Quantitative Management Modeliing Assignment-5

1. The Hope Valley Health Care Association owns and operates six nursing homes in adjoining states. An evaluation of their efficiency has been undertaken using two inputs and two outputs. The inputs are staffing labor (measured in average hours per day) and the cost of supplies (in thousands of dollars per day). The outputs are the number of patient-days reimbursed by third-party sources and the number of patient-days reimbursed privately. A summary of performance data is shown in the table below.

DMU	Staff Hours per Day	Supplies per Day	Reimbursed Patient-Days	Privately Paid Patient-Days
Facility 1	150	0.2	14,000	3,500
Facility 2	400	0.7	14,000	21,000
Facility 3	320	1.2	42,000	10,500
Facility 4	520	2.0	28,000	42,000
Facility 5	350	1.2	19,000	25,000
Facility 6	320	0.7	14,000	15,000

Using Benchmarking Library for DEA

```
library(Benchmarking) #comparison of performance measures between similar entities against recognized s
library(readxl) # for loading from the excelsheet data(Hope Valley Health Care Association) for DEA ana
```

Now, we read our input data_DEA. We will read the data_DEA from an excel file.

The problem has 6 DMUs with two inputs and two outputs.

Inputs: Staffing Labor, Cost of Supplies

Outputs: No of patient-days reimbursed by third party, No of patient-days reimbursed privately

```
#Read the data_DEA from excel file
data_DEA <- read_excel("C:/Users/khush/Documents/DEA.xlsx")
#See the data_DEA
data_DEA
```

```
## # A tibble: 6 x 5
##   DMU        `Staff Hours pe~ `Supplies per D~ `Reimbursed Pat~ `Privately Paid~
##   <chr>                 <dbl>            <dbl>            <dbl>            <dbl>
## 1 Facility 1              150              0.2            14000             3500
## 2 Facility 2              400              0.7            14000            21000
## 3 Facility 3              320              1.2            42000            10500
## 4 Facility 4              520              2              28000            42000
## 5 Facility 5              350              1.2            19000            25000
## 6 Facility 6              320              0.7            14000            15000
```

```r
#Facility1 to Facility 6 are the DMUs
DMU_names <- data_DEA[1]
DMU_names
```

```
## # A tibble: 6 x 1
##   DMU
##   <chr>
## 1 Facility 1
## 2 Facility 2
## 3 Facility 3
## 4 Facility 4
## 5 Facility 5
## 6 Facility 6
```

```r
#Lets see the Inputs
inputs <- data_DEA[c(2,3)]
inputs
```

```
## # A tibble: 6 x 2
##   `Staff Hours per Day` `Supplies per Day`
##                   <dbl>              <dbl>
## 1                   150                0.2
## 2                   400                0.7
## 3                   320                1.2
## 4                   520                2
## 5                   350                1.2
## 6                   320                0.7
```

```r
#Now, see the outputs
outputs <- data_DEA[c(4,5)]
outputs
```

```
## # A tibble: 6 x 2
##   `Reimbursed Patient-Days` `Privately Paid Patient-Days`
##                       <dbl>                         <dbl>
## 1                     14000                          3500
## 2                     14000                         21000
## 3                     42000                         10500
## 4                     28000                         42000
## 5                     19000                         25000
## 6                     14000                         15000
```

```r
#Create the input matrix
input_matrix <- matrix(c(data_DEA$`Staff Hours per Day`,data_DEA$`Supplies per Day`),ncol = 2)
#Lets see the input matrix
input_matrix
```

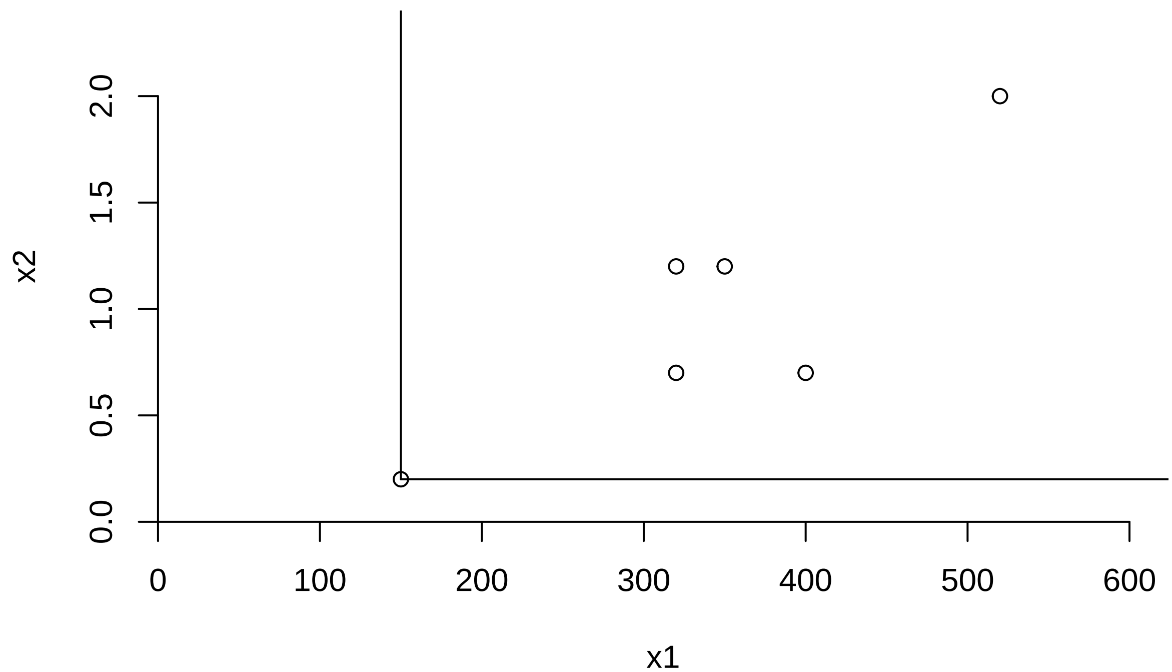
```
##      [,1] [,2]
## [1,]  150  0.2
## [2,]  400  0.7
## [3,]  320  1.2
```

```
## [4,]  520  2.0
## [5,]  350  1.2
## [6,]  320  0.7
```

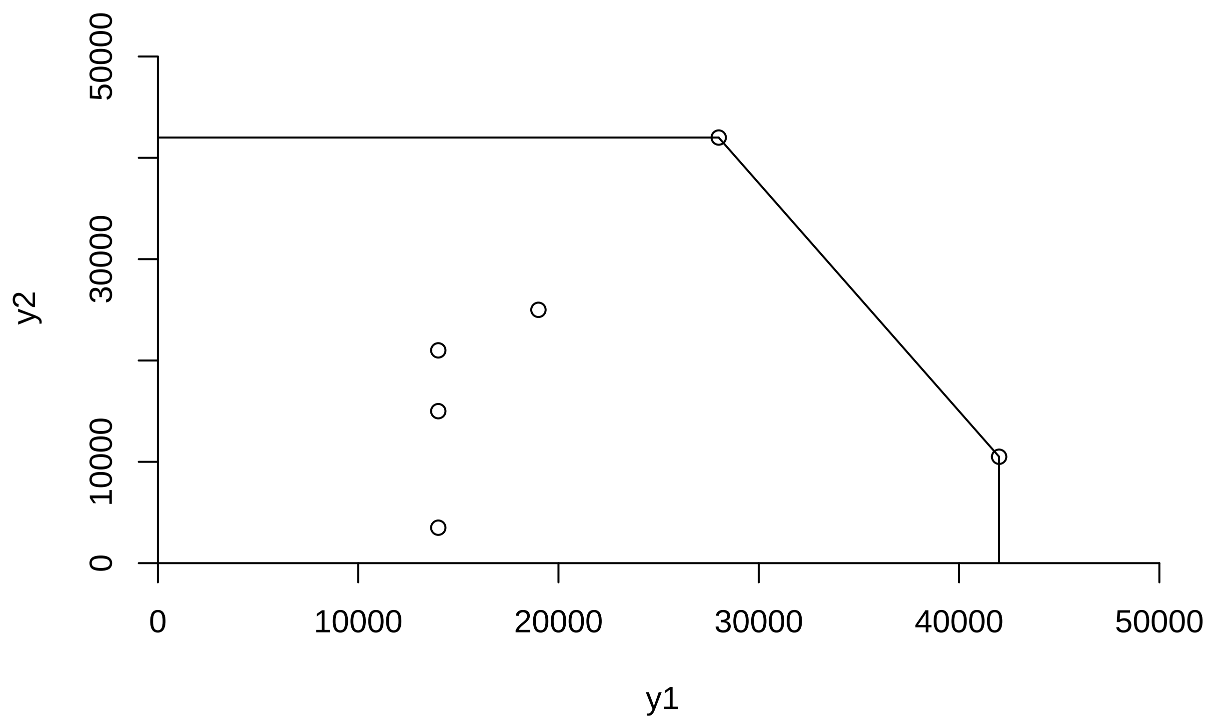
```
#Create the output matrix
output_matrix <- matrix(c(data_DEA$`Reimbursed Patient-Days`,data_DEA$`Privately Paid Patient-Days`),nc
#Lets see the output matrix
output_matrix
```

