Assignment 4

Kiran Kour

2022-10-23

Getting required packages

```
#install.packages("tidyverse")
#install.packages("Benchmarking")
```

Loading the libraries

```
library(Benchmarking)
```

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

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

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

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

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

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
```

```
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.1     v stringr 1.4.1
## v readr   2.1.2     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Compute the Formulation

```r
# Creating the vectors with the values

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 Column names

colnames(output) <- c("staff_hours_daily","supplies_daily")
colnames(input) <- c("reimbursed_patient_daily","privately_paid_patients_daily")

#Values of the input

input
```

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

```r
#Values of the output

output
```

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

Here we are getting the same values as in the performance data table from the six nursing homes owned by Hope Valley Health Care Association

Further, we will perform a Data Envelopment Analysis (DEA), which is an analytical tool which helps organisations to identify and allocate their resources to enhance their efficiency and have better practices.

DEA Analysis using FDH

Let's formulate and compute the DEA analysis using FDH

The Free disposability hull (FDH) is the assumption of dispose unwanted inpputs and outputs. "Free disposability means that we can always produce fewer outputs with more inputs."

```r
#Provide the input and output
FDH_analysis <- dea(input,output,RTS= "fdh")

#Create a data frame with efficiency values
```

```
eff_fdh <- as.data.frame(FDH_analysis$eff)

#To assign an appropriate name
colnames(eff_fdh) <- c("efficiency_fdh")

#Identify the peers
peer_fdh <- peers(FDH_analysis)

#To assign an appropriate name
colnames(peer_fdh) <- c("peer1_fdh")

#Identify the relative weights given to the peers using lamba function
lambda_fdh <- lambda(FDH_analysis)

#To assign an appropriate column name for lambda
colnames(lambda_fdh)<- c("L1_fdh","L2_fdh","L3_fdh","L4_fdh","L5_fdh","L6_fdh")

#Create a tabular data with peer, lambda, efficiency
peer_lamb_eff_fdh <- cbind(peer_fdh,lambda_fdh,eff_fdh)

#Show a summary chart
peer_lamb_eff_fdh
```

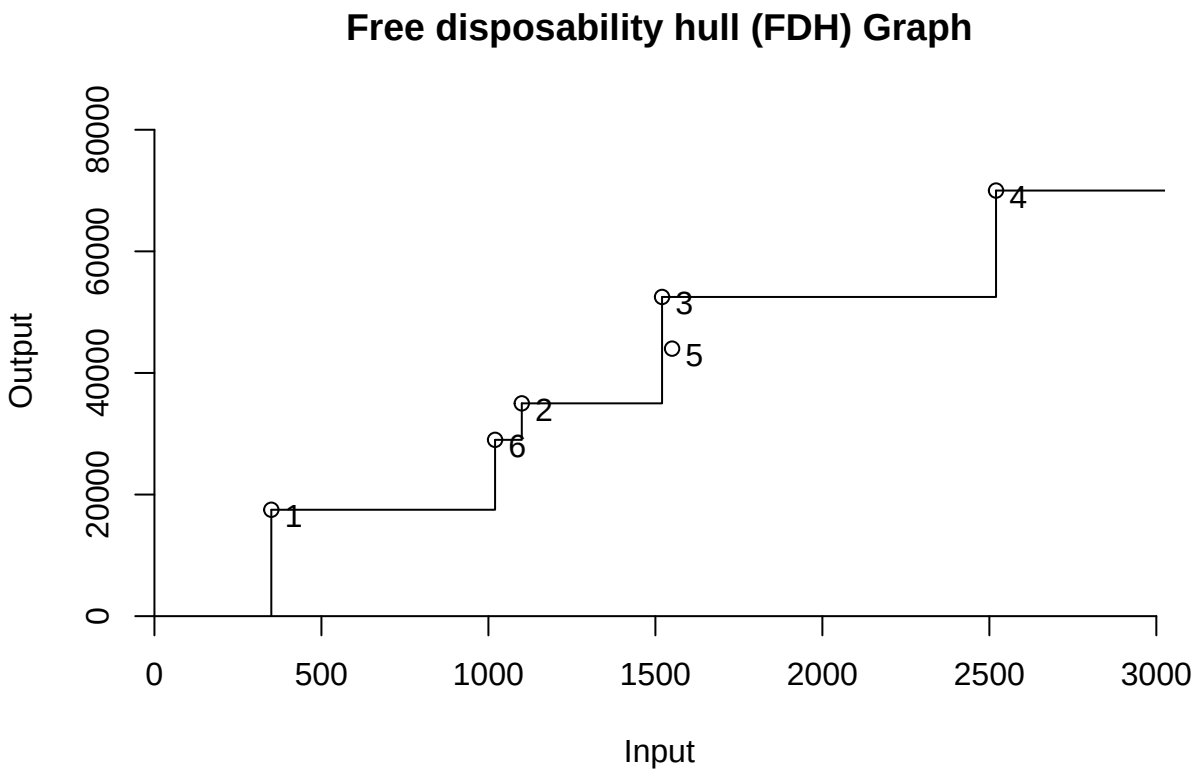
```
##   peer1_fdh L1_fdh L2_fdh L3_fdh L4_fdh L5_fdh L6_fdh efficiency_fdh
## 1         1      1      0      0      0      0      0              1
## 2         2      0      1      0      0      0      0              1
## 3         3      0      0      1      0      0      0              1
## 4    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

As we learned during this module, peers are the way we could identify inefficent DMU or units, and Lambda values are the raw weights assigned from the peer units when solving the DEA model.

The summary chart shown above, confirms that every DMU or facility is working using all its capacity and efficiency. Every peer was assigned one unit, for that reason, the Lambda values are 1, and efficiency are 1 as well.

Now, let's see the graph.

