Assignment - Module 8

Sai Gautham Sabhavathu

2022-10-29

1. Formulate and perform DEA analysis under all DEA assumptions of FDH, CRS, VRS, IRS, DRS, and FRH.
2. Determine the Peers and Lambdas under each of the above assumptions

```
#Loading required libraries
library("Benchmarking")
```

```
## Warning: package 'Benchmarking' was built under R version 4.1.3

## Loading required package: lpSolveAPI

## Warning: package 'lpSolveAPI' was built under R version 4.1.3

## Loading required package: ucminf

## Warning: package 'ucminf' was built under R version 4.1.3

## Loading required package: quadprog

##
## Loading Benchmarking version 0.30h, (Revision 244, 2022/05/05 16:31:31) ...

## Build 2022/05/05 16:31:40
```

```
data.df.values <- matrix(c("Facility 1","Facility 2","Facility 3","Facility 4","Facility 5", "Facility 
                150,400,320,520,350,320,
                0.2,0.7,1.2,2.0,1.2,0.7,
                14000,14000,42000,28000,19000,14000,
                3500,21000,10500,42000,25000,15000), ncol=5, byrow=F)

colnames(data.df.values) <- c("DMU", "Staff_Hours_Per_Day","Supplies_Per_Day","Reimbursed_Patient_Days"

table.df <- as.table(data.df.values)
table.df
```

```
##   DMU        Staff_Hours_Per_Day Supplies_Per_Day Reimbursed_Patient_Days
## A Facility 1 150                 0.2              14000
## B Facility 2 400                 0.7              14000
## C Facility 3 320                 1.2              42000
## D Facility 4 520                 2                28000
## E Facility 5 350                 1.2              19000
## F Facility 6 320                 0.7              14000
##   Privately_Paid_Patient_Days
## A 3500
## B 21000
## C 10500
## D 42000
## E 25000
## F 15000
```

Calculating Constant Returns to Scale (CRS)

```
x <- matrix(c(150,400,320,520,350,320,
            0.2,0.7,1.2,2.0,1.2,0.7),ncol=2)

y <- matrix(c(14000,14000,42000,28000,19000,14000,
                3500,21000,10500,42000,25000,15000),ncol=2)

colnames(y) <- c("Reimbursed_Patient_Days","Privately_Paid_Patient_Days")

colnames(x) <- c("Staff_Hours_Per_Day","Supplies_Per_Day")

D_E_A_crs<-dea(x, y, RTS = "crs")
D_E_A_crs
```

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

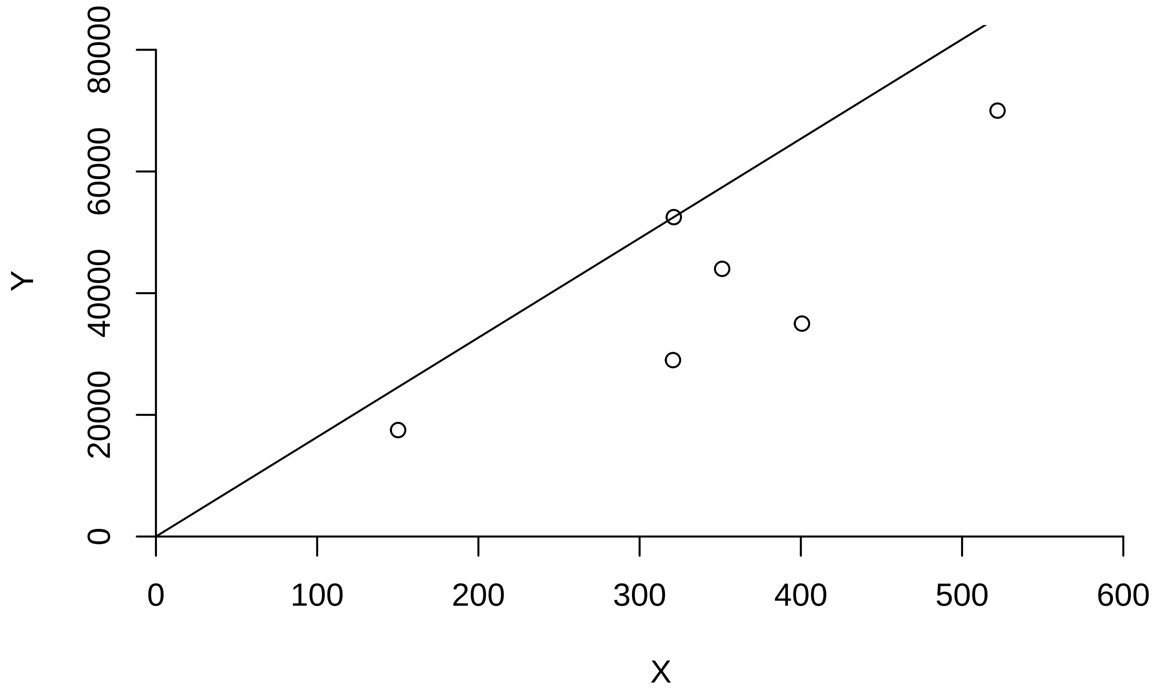
```
peers(D_E_A_crs)
```

```
##      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
```

```
lambda(D_E_A_crs)
```

```
##              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
```

```
dea.plot(x, y, RTS='crs')
```