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

```
#plot the graph for Inputs and outputs for better understanding of distribution of data

dea.plot.isoquant(input_matrix[,1],input_matrix[,2]) #inputs
```




```
dea.plot.transform(output_matrix[,1],output_matrix[,2]) #outputs
```



Now, let's run the DEA analysis for different assumptions.

Here, we will use all the 6 model assumptions:

- **FDH**
- **CRS**
- **VRS**
- **IRS**
- **DRS**
- **FRH**

Starting with the first :

1.1 FDH (Free disposability hull, no convexity assumption)

```
#DEA input or output efficiency measures, peers, lambdas and slacks
FDH_efficiency <- dea(input_matrix,output_matrix,RTS = "FDH")
#Show the Efficiency
FDH_efficiency
```

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

```r
#Show the list of objects calculated
str(FDH_efficiency)
```

```
## List of 7
##  $ eff        : num [1:6] 1 1 1 1 1 1
##  $ objval     : num [1:6] 1 1 1 1 1 1
##  $ peers      : int [1:6] 1 2 3 4 5 6
##  $ lambda     : num [1:6, 1:6] 1 0 0 0 0 0 0 1 0 0 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:6] "L1" "L2" "L3" "L4" ...
##  $ RTS        : chr "fdh"
##  $ ORIENTATION: chr "in"
##  $ TRANSPOSE  : logi FALSE
##  - attr(*, "class")= chr "Farrell"
```

```r
#Show the peers
peers(FDH_efficiency)
```

```
##      peer1
## [1,]     1
## [2,]     2
## [3,]     3
## [4,]     4
## [5,]     5
## [6,]     6
```

```r
#Show the lambda
lambda(FDH_efficiency)
```

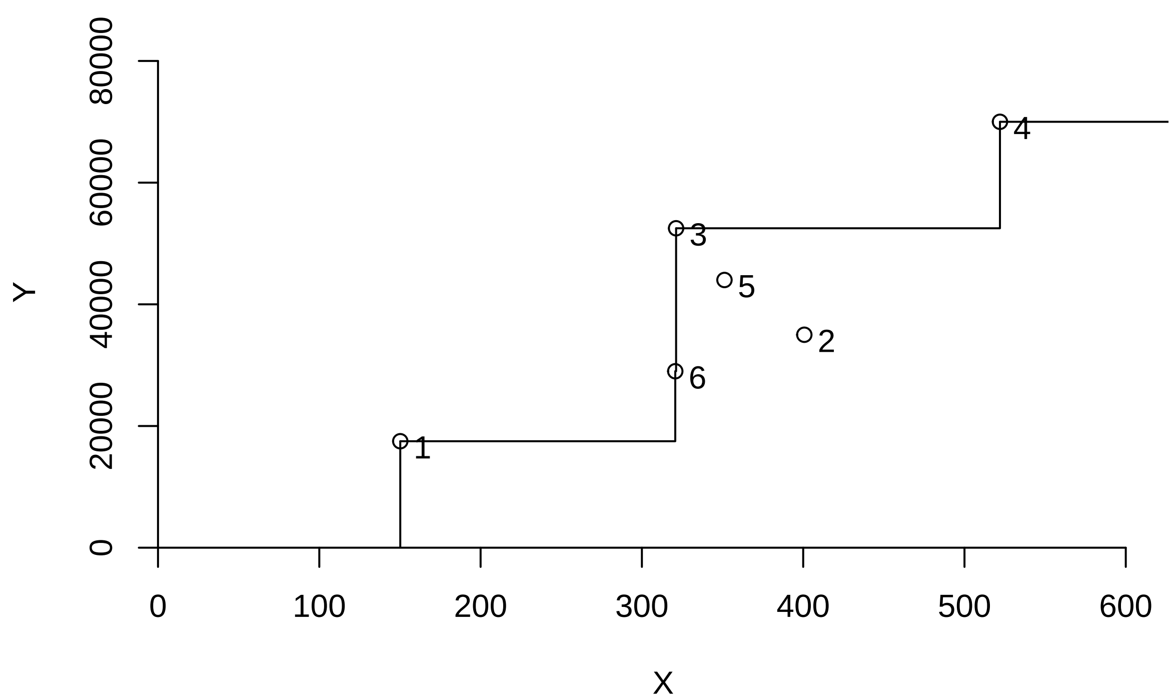
```
##      L1 L2 L3 L4 L5 L6
## [1,]  1  0  0  0  0  0
## [2,]  0  1  0  0  0  0
## [3,]  0  0  1  0  0  0
## [4,]  0  0  0  1  0  0
## [5,]  0  0  0  0  1  0
## [6,]  0  0  0  0  0  1
```

```r
#Add the Efficiency, Peers & Lambda values in the table
report1 <- cbind(data_DEA, FDH_efficiency$eff, FDH_efficiency$lambda, FDH_efficiency$peers)

#Name the columns of the table
colnames(report1)<- c(names(DMU_names),names(inputs), names(outputs),'Efficiency','Lambda1','Lambda2','L

#Show the table
report1
```

```
##          DMU Staff Hours per Day Supplies per Day Reimbursed Patient-Days
## 1 Facility 1                 150              0.2                   14000
## 2 Facility 2                 400              0.7                   14000
## 3 Facility 3                 320              1.2                   42000
```

```
## 4 Facility 4                  520               2.0                   28000
## 5 Facility 5                  350               1.2                   19000
## 6 Facility 6                  320               0.7                   14000
##   Privately Paid Patient-Days Efficiency Lambda1 Lambda2 Lambda3 Lambda4
## 1                        3500          1       1       0       0       0
## 2                       21000          1       0       1       0       0
## 3                       10500          1       0       0       1       0
## 4                       42000          1       0       0       0       1
## 5                       25000          1       0       0       0       0
## 6                       15000          1       0       0       0       0
##   Lambda5 Lambda6 Peers
## 1       0       0     1
## 2       0       0     2
## 3       0       0     3
## 4       0       0     4
## 5       1       0     5
## 6       0       1     6
```

```
#plot the graph for FDH Assumption
dea.plot(input_matrix,output_matrix,RTS="FDH",txt = rownames(report1))
```



Observations:

- Successfully able to determine the Peers and Lambdas under FDH Assumption.
- The results indicate that DMUs 1, 2, 3 , 4, 5 and 6 all are efficient.