```
#Plot the results
dea.plot(input,output,RTS="fdh", ORIENTATION="in-out", txt= TRUE,xlab= 'Input', ylab='Output',main="Fre
```



DEA Analysis using CRS

Now, we are going to formulate and compute the DEA analysis using Constant Returns to Scale (CRS).

The CRS is part of the scaling assumption, and it allows us to see if there is any possible combination to scale up or down.

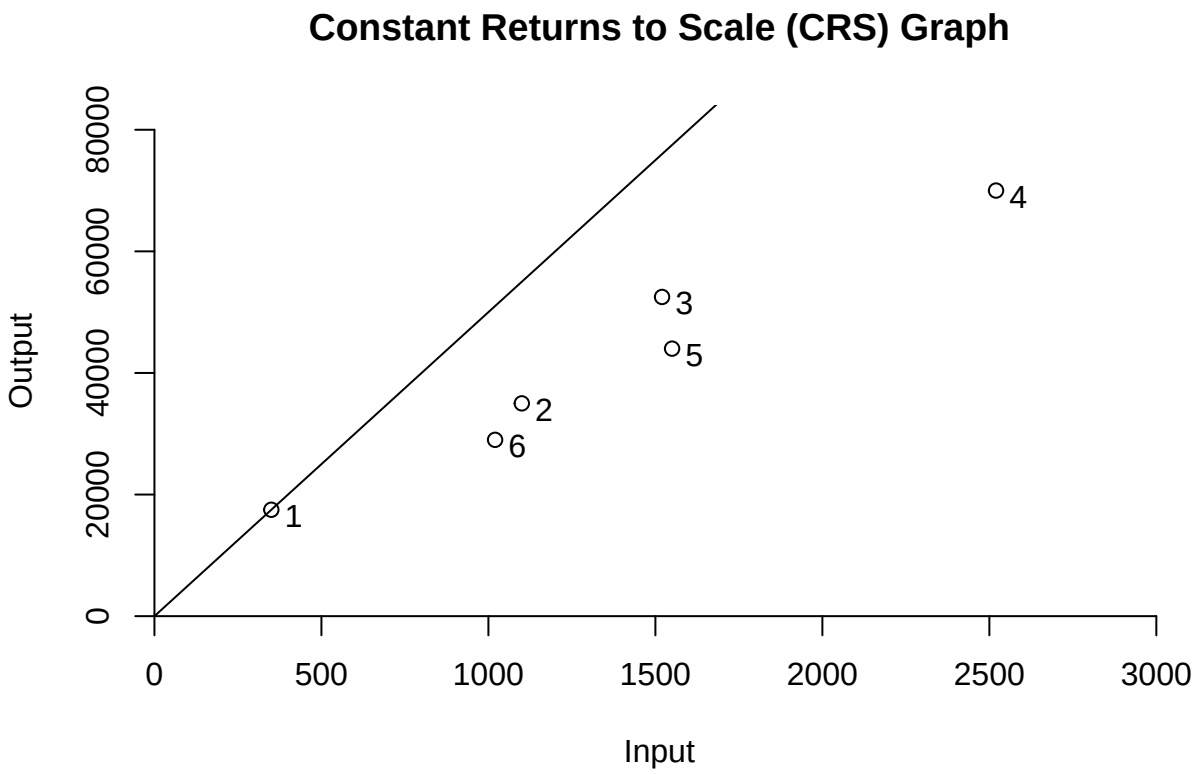
```
#Provide the input and output
CRS_analysis <- dea(input,output,RTS="crs")

#To see the efficiency values
CRS_eff<- as.data.frame(CRS_analysis$eff)

#To assign an appropriate name
colnames(CRS_eff)<- c("CRS_efficiency")

#Identify the peers
CRS_peer<- peers(CRS_analysis)

#To assign an appropriate name
colnames(CRS_peer)<- c("CRS-peer1","CRS-peer2","CRS_peer3")

#Identify the relative weights given to the peers using Lambda function
CRS_lambda<- lambda(CRS_analysis)
```

```r
#To assign an appropriate column name for Lambda
colnames(CRS_lambda)<- c("CRS_L1","CRS_L2","CRS_L3","CRS_L4")

#Create a tabular data with peer, lambda and efficiency
peer_lamb_eff_CRS<- cbind(CRS_peer,CRS_lambda,CRS_eff)

#Show the summary chart
peer_lamb_eff_CRS
```

```
##   CRS-peer1 CRS-peer2 CRS_peer3    CRS_L1     CRS_L2 CRS_L3    CRS_L4
## 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
##   CRS_efficiency
## 1      1.0000000
## 2      1.0000000
## 3      1.0000000
## 4      1.0000000
## 5      0.9774987
## 6      0.8674521
```

Regarding Constant Returns to Scale (CRS), the facilities 1, 2, 3, and 4 are using all its efficiency as the lambdas and peers prove. Facility 5 and 6, on the other hand, need parts of 1, 2, and 4 as the peers and lambdas show above. It means these two facilities (5 and 6) have room to improve because they are getting an efficiency of 97.74% and 86.74% respectively.

```r
#Plot the graph
dea.plot(input,output,RTS="crs",ORIENTATION="in-out",
txt=TRUE, xlab = "Input", ylab= "Output", main="Constant Returns to Scale (CRS) Graph")
```



DEA Analysis using VRS

Now, we are going to formulate and compute the DEA analysis using Variable Returns to Scale (VRS).VRS is also part of the scaling assumption, and it helps to estimate the efficiency of the variables whether an increase or decrease is not proportional.

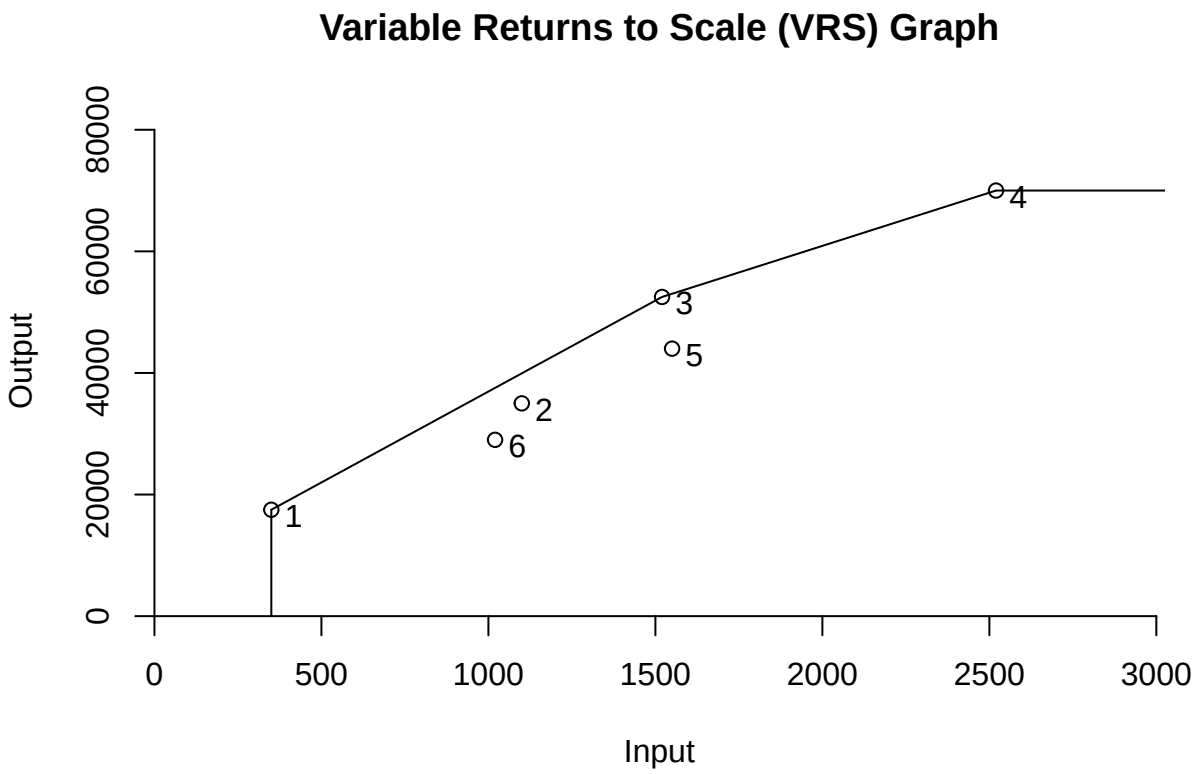
```
# Provide the input and output
VRS_analysis <- dea(input,output,RTS = "vrs")