Calculating Decreasing Returns to Scale (DRS)

```
D_E_A_drs <- dea(x, y, RTS = "drs")
D_E_A_drs
```

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

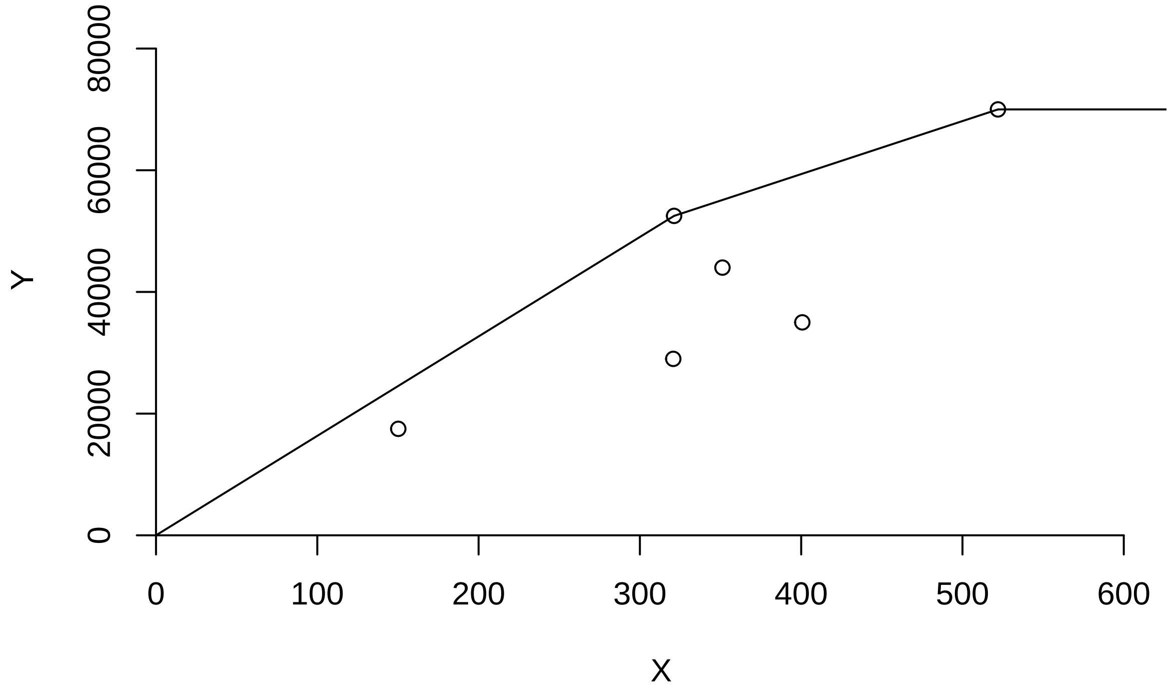
```
peers(D_E_A_drs)
```

```
##      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
```

```
lambda(D_E_A_drs)
```

```
##             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
```

```
dea.plot(x,y,RTS="drs")
```



Calculating Increasing Returns to Scale (IRS)

```
D_E_A_irs <- dea(x, y, RTS = "irs")
D_E_A_irs
```