1.2 CRS (Constant Return to Scale, convexity and free disposability)

```
#DEA input or output efficiency measures, peers, lambdas and slacks
CRS_efficiency <- dea(input_matrix,output_matrix,RTS = "CRS")
#Show the Efficiency
CRS_efficiency
```

```
## [1] 1.0000 1.0000 1.0000 1.0000 0.9775 0.8675
```

```
#Show the list of objects calculated
str(CRS_efficiency)
```

```
## List of 12
##  $ eff        : num [1:6] 1 1 1 1 0.977 ...
##  $ lambda     : num [1:6, 1:6] 1 0 0 0 0.2 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:6] "L1" "L2" "L3" "L4" ...
##  $ objval     : num [1:6] 1 1 1 1 0.977 ...
##  $ RTS        : chr "crs"
##  $ primal     : NULL
##  $ dual       : NULL
##  $ ux         : NULL
##  $ vy         : NULL
##  $ gamma      :function (x)
##  $ ORIENTATION: chr "in"
##  $ TRANSPOSE  : logi FALSE
##  $ param      : NULL
##  - attr(*, "class")= chr "Farrell"
```

```
#Show the peers
peers(CRS_efficiency)
```

```
##      peer1 peer2 peer3
## [1,]     1    NA    NA
## [2,]     2    NA    NA
## [3,]     3    NA    NA
## [4,]     4    NA    NA
## [5,]     1     2     4
## [6,]     1     2     4
```

```
#Show the lambda
lambda(CRS_efficiency)
```

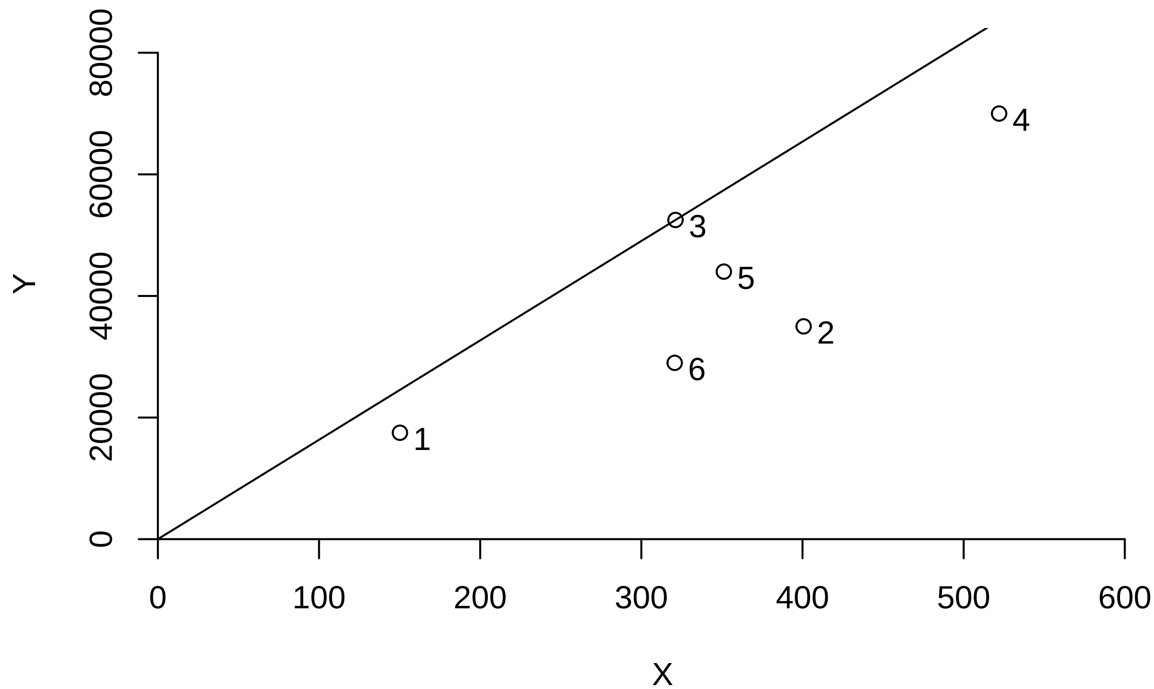
```
##             L1         L2 L3        L4
## [1,] 1.0000000 0.00000000  0 0.0000000
## [2,] 0.0000000 1.00000000  0 0.0000000
## [3,] 0.0000000 0.00000000  1 0.0000000
```

```
## [4,] 0.0000000 0.00000000  0 1.0000000
## [5,] 0.2000000 0.08048142  0 0.5383307
## [6,] 0.3428571 0.39499264  0 0.1310751
```

```
#Add the Efficiency & Lambda values in the table
report2 <- cbind(data_DEA, CRS_efficiency$eff, CRS_efficiency$lambda)
#Name the columns of the table
colnames(report2)<- c(names(DMU_names),names(inputs), names(outputs),'Efficiency','Lambda1','Lambda2','
#Sow the table
report2
```

```
##          DMU Staff Hours per Day Supplies per Day Reimbursed Patient-Days
## 1 Facility 1                 150              0.2                   14000
## 2 Facility 2                 400              0.7                   14000
## 3 Facility 3                 320              1.2                   42000
## 4 Facility 4                 520              2.0                   28000
## 5 Facility 5                 350              1.2                   19000
## 6 Facility 6                 320              0.7                   14000
##   Privately Paid Patient-Days Efficiency   Lambda1    Lambda2 Lambda3   Lambda4
## 1                        3500  1.0000000 1.0000000 0.00000000       0 0.0000000
## 2                       21000  1.0000000 0.0000000 1.00000000       0 0.0000000
## 3                       10500  1.0000000 0.0000000 0.00000000       1 0.0000000
## 4                       42000  1.0000000 0.0000000 0.00000000       0 1.0000000
## 5                       25000  0.9774987 0.2000000 0.08048142       0 0.5383307
## 6                       15000  0.8674521 0.3428571 0.39499264       0 0.1310751
##   Lambda5 Lambda6
## 1       0       0
## 2       0       0
## 3       0       0
## 4       0       0
## 5       0       0
## 6       0       0
```

```
#plot the graph for CRS Assumption
dea.plot(input_matrix,output_matrix,RTS="CRS",txt = rownames(report2))
```



Observations:

- Successfully able to determine the Peers and Lambdas under CRS Assumption.
- The results indicate that DMUs 1, 2, 3 and 4 are efficient. **DMU(5)** is only **97.7% efficient**, and **DMU(6)** is **86.7% efficient**.

1.3.VRS (Variable returns to scale, convexity and free disposability)

```
#DEA input or output efficiency measures, peers, lambdas and slacks
VRS_efficiency <- dea(input_matrix,output_matrix,RTS = "VRS")
#Show the Efficiency
VRS_efficiency
```

```
## [1] 1.0000 1.0000 1.0000 1.0000 1.0000 0.8963
```

```
#Show the list of objects calculated
str(VRS_efficiency)
```

```
## List of 12
##  $ eff        : num [1:6] 1 1 1 1 1 ...
```

```
##  $ lambda     : num [1:6, 1:6] 1 0 0 0 0 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:6] "L1" "L2" "L3" "L4" ...
##  $ objval     : num [1:6] 1 1 1 1 1 ...
##  $ RTS        : chr "vrs"
##  $ primal     : NULL
##  $ dual       : NULL
##  $ ux         : NULL
##  $ vy         : NULL
##  $ gamma      :function (x)
##  $ ORIENTATION: chr "in"
##  $ TRANSPOSE  : logi FALSE
##  $ param      : NULL
##  - attr(*, "class")= chr "Farrell"
```

```
#Show the peers
peers(VRS_efficiency)
```

```
##      peer1 peer2 peer3
## [1,]     1    NA    NA
## [2,]     2    NA    NA
## [3,]     3    NA    NA
## [4,]     4    NA    NA
## [5,]     5    NA    NA
## [6,]     1     2     5
```

```
#Show the lambda
lambda(VRS_efficiency)
```

