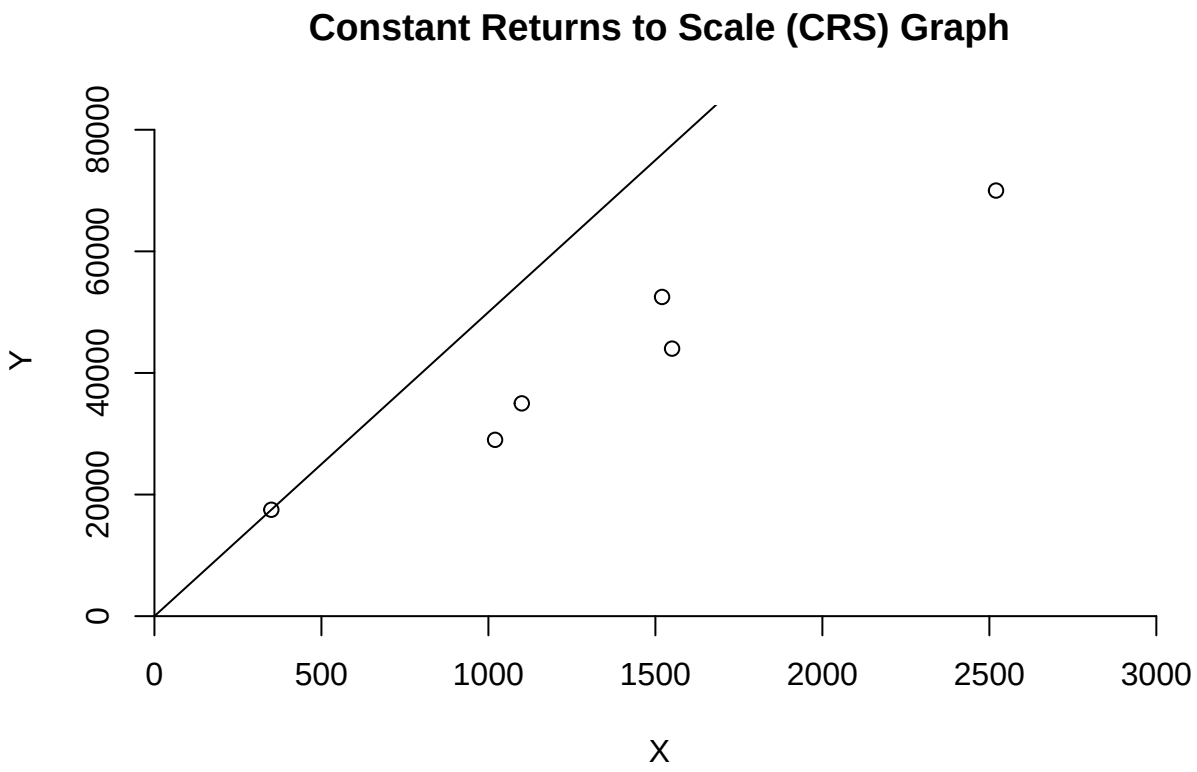

```
dea.plot(input,output,RTS="drs", main="Decreasing Returns to Scale (DRS) Graph")
```