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

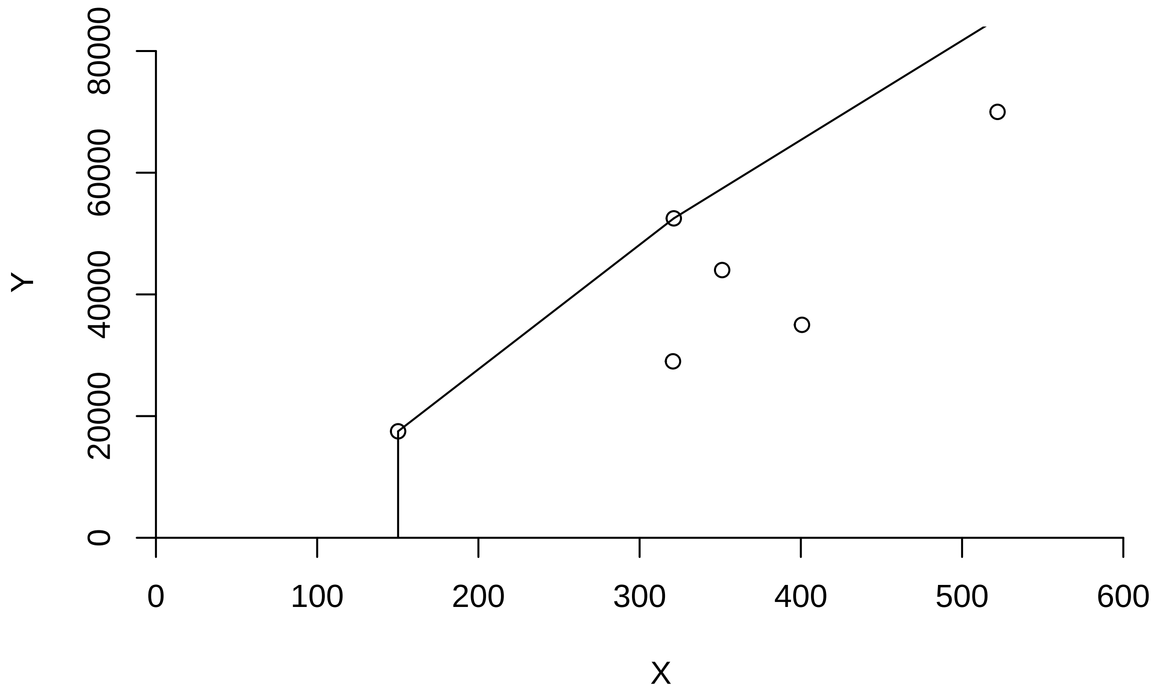
```
peers(D_E_A_irs)
```

```
##      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
```

```
lambda(D_E_A_irs)
```

```
##             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
```

```
dea.plot(x,y,RTS="irs")
```



Calculating Variable Returns to Scale (VRS)

```
D_E_A_vrs <- dea(x, y, RTS = "vrs")
D_E_A_vrs
```

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

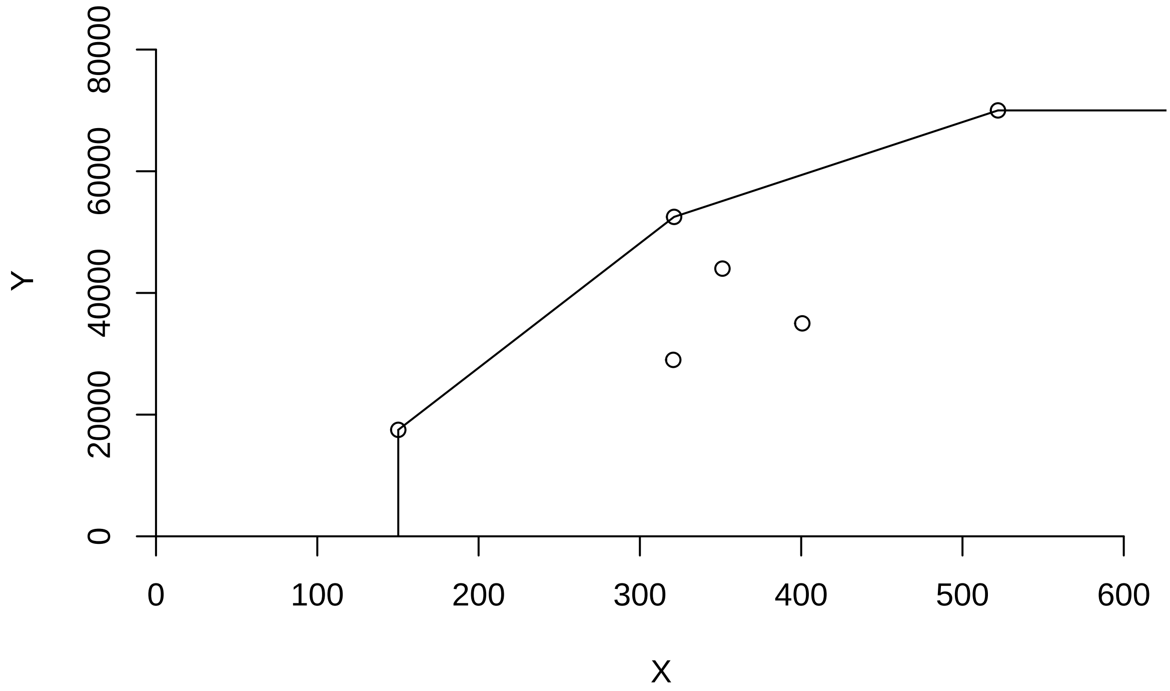
```
peers(D_E_A_vrs)
```

```
##      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
```

```
lambda(D_E_A_vrs)
```

```
##              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
```

```
dea.plot(x,y,RTS="vrs")
```