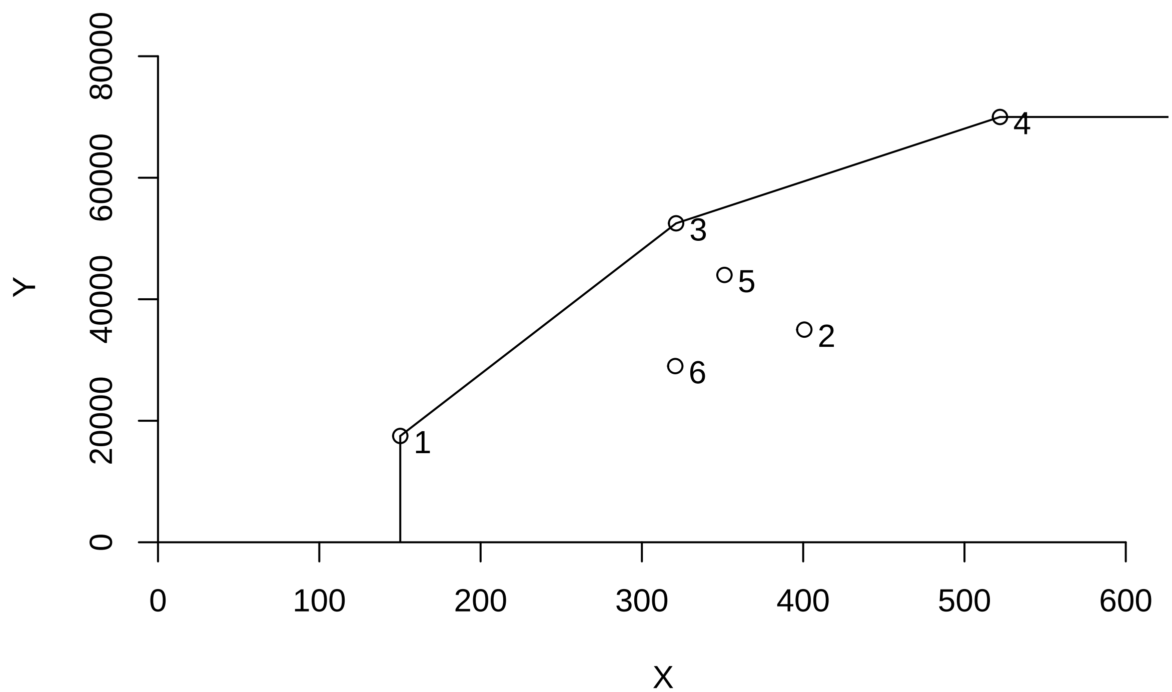
```
##             L1        L2 L3 L4        L5
## [1,] 1.0000000 0.0000000  0  0 0.0000000
## [2,] 0.0000000 1.0000000  0  0 0.0000000
## [3,] 0.0000000 0.0000000  1  0 0.0000000
## [4,] 0.0000000 0.0000000  0  1 0.0000000
## [5,] 0.0000000 0.0000000  0  0 1.0000000
## [6,] 0.4014399 0.3422606  0  0 0.2562995
```

```
#Add the Efficiency & Lambda values in the table
report3 <- cbind(data_DEA, VRS_efficiency$eff, VRS_efficiency$lambda)
#Name the columns of the table
colnames(report3)<- c(names(DMU_names),names(inputs), names(outputs),'Efficiency','Lambda1','Lambda2','L
#Show the table
report3
```

```
##          DMU Staff Hours per Day Supplies per Day Reimbursed Patient-Days
## 1 Facility 1                 150              0.2                   14000
## 2 Facility 2                 400              0.7                   14000
## 3 Facility 3                 320              1.2                   42000
## 4 Facility 4                 520              2.0                   28000
## 5 Facility 5                 350              1.2                   19000
## 6 Facility 6                 320              0.7                   14000
```

```
##   Privately Paid Patient-Days Efficiency   Lambda1   Lambda2 Lambda3 Lambda4
## 1                         3500  1.0000000 1.0000000 0.0000000       0       0
## 2                        21000  1.0000000 0.0000000 1.0000000       0       0
## 3                        10500  1.0000000 0.0000000 0.0000000       1       0
## 4                        42000  1.0000000 0.0000000 0.0000000       0       1
## 5                        25000  1.0000000 0.0000000 0.0000000       0       0
## 6                        15000  0.8963283 0.4014399 0.3422606       0       0
##     Lambda5 Lambda6
## 1 0.0000000       0
## 2 0.0000000       0
## 3 0.0000000       0
## 4 0.0000000       0
## 5 1.0000000       0
## 6 0.2562995       0
```

```
#plot the graph for VRS Assumption
dea.plot(input_matrix,output_matrix,RTS="VRS",txt = rownames(report3))
```



Observations:

- Successfully able to determine the Peers and Lambdas under VRS Assumption.
- The results indicate that DMUs 1, 2, 3, 4 and 5 are efficient. **DMU(6)** is only **89.6% efficient**.

1.4. IRS (Increasing returns to scale, convexity and free disposability)

```
#DEA input or output efficiency measures, peers, lambdas and slacks
IRS_efficiency <- dea(input_matrix,output_matrix,RTS = "IRS")
#Show the Efficiency
IRS_efficiency
```

```
## [1] 1.0000 1.0000 1.0000 1.0000 1.0000 0.8963
```

```
#Show the list of objects calculated
str(IRS_efficiency)
```

```
## List of 12
##  $ eff        : num [1:6] 1 1 1 1 1 ...
##  $ lambda     : num [1:6, 1:6] 1 0 0 0 0 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:6] "L1" "L2" "L3" "L4" ...
##  $ objval     : num [1:6] 1 1 1 1 1 ...
##  $ RTS        : chr "irs"
##  $ primal     : NULL
##  $ dual       : NULL
##  $ ux         : NULL
##  $ vy         : NULL
##  $ gamma      :function (x)
##  $ ORIENTATION: chr "in"
##  $ TRANSPOSE  : logi FALSE
##  $ param      : NULL
##  - attr(*, "class")= chr "Farrell"
```

```
#Show the peers
peers(IRS_efficiency)
```

```
##      peer1 peer2 peer3
## [1,]     1    NA    NA
## [2,]     2    NA    NA
## [3,]     3    NA    NA
## [4,]     4    NA    NA
## [5,]     5    NA    NA
## [6,]     1     2     5
```