# To see the efficiency values
VRS_eff <- as.data.frame(VRS_analysis$eff)

# To assign an appropiate name
colnames(VRS_eff) <- c("VRS_efficiency")

# Identify the peers
VRS_peer <- peers(VRS_analysis)

# To assign an appropiate name
colnames(VRS_peer) <- c("VRS_peer1", "VRS_peer2", "VRS_peer3")

# Identify the relative weights given to the peers using lambda function
VRS_lambda <- lambda(VRS_analysis)
```

```r
# To assign an appropiate column name for Lambda
colnames(VRS_lambda) <- c("VRS_L1", "VRS_L2", "VRS_L3", "VRS_L4", "VRS_L5")

# Create a tabular data with peer, lambda, and efficiency
peer_lamb_eff_VRS <- cbind(VRS_peer, VRS_lambda, VRS_eff)

# Show the summary chart
peer_lamb_eff_VRS
```

```
##   VRS_peer1 VRS_peer2 VRS_peer3    VRS_L1    VRS_L2 VRS_L3 VRS_L4    VRS_L5
## 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
##   VRS_efficiency
## 1      1.0000000
## 2      1.0000000
## 3      1.0000000
## 4      1.0000000
## 5      1.0000000
## 6      0.8963283
```

Now we run the Variable Returns to Scale (VRS), we can identify that facility 1, 2, 3, 4, and 5 are working in all its capacity or efficiency. However, that does not happen with facility 6, which has an efficiency of 89.63%. As peers and lambdas show, facility 6 needs part of facility 1, 2, and 5 to achieve better efficiency.

```r
# Plot the graph
dea.plot(input,output,RTS="vrs",ORIENTATION="in-out",
txt=TRUE, xlab = "Input", ylab= "Output", main="Variable Returns to Scale (VRS) Graph")
```



DEA Analysis using IRS

Now, we are going to formulate and compute the DEA analysis using Increasing Returns to Scale (IRS).IRS indicates if it is possible to increate the operation scale.

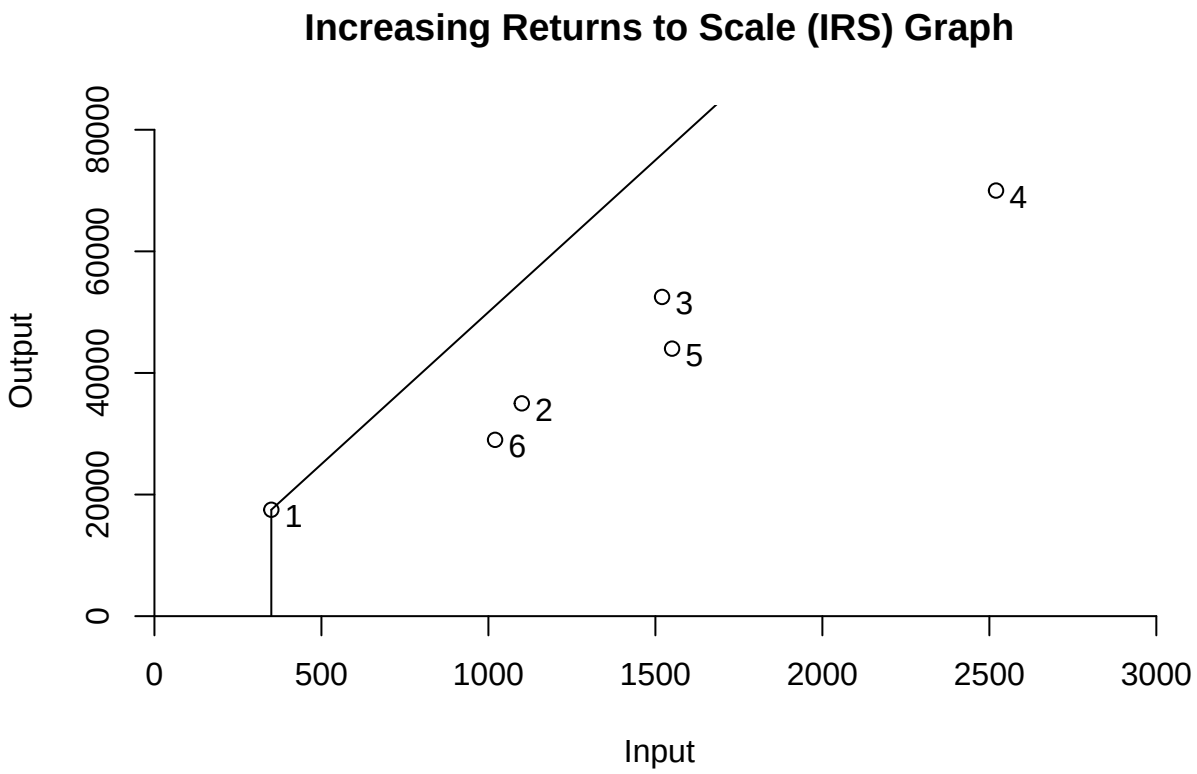
```
# Provide the input and output
IRS_analysis<- dea(input,output,RTS = "irs")