Calculating Free Disposal Hull (FDH)

```
D_E_A_fdh <- dea(x, y, RTS = "fdh")
D_E_A_fdh
```

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

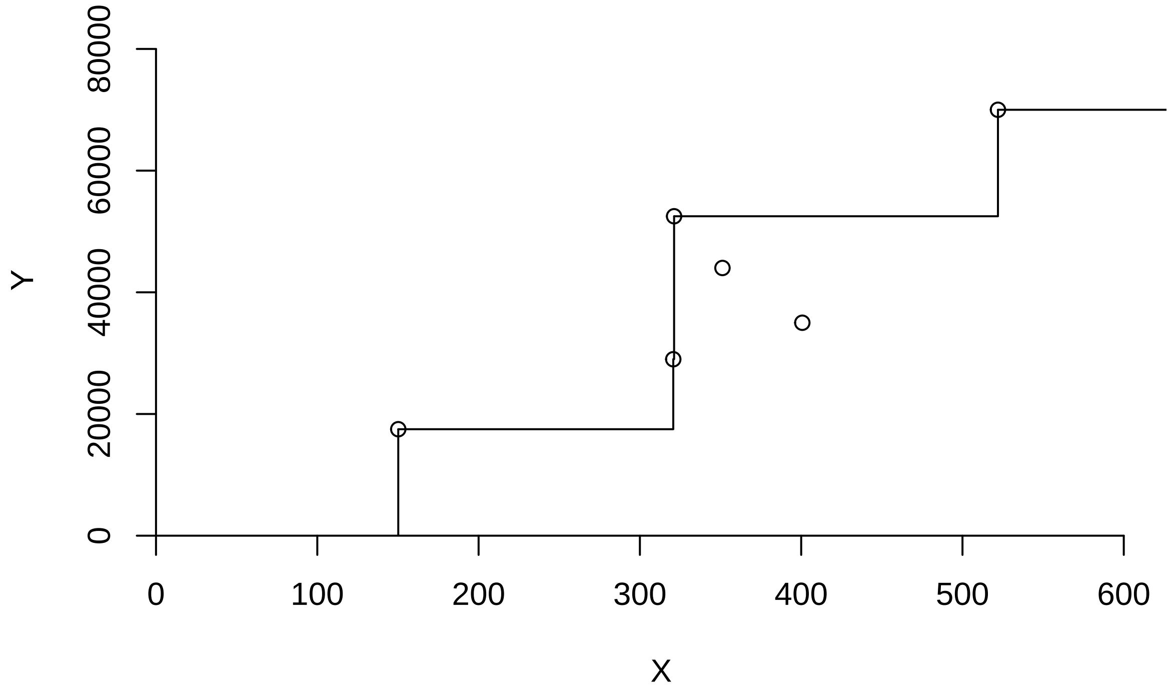
```
peers(D_E_A_fdh)
```

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

```
lambda(D_E_A_fdh)
```

```
##       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
```

```
dea.plot(x,y,RTS="fdh")
```



Calculating Free Replicability Hull (FRH)

```
#FRH is calculated by specifying RTS = "add"
D_E_A_frh <- dea(x, y, RTS = "add")
D_E_A_frh
```