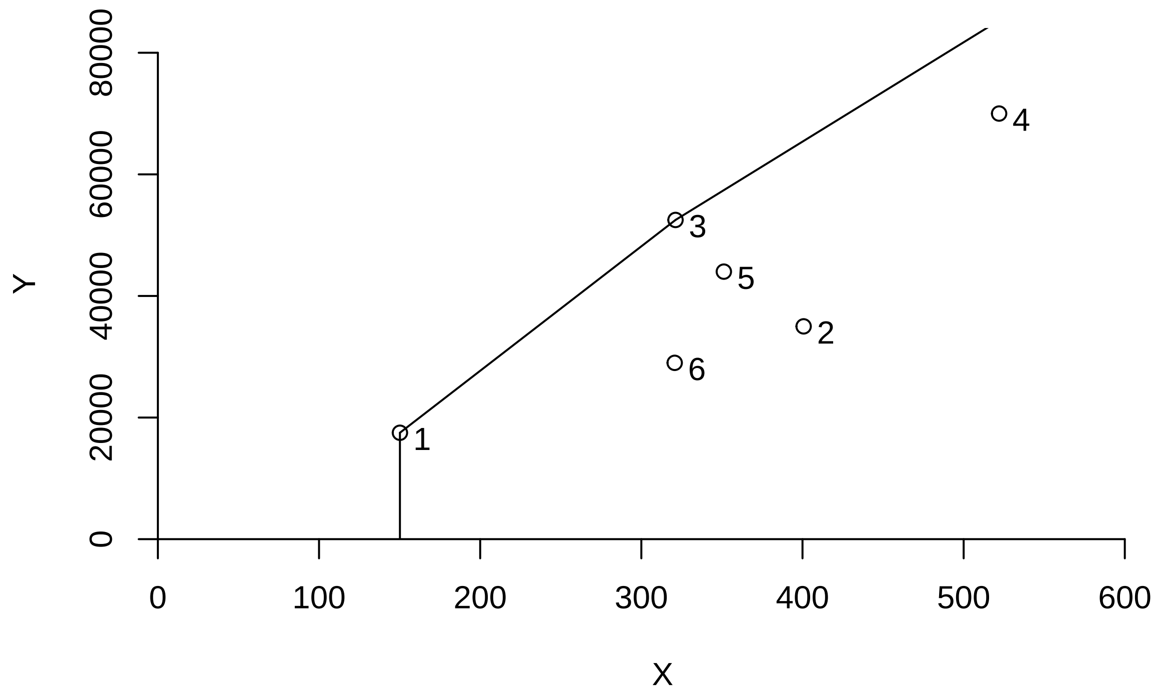
```
#Show the lambda
lambda(IRS_efficiency)
```

```
##             L1        L2 L3 L4        L5
## [1,] 1.0000000 0.0000000  0  0 0.0000000
## [2,] 0.0000000 1.0000000  0  0 0.0000000
## [3,] 0.0000000 0.0000000  1  0 0.0000000
## [4,] 0.0000000 0.0000000  0  1 0.0000000
## [5,] 0.0000000 0.0000000  0  0 1.0000000
## [6,] 0.4014399 0.3422606  0  0 0.2562995
```

```r
#Add the Efficiency & Lambda values in the table
report4 <- cbind(data_DEA, IRS_efficiency$eff, IRS_efficiency$lambda)
#Name the columns of the table
colnames(report4)<- c(names(DMU_names),names(inputs), names(outputs),'Efficiency','Lambda1','Lambda2','
#Show the table
report4
```

```
##          DMU Staff Hours per Day Supplies per Day Reimbursed Patient-Days
## 1 Facility 1                 150              0.2                   14000
## 2 Facility 2                 400              0.7                   14000
## 3 Facility 3                 320              1.2                   42000
## 4 Facility 4                 520              2.0                   28000
## 5 Facility 5                 350              1.2                   19000
## 6 Facility 6                 320              0.7                   14000
##   Privately Paid Patient-Days Efficiency   Lambda1   Lambda2 Lambda3 Lambda4
## 1                        3500  1.0000000 1.0000000 0.0000000       0       0
## 2                       21000  1.0000000 0.0000000 1.0000000       0       0
## 3                       10500  1.0000000 0.0000000 0.0000000       1       0
## 4                       42000  1.0000000 0.0000000 0.0000000       0       1
## 5                       25000  1.0000000 0.0000000 0.0000000       0       0
## 6                       15000  0.8963283 0.4014399 0.3422606       0       0
##     Lambda5 Lambda6
## 1 0.0000000       0
## 2 0.0000000       0
## 3 0.0000000       0
## 4 0.0000000       0
## 5 1.0000000       0
## 6 0.2562995       0
```

```r
#plot the graph for IRS Assumption
dea.plot(input_matrix,output_matrix,RTS="IRS",txt = rownames(report4))
```



Observations:

- Successfully able to determine the Peers and Lambdas under IRS Assumption.
- The results indicate that DMUs 1, 2, 3, 4 and 5 are efficient. **DMU(6)** is only **89.6% efficient**.

1.5. DRS (Decreasing returns to scale, convexity, down-scaling and free disposability)

```
#DEA input or output efficiency measures, peers, lambdas and slacks
DRS_efficiency <- dea(input_matrix,output_matrix,RTS = "DRS")
#Show the Efficiency
DRS_efficiency
```

```
## [1] 1.0000 1.0000 1.0000 1.0000 0.9775 0.8675
```

```
#Show the list of objects calculated
str(DRS_efficiency)
```

```
## List of 12
##  $ eff        : num [1:6] 1 1 1 1 0.977 ...
##  $ lambda     : num [1:6, 1:6] 1 0 0 0 0.2 ...
```

```
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:6] "L1" "L2" "L3" "L4" ...
##  $ objval     : num [1:6] 1 1 1 1 0.977 ...
##  $ RTS        : chr "drs"
##  $ primal     : NULL
##  $ dual       : NULL
##  $ ux         : NULL
##  $ vy         : NULL
##  $ gamma      :function (x)
##  $ ORIENTATION: chr "in"
##  $ TRANSPOSE  : logi FALSE
##  $ param      : NULL
##  - attr(*, "class")= chr "Farrell"
```

```
#Show the peers
peers(DRS_efficiency)
```

```
##      peer1 peer2 peer3
## [1,]     1    NA    NA
## [2,]     2    NA    NA
## [3,]     3    NA    NA
## [4,]     4    NA    NA
## [5,]     1     2     4
## [6,]     1     2     4
```

```
#Show the lambda
lambda(DRS_efficiency)
```

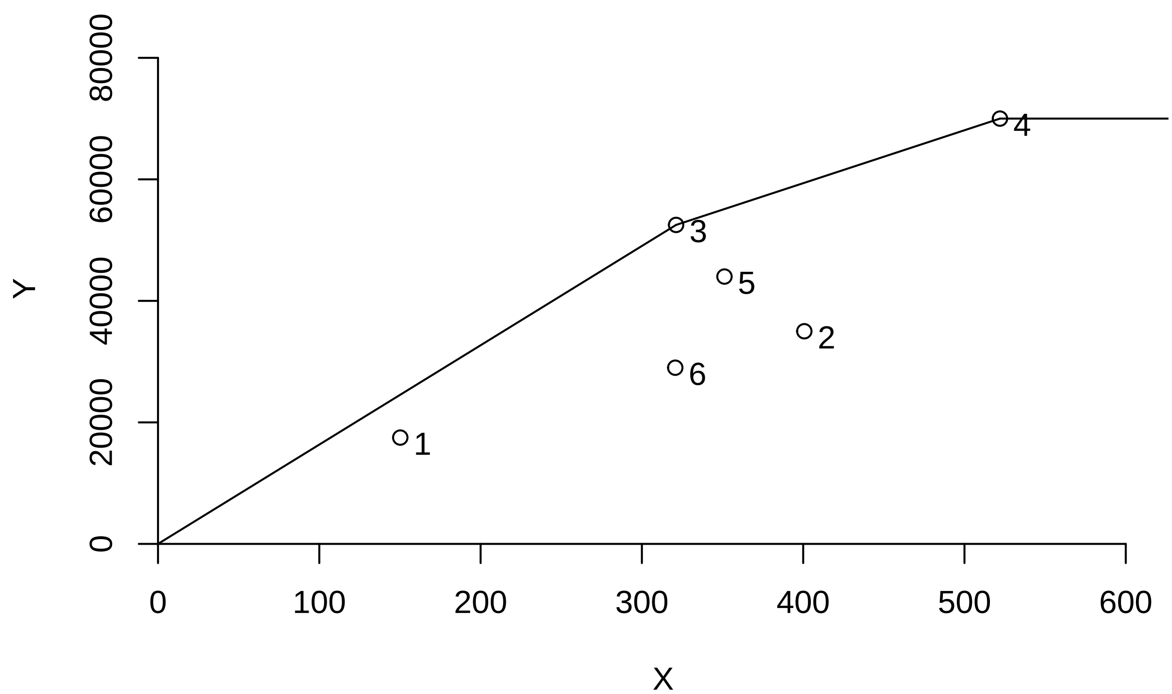
```
##             L1         L2 L3        L4
## [1,] 1.0000000 0.00000000  0 0.0000000
## [2,] 0.0000000 1.00000000  0 0.0000000
## [3,] 0.0000000 0.00000000  1 0.0000000
## [4,] 0.0000000 0.00000000  0 1.0000000
## [5,] 0.2000000 0.08048142  0 0.5383307
## [6,] 0.3428571 0.39499264  0 0.1310751
```

```
#Add the Efficiency, Peers & Lambda values in the table
report5 <- cbind(data_DEA, DRS_efficiency$eff, DRS_efficiency$lambda)
#Name the columns of the table
colnames(report5)<- c(names(DMU_names),names(inputs), names(outputs),'Efficiency','Lambda1','Lambda2','I
#Show the table
report5
```

```
##          DMU Staff Hours per Day Supplies per Day Reimbursed Patient-Days
## 1 Facility 1                 150              0.2                   14000
## 2 Facility 2                 400              0.7                   14000
## 3 Facility 3                 320              1.2                   42000
## 4 Facility 4                 520              2.0                   28000
## 5 Facility 5                 350              1.2                   19000
## 6 Facility 6                 320              0.7                   14000
##   Privately Paid Patient-Days Efficiency   Lambda1    Lambda2 Lambda3   Lambda4
```

```
## 1                          3500   1.0000000 1.0000000 0.00000000       0 0.0000000
## 2                         21000   1.0000000 0.0000000 1.00000000       0 0.0000000
## 3                         10500   1.0000000 0.0000000 0.00000000       1 0.0000000
## 4                         42000   1.0000000 0.0000000 0.00000000       0 1.0000000
## 5                         25000   0.9774987 0.2000000 0.08048142       0 0.5383307
## 6                         15000   0.8674521 0.3428571 0.39499264       0 0.1310751
##   Lambda5 Lambda6
## 1       0       0
## 2       0       0
## 3       0       0
## 4       0       0
## 5       0       0
## 6       0       0
```

```
#plot the graph for IRS Assumption
dea.plot(input_matrix,output_matrix,RTS="DRS",txt = rownames(report5))
```