# To see the effciency values
IRS_eff <- as.data.frame(IRS_analysis$eff)

# To assign an appropiate name
colnames(IRS_eff) <- c("IRS_efficiency")

# Identify the peers
IRS_peer <- peers(IRS_analysis)

# To assign an appropiate name
colnames(IRS_peer) <- c("IRS_peer1", "IRS_peer2", "IRS_peer3")

# Identify the relative weights given to the peers using lambda function
IRS_lambda <- lambda(IRS_analysis)

# To assign an appropiate column name for Lambda
```

```r
colnames(IRS_lambda) <- c("IRS_L1", "IRS_L2", "IRS_L3", "IRS_L4", "IRS_L5")

# Create a tabular data with peer, lambda, and efficiency
peer_lamb_eff_IRS <- cbind(IRS_peer, IRS_lambda, IRS_eff)

# Show the summary chart
peer_lamb_eff_IRS
```

```
##   IRS_peer1 IRS_peer2 IRS_peer3    IRS_L1    IRS_L2 IRS_L3 IRS_L4    IRS_L5
## 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
##   IRS_efficiency
## 1      1.0000000
## 2      1.0000000
## 3      1.0000000
## 4      1.0000000
## 5      1.0000000
## 6      0.8963283
```

Increasing Returns to Scale (IRS) behives the same as Variable Returns to Scale (VRS) by getting facility 1, 2, 3, 4, and 5 are working all its efficiency, but facility 6 needs to improve needs from units 1, 2, and 5 to improve its efficiency which is 89.63%.

```r
# Plot the graph
dea.plot(input,output,RTS="irs",ORIENTATION="in-out",
txt=TRUE, xlab = "Input", ylab= "Output", main="Increasing Returns to Scale (IRS) Graph")
```



DEA Analysis using DRS

Now, we are going to formulate and compute the DEA analysis using Decreasing Returns to Scale (DRS).DRS is the opposite of IRS, which its goal is to decrease the operation scale on any possible production process.

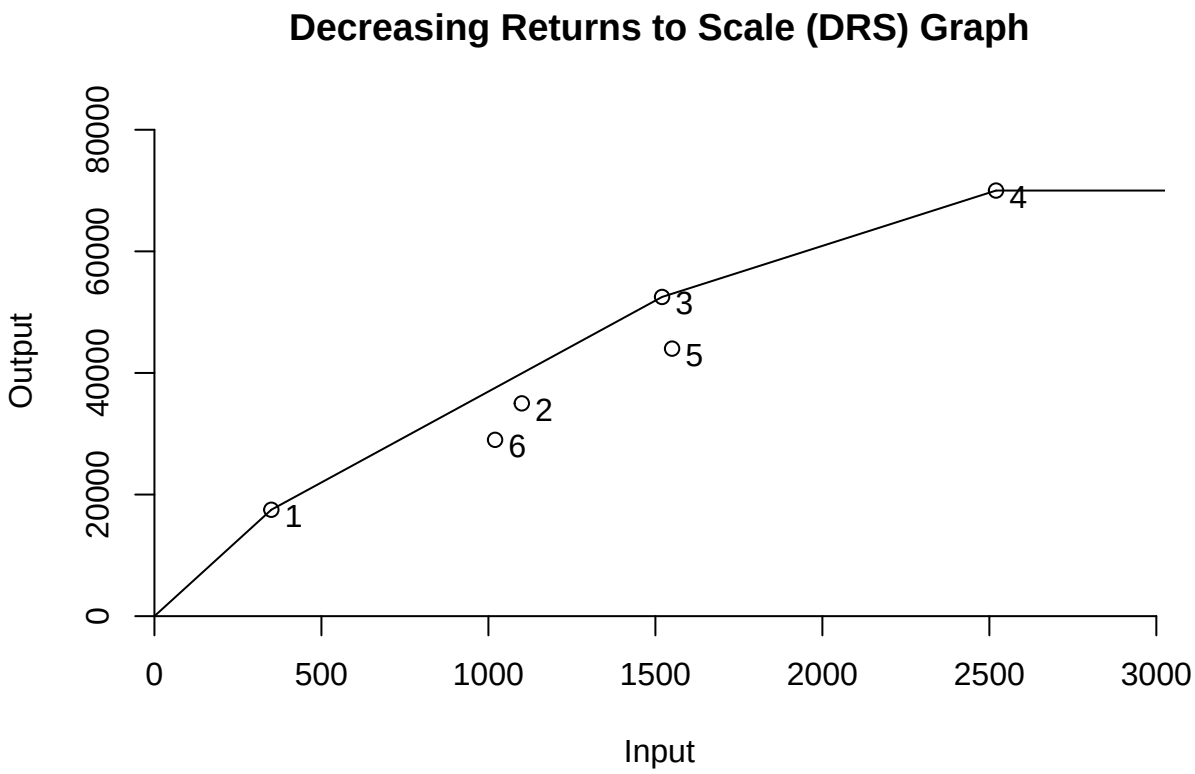
```
# Provide the input and output
DRS_analysis <- dea(input,output,RTS = "drs")

# To see the effciency values
DRS_eff <- as.data.frame(DRS_analysis$eff)

# To assign an appropiate name
colnames(DRS_eff) <- c("DRS_efficiency")

# Identify the peers
DRS_peer <- peers(DRS_analysis)

# To assign an appropiate name
colnames(DRS_peer) <- c("DRS_peer1", "DRS_peer2", "DRS_peer3")

# Identify the relative weights given to the peers using lambda function
DRS_lambda <- lambda(DRS_analysis)

# To assign an appropiate column name for Lambda
```

```r
colnames(DRS_lambda) <- c("DRS_L1", "DRS_L2", "DRS_L3", "DRS_L4")

# Create a tabular data with peer, lambda, and efficiency
peer_lamb_eff_DRS <- cbind(DRS_peer, DRS_lambda, DRS_eff)

# Show the summary chart
peer_lamb_eff_DRS
```

```
##   DRS_peer1 DRS_peer2 DRS_peer3    DRS_L1     DRS_L2 DRS_L3    DRS_L4
## 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
##   DRS_efficiency
## 1      1.0000000
## 2      1.0000000
## 3      1.0000000
## 4      1.0000000
## 5      0.9774987
## 6      0.8674521
```

Decreasing Returns to Scale (DRS) has a good efficiency in facility 1, 2, 3, and 4. Regarding facility 5 and 6, there is room they can improve. Both of them need part of facilities 1, 2, and 4 to be able to achieve their highest efficiency of 1 as we can prove in the previous table.