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

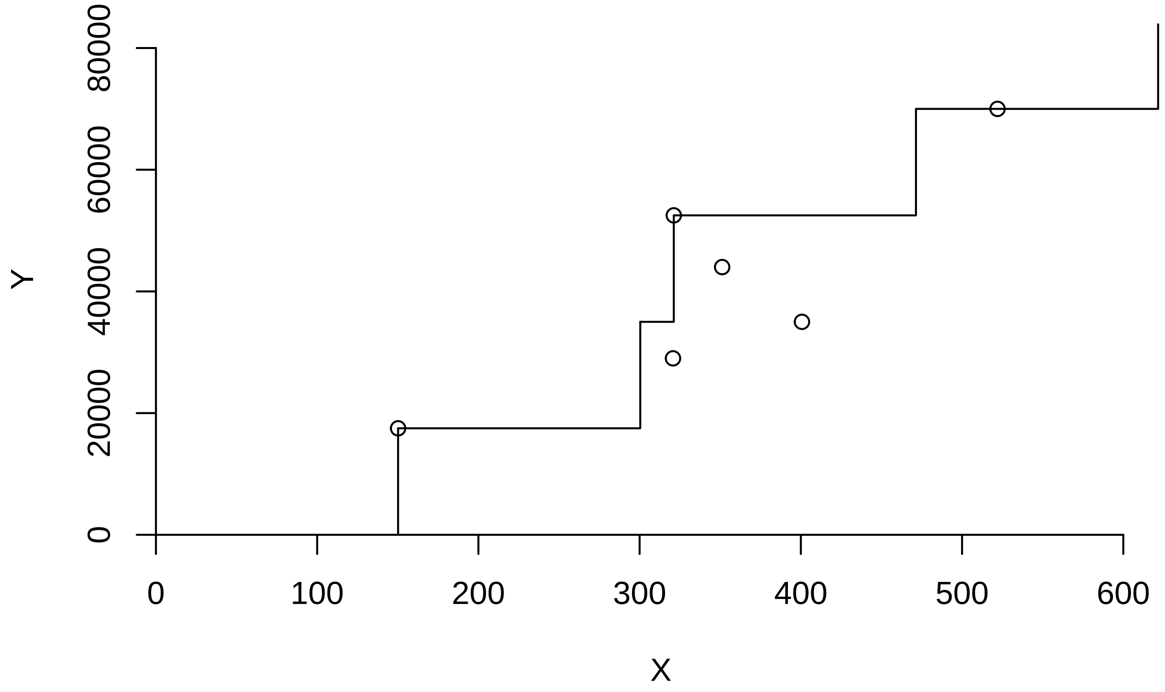
```
peers(D_E_A_frh)
```

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

```
lambda(D_E_A_frh)
```

```
##      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
```

```
dea.plot(x,y,RTS="add")
```



3. Summarize your results in a tabular format

```
#Summary of Results (Inefficient DMUs)
data.df.summarise.inefficient <- matrix(c("CRS","DRS","IRS","VRS","FDH","FRH",
2,2,1,1,0,0,
"Facility 5 & 6", "Facility 5 & 6","Facility 6", "Facility 6", "-","-",
"97.75% & 86.7%","97.75% & 86.7%","89.63%","89.63%","-","-",
"Facility 1, 2 & 4","Facility 1, 2 & 4","Facility 1, 2 & 5","Facility 1, 2 & 5","-","-",
"0.2, 0.08, 0.54 and 0.34, 0.4, 0.13", "0.2, 0.08, 0.54 and 0.34, 0.4, 0.13", "0.4, 0.34 and 0.26", "0.4


colnames(data.df.summarise.inefficient) <- c("RTS","Count_Inefficient_DMUs","Name_DMUs","%_Inefficiency

as.table(data.df.summarise.inefficient)
```

```
##   RTS Count_Inefficient_DMUs Name_DMUs      %_Inefficiency Peers
## A CRS 2                      Facility 5 & 6 97.75% & 86.7% Facility 1, 2 & 4
## B DRS 2                      Facility 5 & 6 97.75% & 86.7% Facility 1, 2 & 4
## C IRS 1                      Facility 6     89.63%         Facility 1, 2 & 5
## D VRS 1                      Facility 6     89.63%         Facility 1, 2 & 5
## E FDH 0                      -              -              -
## F FRH 0                      -              -              -
##   Lambda
## A 0.2, 0.08, 0.54 and 0.34, 0.4, 0.13
## B 0.2, 0.08, 0.54 and 0.34, 0.4, 0.13
## C 0.4, 0.34 and 0.26
## D 0.4, 0.34 and 0.26
## E -
## F -
```

```
#Summary of Results (Efficient DMUs)
data.df.summarise.efficient <- matrix(c("CRS","DRS","IRS","VRS","FDH","FRH",
"Facility 1, 2, 3 & 4","Facility 1, 2, 3 & 4","Facility 1, 2, 3, 4 & 5", "Facility 1, 2, 3, 4 & 5", "All

colnames(data.df.summarise.efficient) <- c("RTS", "Efficient_DMUs")

as.table(data.df.summarise.efficient)
```

```
##   RTS Efficient_DMUs
## A CRS Facility 1, 2, 3 & 4
## B DRS Facility 1, 2, 3 & 4
## C IRS Facility 1, 2, 3, 4 & 5
## D VRS Facility 1, 2, 3, 4 & 5
## E FDH All DMUs
## F FRH All DMUs
```