Observations:

- Successfully able to determine the Peers and Lambdas under DRS Assumption.
- The results indicate that DMUs 1, 2, 3 and 4 are efficient. **DMU(5)** is only **97.7% efficient**, and **DMU(6)** is **86.7% efficient**.

1.6.FRH (Additivity (scaling up and down, but only with integers), and free disposability)

```
#DEA input or output efficiency measures, peers, lambdas and slacks
FRH_efficiency <- dea(input_matrix,output_matrix,RTS = "ADD")
#Show the Efficiency
FRH_efficiency
```

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

```
#Show the list of objects calculated
str(FRH_efficiency)
```

```
## List of 12
##  $ eff        : num [1:6] 1 1 1 1 1 1
##  $ lambda     : num [1:6, 1:6] 1 0 0 0 0 0 0 1 0 0 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:6] "L1" "L2" "L3" "L4" ...
##  $ objval     : num [1:6] 1 1 1 1 1 1
##  $ RTS        : chr "add"
##  $ primal     : NULL
##  $ dual       : NULL
##  $ ux         : NULL
##  $ vy         : NULL
##  $ gamma      :function (x)
##  $ ORIENTATION: chr "in"
##  $ TRANSPOSE  : logi FALSE
##  $ param      : NULL
##  - attr(*, "class")= chr "Farrell"
```

```
#Show the peers
peers(FRH_efficiency)
```

```
##      peer1
## [1,]     1
## [2,]     2
## [3,]     3
## [4,]     4
## [5,]     5
## [6,]     6
```

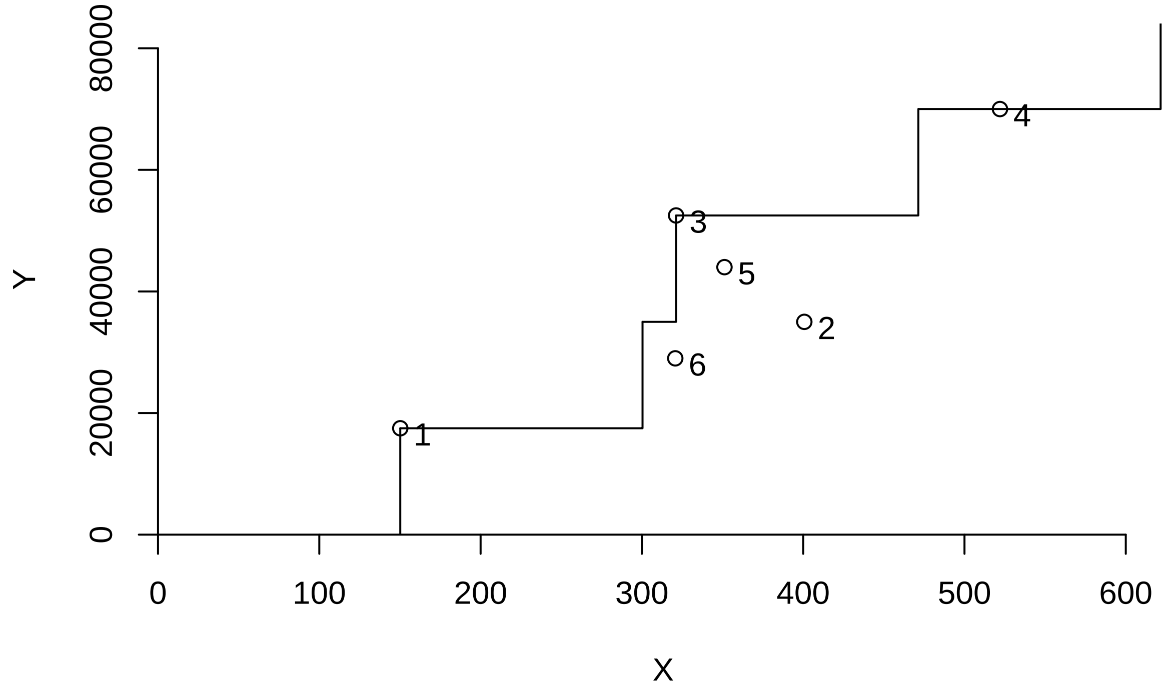
```
#Show the lambda
lambda(FRH_efficiency)
```

```
##      L1 L2 L3 L4 L5 L6
## [1,]  1  0  0  0  0  0
## [2,]  0  1  0  0  0  0
## [3,]  0  0  1  0  0  0
## [4,]  0  0  0  1  0  0
## [5,]  0  0  0  0  1  0
## [6,]  0  0  0  0  0  1
```

```r
#Add the Efficiency, Peers & Lambda values in the table
report6 <- cbind(data_DEA, FRH_efficiency$eff, FRH_efficiency$lambda)
#Name the columns of the table
colnames(report6)<- c(names(DMU_names),names(inputs), names(outputs),'Efficiency','Lambda1','Lambda2','L
#Sow the table
report6
```

```
##          DMU Staff Hours per Day Supplies per Day Reimbursed Patient-Days
## 1 Facility 1                 150              0.2                   14000
## 2 Facility 2                 400              0.7                   14000
## 3 Facility 3                 320              1.2                   42000
## 4 Facility 4                 520              2.0                   28000
## 5 Facility 5                 350              1.2                   19000
## 6 Facility 6                 320              0.7                   14000
##   Privately Paid Patient-Days Efficiency Lambda1 Lambda2 Lambda3 Lambda4
## 1                        3500          1       1       0       0       0
## 2                       21000          1       0       1       0       0
## 3                       10500          1       0       0       1       0
## 4                       42000          1       0       0       0       1
## 5                       25000          1       0       0       0       0
## 6                       15000          1       0       0       0       0
##   Lambda5 Lambda6
## 1       0       0
## 2       0       0
## 3       0       0
## 4       0       0
## 5       1       0
## 6       0       1
```

```r
#plot the graph for FDH Assumption
dea.plot(input_matrix,output_matrix,RTS="ADD",txt = rownames(report6))
```



Observations:

- Successfully able to determine the Peers and Lambdas under FRH Assumption.
- The results indicate that DMUs 1, 2, 3 , 4, 5 and 6 all are efficient.