```r
# Plot the graph
dea.plot(input,output,RTS="drs",ORIENTATION="in-out",
txt=TRUE, xlab = "Input", ylab= "Output", main="Decreasing Returns to Scale (DRS) Graph")
```



DEA Analysis using FRH

Now, we are going to formulate and compute the DEA analysis using Free Replicability Hull (FRH).

FRH as well as FDH use mixed integer programming, which refers that the variables must be integers to find the optimal solution. The goal of FRH is to replace deterministic data using random variables.

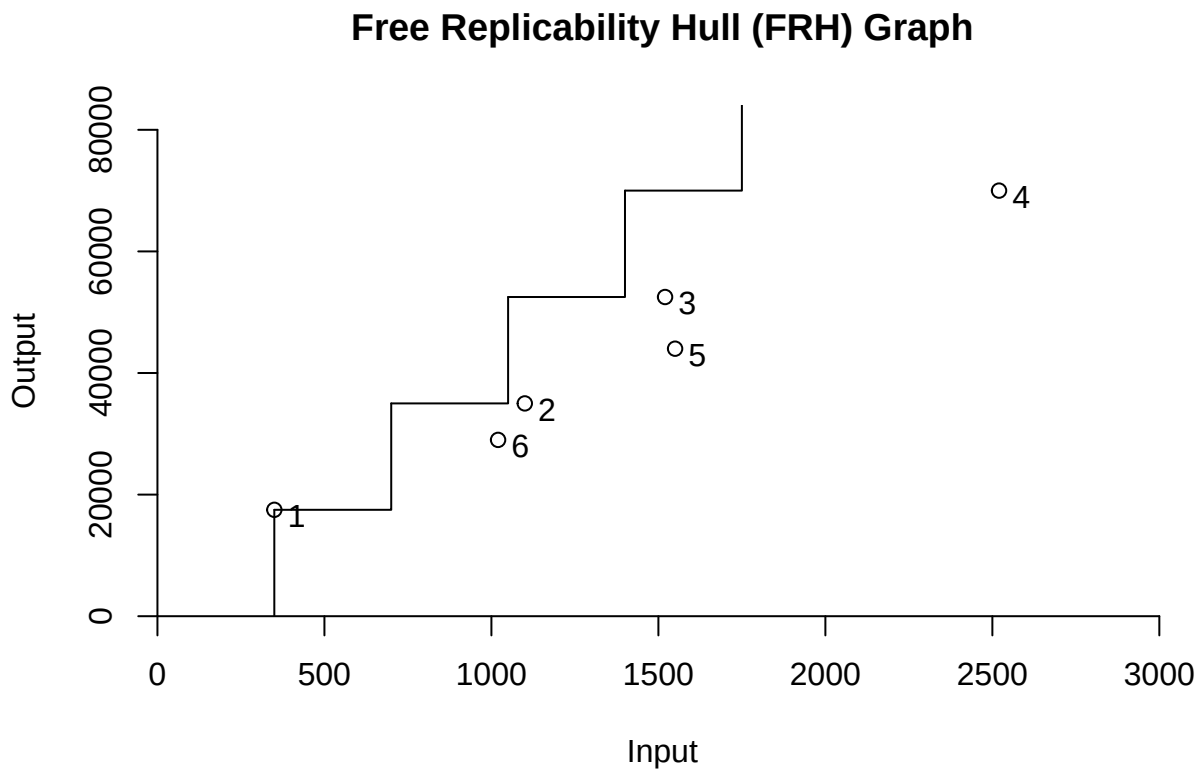
```
# Provide the input and output
FRH_analysis <- dea(input,output,RTS = "add")

# To see the effciency values
FRH_eff <- as.data.frame(FRH_analysis$eff)

# To assign an appropiate name
colnames(FRH_eff) <- c("FRH_efficiency")

# Identify the peers
FRH_peer <- peers(FRH_analysis)

# To assign an appropiate name
colnames(FRH_peer) <- c("FRH_peer1")

# Identify the relative weights given to the peers using lambda function
FRH_lambda <- lambda(FRH_analysis)
```

```r
# To assign an appropiate column name for Lambda
colnames(FRH_lambda) <- c("FRH_L1", "FRH_L2", "FRH_L3", "FRH_L4", "FRH_L5", "FRH_L6")

# Create a tabular data with peer, lambda, and efficiency
peer_lamb_eff_FRH <- cbind(FRH_peer, FRH_lambda, FRH_eff)

# Show the summary chart
peer_lamb_eff_FRH
```

```
##   FRH_peer1 FRH_L1 FRH_L2 FRH_L3 FRH_L4 FRH_L5 FRH_L6 FRH_efficiency
## 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
```

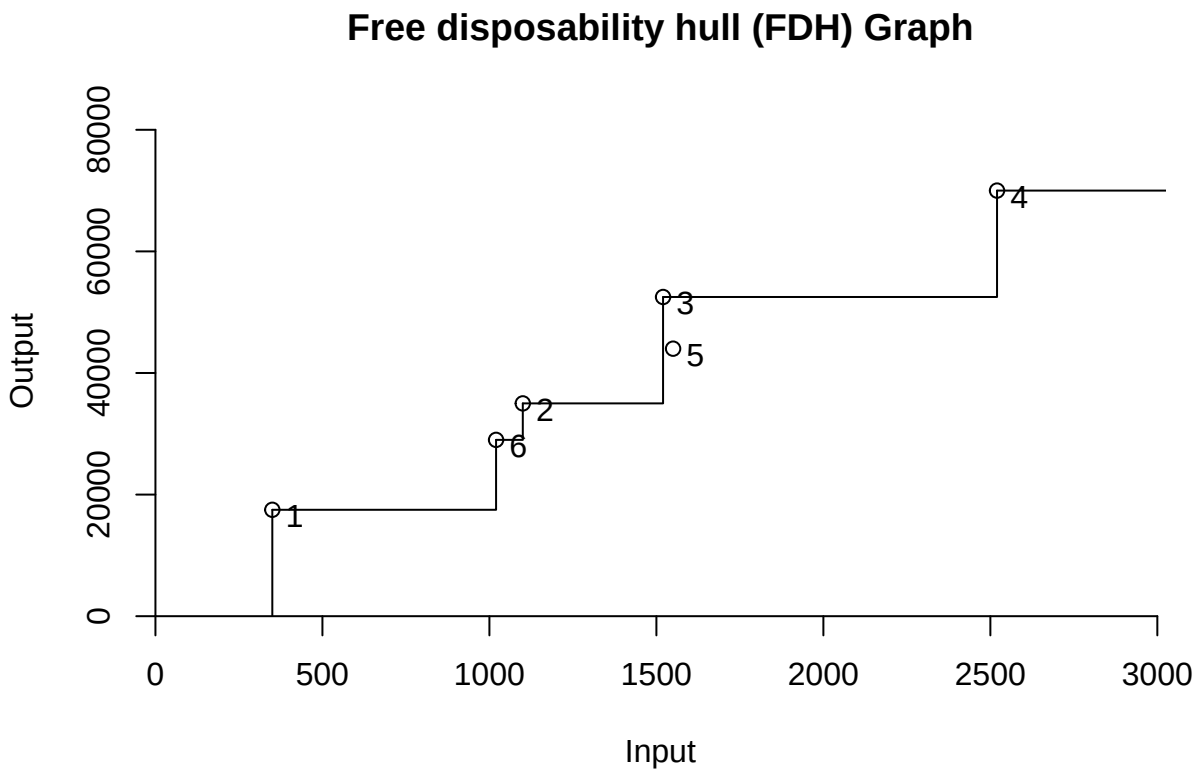
Free Replicability Hull (FRH) has a great efficiency in all its DMU. It behives the same as Free disposability hull (FDH), which all its values have their own peer, lambas and efficiency of 1.