4. Compare and contrast the above results It is very important to understand the difference between the scales. FDH and FRH are considered as non-parametric methods to measure the efficiency of the DMUs. DRS,VRS and IRS are the dispersion scales which help us knowing what to increase and decrease based on the input deployment.

FDH - Free Disposal Hull The results from FDH clearly indicates that all the DMUs are efficient. FDH allows the scale to capture even the tiniest level of efficiency and also as there is no convexity assumption.

CRS - Constant Returns to Scale Constant returns to scale occur when increasing the number of inputs leads to an equivalent increase in the output. It helps us to know if the DMUs can be scaled up or down. DMUs 1,2,3 and 4 are clearly efficient from the results whereas DMU 5 and 6 are 97.75% and 86.7% efficient respectively. The peer units for DMU 5 are 1,2 and 4 and their relative weights are 0.2, 0.08 and 0.54 respectively. The peer units units for DMU 6 are 1,2 and 4 and their relative weights are 0.34,0.39 and 0.131. From the above observations, it can be interpreted that the DMUs 1,2,3 and 4 can be scaled up.

VRS - Variable Returns to Scale Variable returns to scale helps to estimate efficiencies whether an increase or decrease in input or outputs does not result in a proportional change in the outputs or inputs respectively. The results from VRS indicate that the DMus 1,2,3,4 and 5 are efficient whereas DMU 6 is 89.63% efficient. The peer units for DMU 6 are 1,2 and 5 and their relative weights are 0.4,0.34 and 0.25 respectively.

IRS - Increasing Returns to Scale An increasing returns to scale occurs when the output increases by a larger proportion than the increase in inputs during the production process. It helps the firm to understand if they can expeditiously increase the scale of operation by looking at the efficiency scores. The results from IRS clearly indicate that the DMUs 1,2,3,4 and 5 are efficient and DMU 6 has an efficiency of only 89.63%. The peer units for DMU 6 are 1,2 and 5 and their relative weights are 0.4,0.34 and 0.25.

DRS - Decreasing Returns to Scale A decreasing returns to scale occurs when the proportion of output is less than the desired increased input during the production process. DRS tells us if there are any possible DMUs where we can scale the operations. This is done by looking at the inefficient DMUs. In this particular case, the inefficient DMUs are 5 and 6, as their efficiencies are 97.75% and 86.75% respectively. The peer units for DMU 5 are 1,2 and 4 and their relative weights are 0.2,0.08 and 0.54 respectively. The peer units for DMU 6 are 1,2 and 4 with relative weights of 0.34,0.4 and 0.13 respectively.

FRH - Free Replicability Hull The FRH results clearly indicates that all the DMUs are efficient. FRH allows the scale to capture the tiniest level of efficiency which is free from replication and disposal.

Final Conclusion :- - Facilities 1,2,3 and 4 are fully efficient for all the assumptions and Facilities 5 and 6 are not efficient. - Facility 5 is fully efficient for FDH,VRS,IRS and FRH assumptions. - Facility 6 is fully efficient for the FDH and FRS assumptions. - For Facility 6, CRS and DRS assumptions are 86.7% efficient - We can see that there is 97.7% efficiency for the DRS assumption. - Finally, for Facility 6, IRS and VRS assumptions are 89.63% efficient.