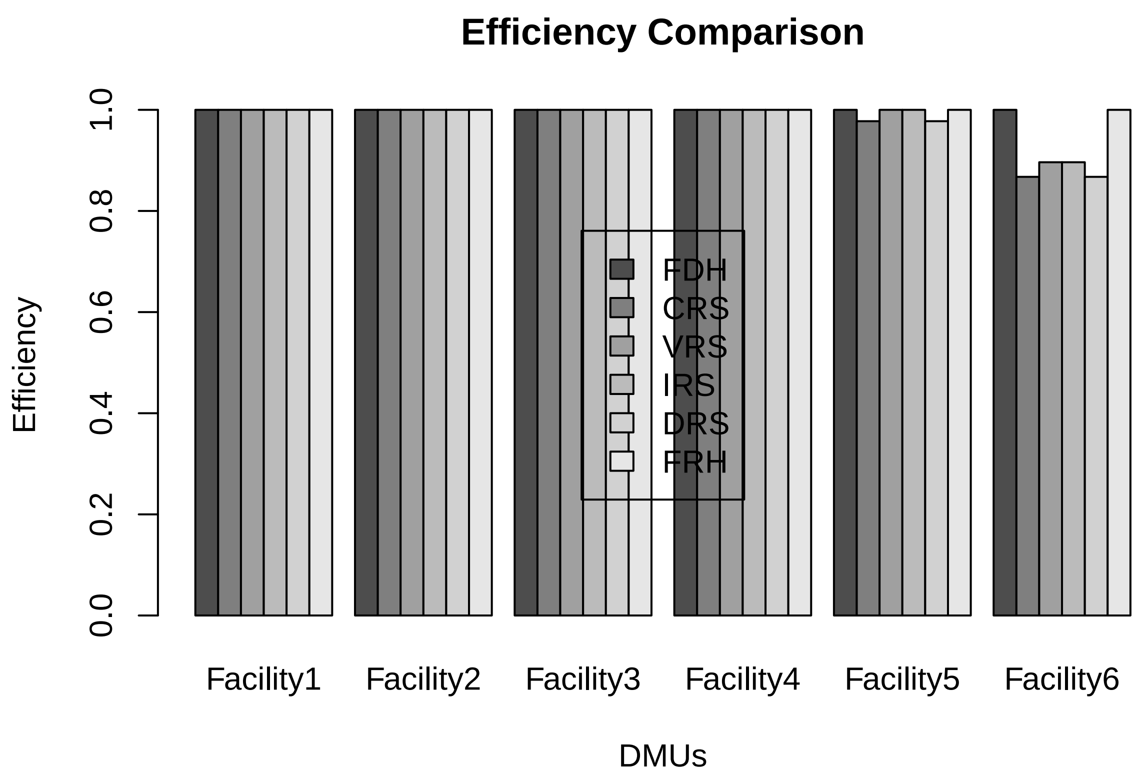
1.7. Compare and Contrast the Results

Let's compare the efficiency of all the DMUs for all the assumptions(tabular and graphical)

```
#Concatenate the Efficiency of all the DMU's

Efficiency_Report <- cbind(FDH_efficiency$eff,CRS_efficiency$eff,VRS_efficiency$eff,IRS_efficiency$eff,I

#Name the rows
rownames(Efficiency_Report) <- c("Facility1","Facility2","Facility3","Facility4","Facility5","Facility6

#Name the columns
colnames(Efficiency_Report) <- c ("FDH","CRS","VRS","IRS","DRS","FRH")

#See the result
Efficiency_Report
```

```
##              FDH       CRS       VRS       IRS       DRS FRH
## Facility1    1 1.0000000 1.0000000 1.0000000 1.0000000   1
## Facility2    1 1.0000000 1.0000000 1.0000000 1.0000000   1
## Facility3    1 1.0000000 1.0000000 1.0000000 1.0000000   1
## Facility4    1 1.0000000 1.0000000 1.0000000 1.0000000   1
## Facility5    1 0.9774987 1.0000000 1.0000000 0.9774987   1
## Facility6    1 0.8674521 0.8963283 0.8963283 0.8674521   1
```

```
#plot the graph

barplot(t(Efficiency_Report),xlab = "DMUs", ylab="Efficiency",beside=TRUE,main = "Efficiency Comparison
                           inset = c(- 0.10, 0)) )
```



Results:

- Successfully able to use 'Benchmarking' library for DEA analysis by comparing performance measures.
- Successfully performed all 6 model assumptions for DEA analysis for all facilities(1-6). Based on the comparison , we can say that:-
- **Facility 1,2,3,4** are **fully efficient** for all the assumptions.
- **Facility 5** is fully efficient for FDH, VRS, IRS and FRH assumptions. For assumptions **DRS and CRS, it is 97.7% efficient**.
- **Facility 6** is fully efficient for FDH and FRS assumptions. **For CRS and DRS assumptions, it is 86.7% efficient. For IRS and VRS assumptions, it is 89.6% efficient**.

2.The Research and Development Division of the Emax Corporation has developed three new products. A decision now needs to be made on which mix of these products should be produced. Management wants primary consideration given to three factors: total profit, stability in the workforce, and achieving an increase in the company's earnings next year from the $75 million achieved this year. In particular, using the units given in the following table, they want to

Maximize Z = P - 6C - 3D, where

P = total (discounted) profit over the life of the new products, C = change (in either direction) in the current level of employment, D = decrease (if any) in next year's earnings from the current year's level.

The amount of any increase in earnings does not enter into Z, because management is concerned primarily with just achieving some increase to keep the stockholders happy. (It has mixed feelings about a large increase that then would be difficult to surpass in subsequent years.)

The impact of each of the new products (per unit rate of production) on each of these factors is shown in the following table:

	Unit Contribution				
	Product:				
Factor	**1**	**2**	**3**	**Goal**	**Units**
Total profit	20	15	25	Maximize	Millions of dollars
Employment level	6	4	5	= 50	Hundreds of employees
Earnings next year	8	7	5	≥ 75	Millions of dollars

1) Define y1+ and y1-, respectively, as the amount over (if any) and the amount under (if any) the employment level goal. Define y2+ and y2- in the same way for the goal regarding earnings next year. Define x1, x2, and x3 as the production rates of Products 1, 2, and 3, respectively. With these definitions, use the goal programming technique to express y1+, y1-, y2+ and y2- algebraically in terms of x1, x2, and x3. Also express P in terms of x1, x2, and x3.

Solution:

This problem has all the Goals which are roughly comparable importance. Hence, it is a **non preemptive goal programming model**.

The Emax corporation problem includes all three possible types of goals: an upper, one-sided goal (Total profit); a two-sided goal (Employment level); and a lower, one-sided goal (Earnings Next year). Let the decision variables be x1, x2, x3 be the production rates of products 1, 2, and 3, respectively. Therefore, Total Profit (P) can be expressed in terms of x1, x2 and x3 as:

Maximize = 20x1+15x2+25x3

Similarly, Employment level and Next year Earnings goals can be expressed as:

6x1+4x2+5x3 =50

x1+7x2+5x3 >=75

As mentioned above , goal of total profit is to maximize it using the employment level and next years earnings goals as constrains, so these goals can be stated as

Max z: 20x1+15x2+25x3

s.t.: 6x1+4x2+5x3 =50

8x1+7x2+5x3 >=75

Now, we need to use **auxilliary variables**,To express this *overall objective mathematically**,(extra variables that are helpful for formulating the model) y1 and y2, defined as follows:

y1=6x1+4x2+5x3-50 (Employment Level minus the target)

y2=8x1+7x2+5x3-75 (Earnings Next Year minus the Target)

Since, each yi can be either positive or negative,we will replace each one by the difference of two non negative variables:

y1=y1p - y1m, where y1p , y1m >=0

y2=y2p - y2m, where y2p , y2m >=0

y1p represents the penalty for **employment level goal exceeding 50** and **y1m** is the penalty for employment level **goal decreasing below 50**.