```r
# Plot the graph
dea.plot(input,output,RTS="add",ORIENTATION="in-out",
txt=TRUE, xlab = "Input", ylab= "Output", main="Free Replicability Hull (FRH) Graph")
```

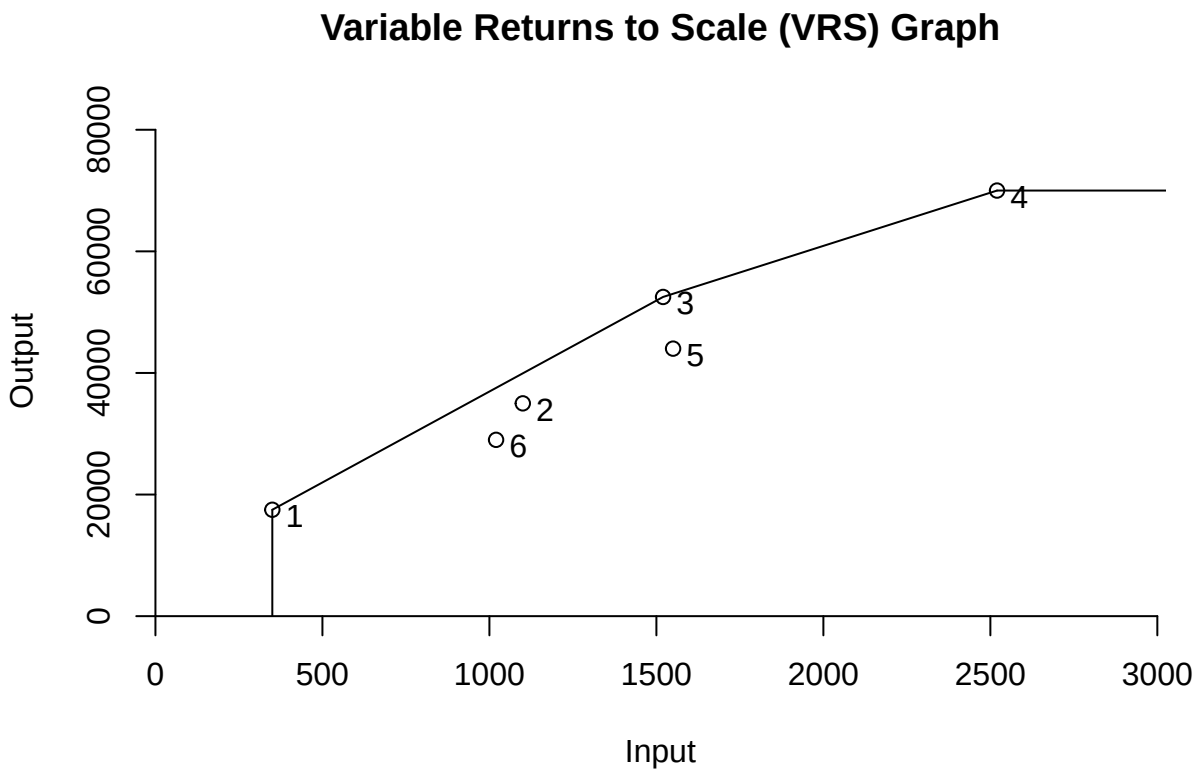


Comparasion between different assumptions

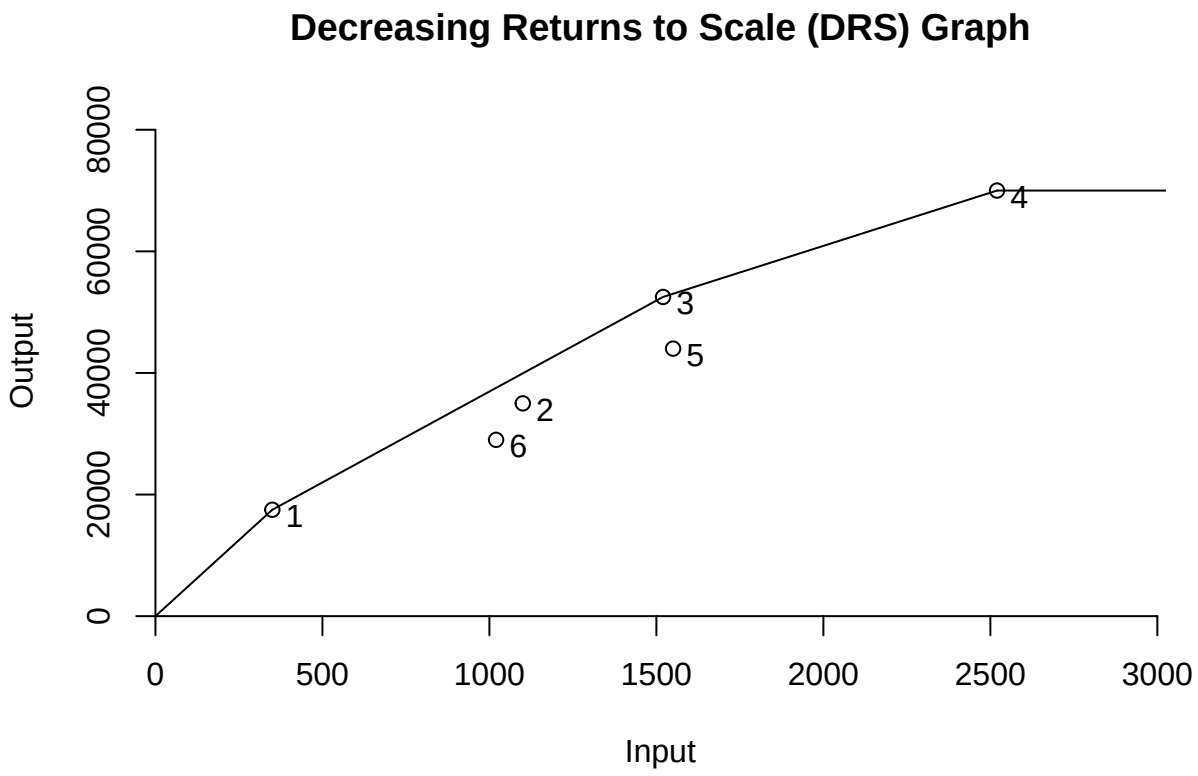
```
dea.plot(input,output,RTS="fdh",ORIENTATION="in-out",
txt=TRUE, xlab = "Input", ylab= "Output", main="Free disposability hull (FDH) Graph")
```



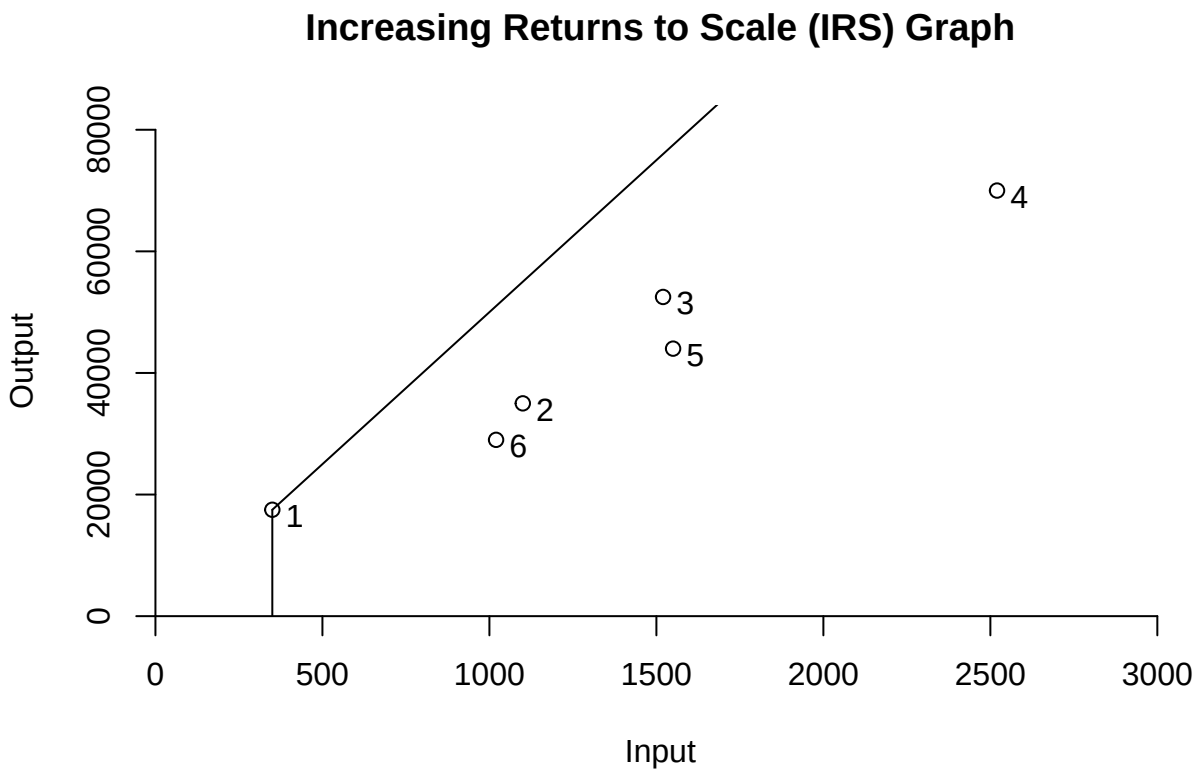