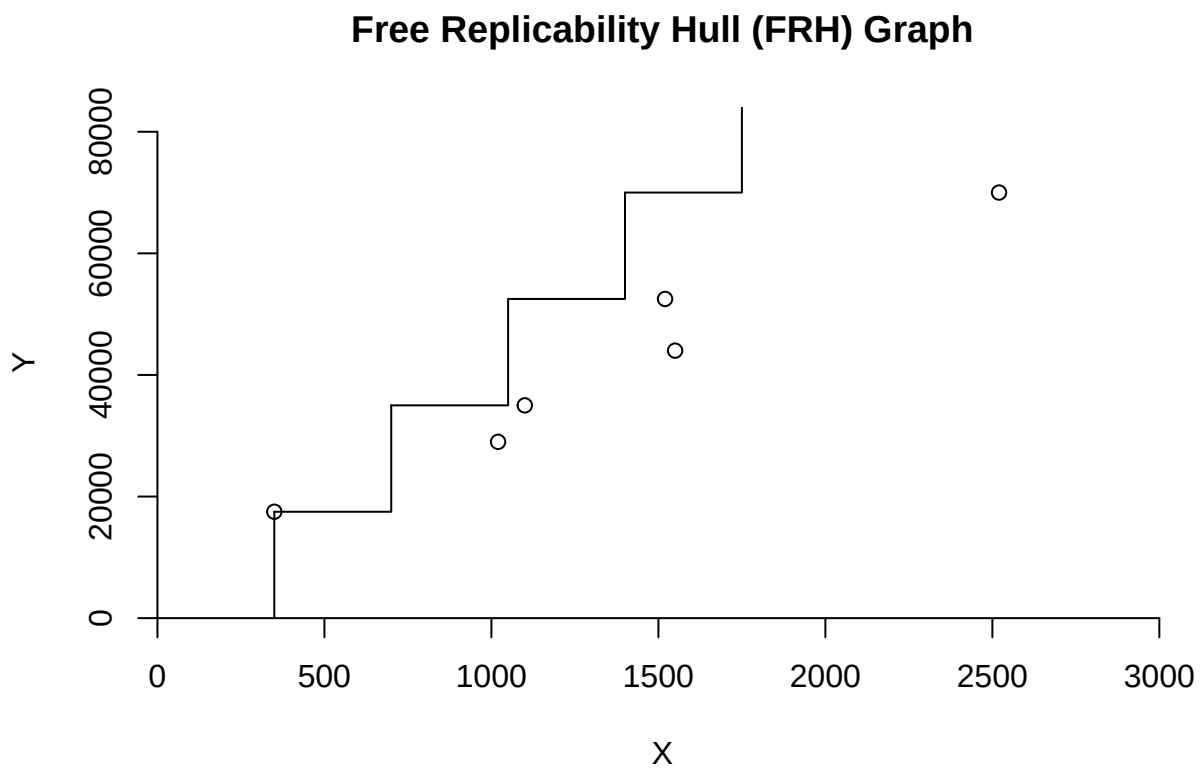

```
dea.plot(input,output,RTS="irs",  main="Increasing  Returns  to  Scale  (IRS)  Graph")
```




```
dea.plot(input,output,RTS="crs", main="Constant Returns to Scale (CRS) Graph")
```




```
dea.plot(input,output,RTS="add", main="Free Replicability Hull (FRH) Graph")
```



The above charts allows us to compare the results of each DEA model.

Let's talk a little about each one of them. FDH is the smallest technology set, as shown and it seeks to create fewer outputs (number of patient days reimbursed by third-party sources and number of patient days reimbursed privately) with more inputs (number of patient days reimbursed by third-party sources and number of patient days reimbursed privately) (staffing labor and the cost of supplies).

FDH is the most popular model among businesses, yet it has several flaws owing to its assumptions. As we can see, all of the efficiency of this model are only 1, however it is not as efficient as we thought when compared to other models since we identify areas/units to improve.

VRS is larger than FDH because it "fills-out" the spaces that FDH reduced. Here we can see that unit 6 can improve its efficiency.

As the graphs show, DRS and IRS are bigger than VRS. For smaller input values, DRS seeks to enlarge the set, whereas the IRS tries to raise the technology. Units 5 and 6 might increase their efficiency, according to DRS, and facility 6 could improve as well, according to IRS.

CRS is the largest technology set, allowing us to assess whether there are any conceivable scaling up or down combinations. Units 5 and 6 require improvement based on the efficiency numbers.

The purpose of FRH, which is larger than FDH but less than CRS, is to replace deterministic data with random variables.

Question 2 - Research and Development Division of Emax Corporation

Objective function:

max: 20 X1 + 15 X2 + 25 X3 - 6 Y1P - 6 Y1M - 3 Y2M S.T : Employment Level

6x1 + 4x2 + 5x3 - (Y1P - Y1M) = 50

Earnings Next Year

8x1 + 7x2 + 5x3 - (Y2P - Y2M) = 75

Non-negativity constraint

X1, X2, X3 >= 0 Y1P, Y1M, Y2P, Y2M >= 0

```
library(lpSolveAPI)

# Load the data
emax <-  read.lp("E_Max.lp")
emax
```

```
## Model name:
##             X1    X2    X3   Y1P   Y1M   Y2M   Y2P
## Maximize    20    15    25    -6    -6    -3     0
## R1           6     4     5    -1     1     0     0  =  50
## R2           8     7     5     0     0     1    -1  =  75
## Kind       Std   Std   Std   Std   Std   Std   Std
## Type      Real  Real  Real  Real  Real  Real  Real
## Upper      Inf   Inf   Inf   Inf   Inf   Inf   Inf
## Lower        0     0     0     0     0     0     0
```

```
solve(emax)
```

```
## [1] 0
```

The solver returns 0 as an output. This means that it is able to find a solution.

```
get.objective(emax)
```

```
## [1] 225
```

Here we are maximizing the profit by reducing other goals of the company. The value 225 is the penalty for failing to meet the goals on the objective function.

```
get.variables(emax)
```

```
## [1]  0  0 15 25  0  0  0
```

The above order results from order of the variables in the objective function.

So for us, the results are:

X1 = 0,X2 = 0,X3 = 15,Y1P = 25,Y1M = 0,Y2M = 0,Y2P = 0

This means that the earning (Y2) expectations are fully satisfied.

For workforce, the goal projected exceeds by 25 and based on the total profit of product 3, it has a negative result on its profit by 15.