Similarly, **y2m** will be the penalty for **not reaching the next year earnings** and **y2p** will be **exceeding the next year earnings.**

---

2) Express management's objective function in terms of x1, x2, x3, y1+, y1- , y2+ and y2-.
3) Formulate and solve the linear programming model. What are your findings?

Now, we can apply the new auxiliary vairables and therefore the overall management's objective function will be expressed mathematically as (maximizing the profit and subtracting the penalties) :

Max z: 20x1+15x2+25x3-6y1p-6y1m-3y2m;

s.t.: 6x1+4x2+5x3-y1p+y1m =50

8x1+7x2+5x3-y2p+y2m >=75

Since, there is no penalty for exceeding the earnings next year, so y2p should not appear in the objective function.

Now,let's solve this Linear programming model using lpSolveAPI on R.

2.1. Solving the Emax linear programming model using lpsolveAPI

```
# loading the lpsolveAPI library
library(lpSolveAPI)

## Let us set up the Emax problem with 7 decision variables, and 2 constraints.
lprec <- make.lp(2, 7)

## Set the maximization objective function
set.objfn(lprec, c(20, 15, 25, -6, -6, 0, -3))
lp.control(lprec,sense='max')
```

```
## $anti.degen
## [1] "fixedvars" "stalling"
##
## $basis.crash
## [1] "none"
##
## $bb.depthlimit
## [1] -50
##
## $bb.floorfirst
## [1] "automatic"
##
## $bb.rule
## [1] "pseudononint" "greedy"       "dynamic"      "rcostfixing"
##
## $break.at.first
## [1] FALSE
##
## $break.at.value
## [1] 1e+30
##
## $epsilon
##       epsb       epsd      epsel     epsint epsperturb   epspivot
##      1e-10      1e-09      1e-12      1e-07      1e-05      2e-07
##
## $improve
## [1] "dualfeas" "thetagap"
##
## $infinite
## [1] 1e+30
##
## $maxpivot
## [1] 250
##
## $mip.gap
## absolute relative
##    1e-11    1e-11
##
## $negrange
## [1] -1e+06
##
## $obj.in.basis
## [1] TRUE
##
## $pivoting
## [1] "devex"    "adaptive"
##
## $presolve
## [1] "none"
##
## $scalelimit
## [1] 5
##
## $scaling
```

```
## [1] "geometric"   "equilibrate" "integers"
##
## $sense
## [1] "maximize"
##
## $simplextype
## [1] "dual"   "primal"
##
## $timeout
## [1] 0
##
## $verbose
## [1] "neutral"
```

```r
# Set values for the rows (set the Left hand side constraints)
set.row(lprec, 1, c(6, 4, 5, -1 , 1, 0, 0), indices = c(1, 2, 3, 4, 5, 6, 7))
set.row(lprec, 2, c(8, 7, 5,0,0,-1,1), indices = c(1, 2, 3, 4, 5, 6, 7))

# Set the right hand side values
rhs <- c(50, 75)
set.rhs(lprec, rhs)

# Set constraint type and set variable types and bound
set.constr.type(lprec, c("=", ">="))
set.bounds(lprec, lower = rep(0, 7))

# Naming the decision variables (column) and constraints (rows)

lp.rownames <- c("EmploymentLevelGoal", "NextYearEarningsGoal")
lp.colnames <- c("x1", "x2", "x3","y1p", "y1m", "y2p","y2m")
dimnames(lprec) <- list(lp.rownames, lp.colnames)

# View the linear program object
lprec
```

```
## Model name:
##                         x1    x2    x3   y1p   y1m   y2p   y2m
## Maximize                20    15    25    -6    -6     0    -3
## EmploymentLevelGoal      6     4     5    -1     1     0     0    =  50
## NextYearEarningsGoal     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
```

```r
# Save this into a file
write.lp(lprec, filename = "emax.lp", type = "lp")

# Now solve the model
solve(lprec)
```

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

```
# Show the value of objective function, variables, constraints and slack
get.objective(lprec)
```

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

```
get.variables(lprec)
```

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

```
get.constraints(lprec)
```

```
## [1] 50 75
```

```
get.constraints(lprec) - rhs
```

```
## [1] 1.421085e-14 0.000000e+00
```

Also,We can now read the lp formulation using an lp file and solve it. I am using the same lp file which I have saved above.

2.2 Solving the lp file

```
emax <- read.lp("emax.lp")     # create an lp object x
solve(emax)                    # Solution
```

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

```
get.objective(emax)            # get objective value
```

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

```
get.variables(emax)            # get values of decision variables
```

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

```
get.constraints(emax)          # get constraints
```

```
## [1] 50 75
```

Observations:

- After applying the simplex method to the above formulation yields an optimal solution **x1 = 0, x2 = 0, x3 = 15, y1p = 25, y1m = 0, y2p = 0, y2m = 0.**
- We can see that, **y1 = 25 and y2 = 0**, so the second goal of Next years Earning is fully satisfied, but the **employment level goal of 50 is exceeded by 25 (2500 Employees)**.
- **The resulting penalty for deviating from the goals is 150**.Therefore, value for the **objective function is 225**.
- This solution is **not feasible**.

2.3 Streamlined Procedure for Preemptive Goal Programming

Since , there is no priority given by the management and in order to get a feasible solution we can use The **Streamlined Procedure for Preemptive Goal Programming.**

Now we need to re-formulate the objective function by assigning different penalty weights.A very large positive number can be substituted for penality.(here I have used 1000):

```
Max z: 20x1+15x2+25x3-6000y1p-6000y1m-3000y2m;

          s.t.: 6x1+4x2+5x3-y1p+y1m =50

                8x1+7x2+5x3-y2p+y2m >=75
```

Now, we will solve the lp file with the above mentioned equations in R and validate the result.

```
abc <- read.lp("New_Formulated_emax.lp")     # create an lp object x
abc
```

```
## Model name:
##                           x1     x2     x3    y1p    y1m    y2m    y2p
## Maximize                  20     15     25  -6000  -6000  -3000      0
## EmploymentLevelGoal        6      4      5     -1      1      0      0   =  50
## NextYearEarningsGoal       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(abc)                  # Solution
```

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

```
get.objective(abc)          # get objective value
```

```
## [1] 208.3333
```

```
get.variables(abc)          # get values of decision variables
```

```
## [1] 0.000000 8.333333 3.333333 0.000000 0.000000 0.000000 0.000000
```

```
get.constraints(abc)
```

```
## [1] 50 75
```

Results:

- I was able to understand that Goal programming is an approach of deriving a best possible 'satisfactory' level of goal attainment by multi-objective optimization problem that balances a trade-off in conflicting by solving objectives.
- In the first objective function **Employment level goal of 50 was exceeded by 25 (2500 Employees)** and solution was **not feasible**.
- After applying streamlined procedure we are getting a **feasible solution**.
- After applying this method to the above formulation yields an optimal solution **x1 = 0, x2 = 8.4(relatively), x3 = 3.4(relatively), y1p = 0, y1m = 0, y2p = 0, y2m = 0.**
- We can see that, **y1 = 0 and y2 = 0**, so the both goals are **fully satisfied** now.
- The value for the **objective function is 208.3333**.