```
dea.plot(input,output,RTS="vrs",ORIENTATION="in-out",
txt=TRUE, xlab = "Input", ylab= "Output", main="Variable Returns to Scale (VRS) Graph")
```



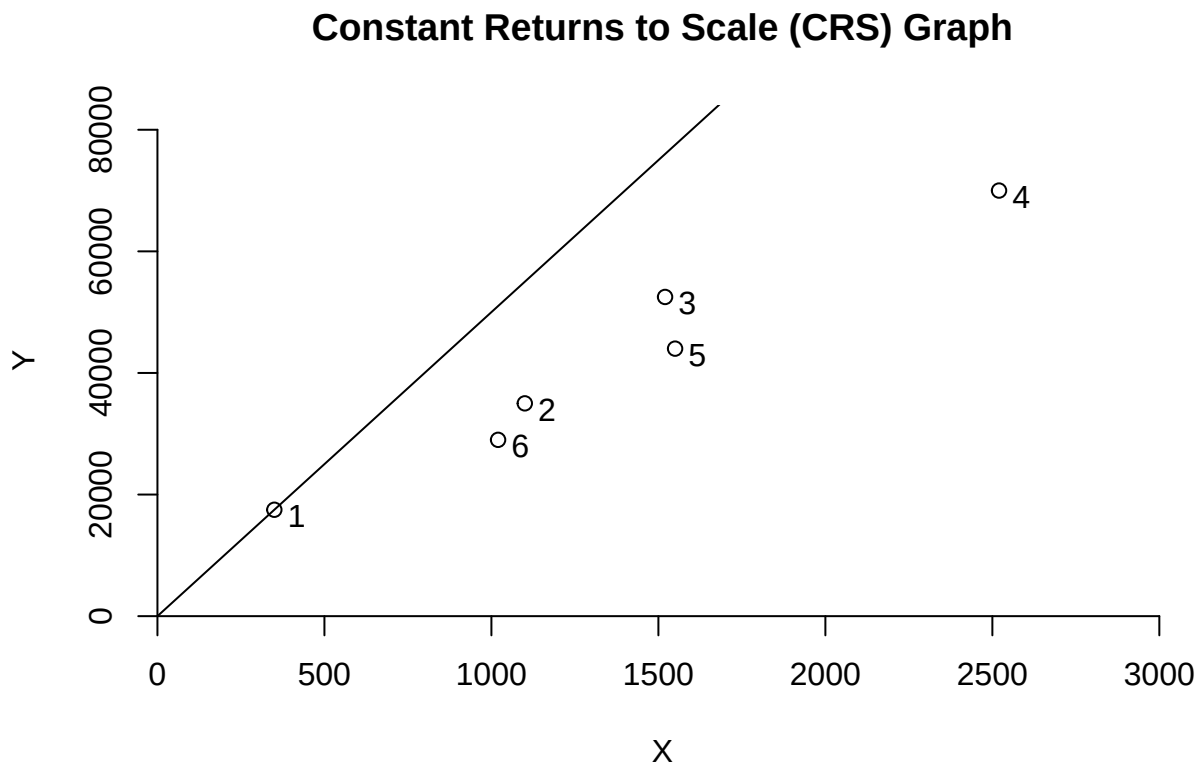

```
dea.plot(input,output,RTS="drs",ORIENTATION="in-out",
txt=TRUE, xlab = "Input", ylab= "Output", main="Decreasing Returns to Scale (DRS) Graph")
```



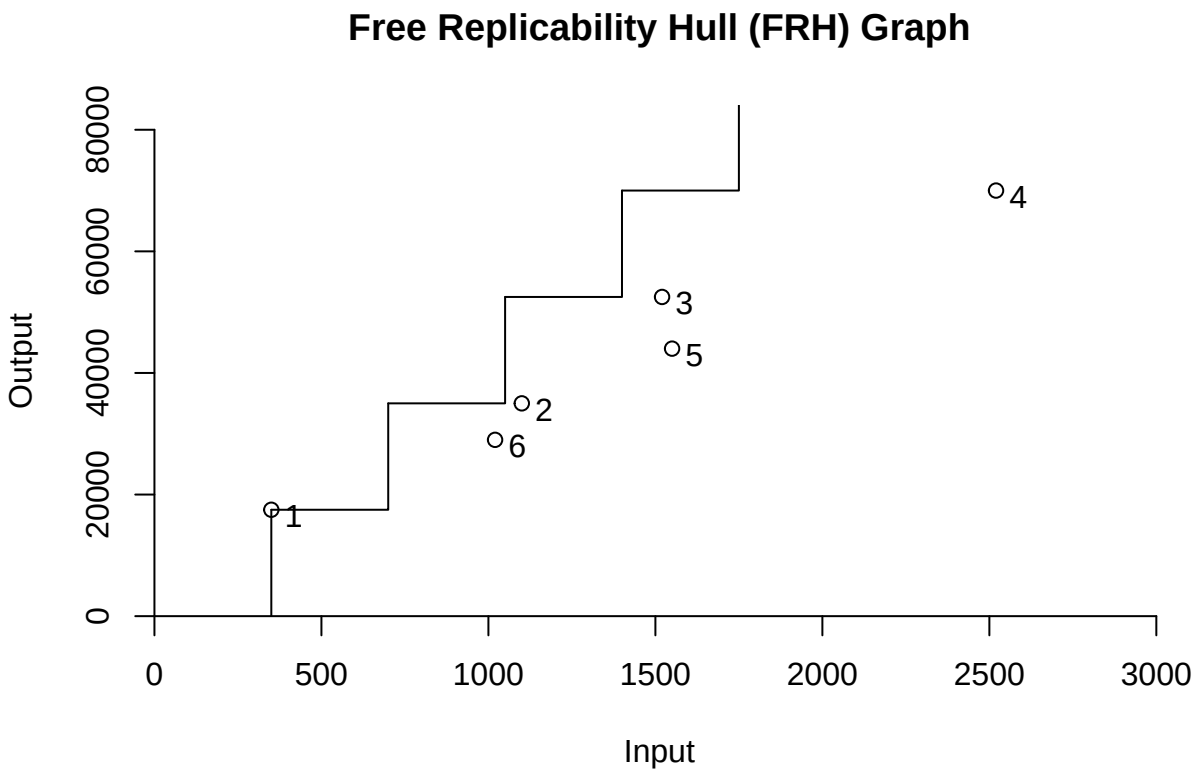

```
dea.plot(input,output,RTS="irs",ORIENTATION="in-out",
txt=TRUE, xlab = "Input",ylab= "Output",main="Increasing Returns to Scale (IRS) Graph")
```




```
dea.plot(input,output,RTS="crs",ORIENTATION="in-out",
txt=TRUE, main="Constant Returns to Scale (CRS) Graph")
```




```
dea.plot(input,output,RTS="add",ORIENTATION="in-out",
txt=TRUE, xlab = "Input", ylab= "Output", main="Free Replicability Hull (FRH) Graph")
```



These charts allow us to compare the results of each DEA model.

As we learned in this module, "all DEA models share the idea of estimating the technology using a minimal extrapolation approach" (DEA Slides).

As we can see FDH is the smallest technology set. It tries to produce fewer outputs (number of patientdays reimbursed by third-party sources and the number of patient-days reimbursed privately) with more inputs (staffing labor and the cost of supplies). FDH is usually the most wanted model by firms, however,it has some drawbacks due to its assumptions. As we can prove, all the efficiencies in this model are 1, but compared to other models it is not as efficient we think because we find 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.

DRS and IRS are larger than VRS as we can see in the charts. DRS tries to increase the set for less input value, while the IRS tries to increase the technology for large input values. DRS indicates that unit 5 and 6 could enhance their efficiency, and IRS shows that facility 6 may improve as well.

CRS is the largest technology set, which allows us to see if there is any possible combination to scale up or down. Based on the efficiency values, units 5 and 6 need to improve.

Regarding FRH, based on the arrow network discussed in class, it is larger than FDH but smaller than CRS,and its goal is to replace deterministic data using random variables. (Bogetoft, P. & Otto, L., 2011, p. 89)

Sources:

Majid, A. Reza, F. (2012). Developing a chance-constrained free replicability hull model for supplier selection. International Journal of Logistics Systems and Management (IJLSM), Vol. 12, No. 4. Retrieved from

http://www.inderscience.com/offer.php?id=48365

Bogetoft, P. & Otto, L. (2011). Benchmarking with DEA, SFA, and R. Retrieved from https://books.google.com/books?id=rBiGxrgFk-kC&pg=PA89&lpg=PA89&dq=fdh+is+smallest+and+crs+is+largest&source=bl&ots

Package Benchmarking. (2020). Retrieved from https://cran.r-project.org/web/packages/Benchmarking/Benchmarking.pdf

Frontier Analyst® FAQs. Retrieved from https://banxia.com/frontier/resources/frequent-questions/