DEA Assignment

2022-10-30

```
knitr::opts_chunk$set(message = FALSE)
knitr::opts_chunk$set(warning = FALSE)
```

```
library("Benchmarking")
x <- matrix(c("Facility 1","Facility 2","Facility 3","Facility 4","Facility 5", "Facility 6",
              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(x) <- c("DMU", "Staff_Hours_Per_Day","Supplies_Per_Day","Reimbursed_Patient_Days","Privately_Pa

table.df <- as.table(x)
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")

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

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

```
peers(DEA_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(DEA_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
```

#CRS Observations *1. We get to see that Facility 1, Facility 2, Facility 3 and Facility 4 are efficient. 2. Also, we get to see that Facility 1, Facility 2 and Facility 4 are the peer members for Facility 5 and Facility 6 which are the inefficient facilities. 3. Facility 5 is 97.75 % efficient leaving 2.25 % as inefficient and Facility 6 is 86.75 % efficient leaving 13.25 % as inefficient.*

#Calculating Decreasing Returns to Scale (DRS)

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

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

```
peers(DEA_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(DEA_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
```

#DRS Observations -
1. We get to see that Facility 1, Facility 2, Facility 3 and Facility 4 are efficient.

2. Also, we get to see that Facility 1, Facility 2 and Facility 4 are the peer members for Facility 5 and Facility 6 which are the inefficient facilities.

3. Facility 5 is 97.75 % efficient leaving 2.25 % as inefficient and Facility 6 is 86.75 % efficient leaving 13.25 % as inefficient.

#Calculating Increasing Returns to Scale (IRS)

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

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

```
peers(DEA_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(DEA_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
```

#IRS Observations - *1. We get to see that Facility 1, Facility 2, Facility 3, Facility 4 and Facility 5 are efficient. 2. Also, we get to see that Facility 1, Facility 2 and Facility 5 are the peer members for Facility 6 which is the only inefficient facility. 3. Facility 6 is 89.63 % efficient leaving 10.37 % as inefficient.*

#Calculating Variable Returns to Scale (VRS)

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

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

```
peers(DEA_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(DEA_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
```

VRS Observations -
1. We get to see that Facility 1, Facility 2, Facility 3, Facility 4 and Facility 5 are efficient.
2. Also, we get to see that Facility 1, Facility 2 and Facility 5 are the peer members for Facility 6 which is the only inefficient facility.
3. Facility 6 is 89.63 % efficient leaving 10.37 % as inefficient.

#Calculating Free Disposability Hull (FDH)

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

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

```
peers(DEA_FDH)
```

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

```
lambda(DEA_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
```

FDH Observations -

All the DMUs are efficient. This is basically due to the principal which FDH technique follows thereby detecting even a small level of efficiency.

#Calculating Free Replicability Hull (FRH)

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

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

```r
peers(DEA_FRH)
```

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

```r
lambda(DEA_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
```

#FRH Observations - *All the DMUs are efficient. It follows the no convexity assumption which ensures that the output is free from disposal and replication.*

#Summary of Results (Inefficient DMUs)

```r
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", "Al

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
```

#Interpretation of the DEA Analysis - *Before the interpretation it's important to know the differences in the scales (RTS),*

Constant Returns to Scale (CRS) is considered as the original scale which is widely used by most of the firms.

The Decreasing, Increasing and Varying Returns to Scale (DRS, IRS and VRS) are the dispersion scales which helps us in understanding what to decrease and what to increase based on the input deployment.

The Free Disposability and Free Replicability Hull (FDH & FRH) are considered as the non-parametric method to measure the efficiency of the DMUs i.e. no convexity assumption.

#CRS - Constant Returns to Scale***

The results indicate that DMUs 1, 2, 3 and 4 are efficient. DMU(5) is only 97.75% efficient, and DMU(6) is 86.7% efficient. This is what we found basis our initial analysis. Further, the peer units for DMU(4) are 1, 2 and 4 with relative weights of 0.2, 0.08 and 0.54. Similarly for DMU(6), the peer units are 1, 2 and 4 with weights of 0.34, 0.4 and 0.13 respectively.

Basically, CRS helps us to know if any possible DMUs can be scaled up or down, in this case DMUs 1, 2, 3 and 4 can be scaled up.

#DRS - Decreasing Returns to Scale***

The results indicate that DMUs 1, 2, 3 and 4 are efficient. DMU(5) is only 97.75% efficient and DMU(6) is 86.7% efficient. This is what we found basis our initial analysis. Further, the peer units for DMU(4) are 1, 2 and 4 with relative weights of 0.2, 0.08 and 0.54. Similarly for DMU(6), the peer units are 1, 2 and 4 with weights of 0.34, 0.4 and 0.13 respectively.

This scale tell us if there are any possible DMUs where we can scale the operations i.e. by looking at the inefficient DMUs in this case it's DMU 5 & 6 can be scaled down. This thing can also be fetched by looking at the CRS values since it's the base original scale.

#IRS - Increasing Returns to Scale***

The results indicate that DMUs 1, 2, 3, 4 and 5 are efficient. DMU(6) is only 89.63% efficient. This is what we found basis our initial analysis. Further, the peer units for DMU(6) are 1, 2 and 5 with relative weights of 0.4, 0.34 and 0.26 respectively.

As the name suggests it let's any firm know if they can arbitrarily increase the scale of operation by looking at the efficiency scores.(Refer data.df.summarise.efficient table)

#VRS - Variable Returns to Scale***

The results indicate that DMUs 1, 2, 3, 4 and 5 are efficient. DMU(6) is only 89.63% efficient. This is what we found basis our initial analysis. Further, the peer units for DMU(6) are 1, 2 and 5 with relative weights of 0.4, 0.34 and 0.26 respectively.
Varying or Variable Returns to Scale helps us understand the scale of operations with variations towards the input and output factor either by increasing or decreasing or by employing both.

#FDH - Free Disposability Hull***

The results indicate that all the DMUs are efficient. This is basically due to no convexity assumption and mostly this technique allows the scale to capture even the smallest level of efficiency.

#FRH - Free Replicability Hull***

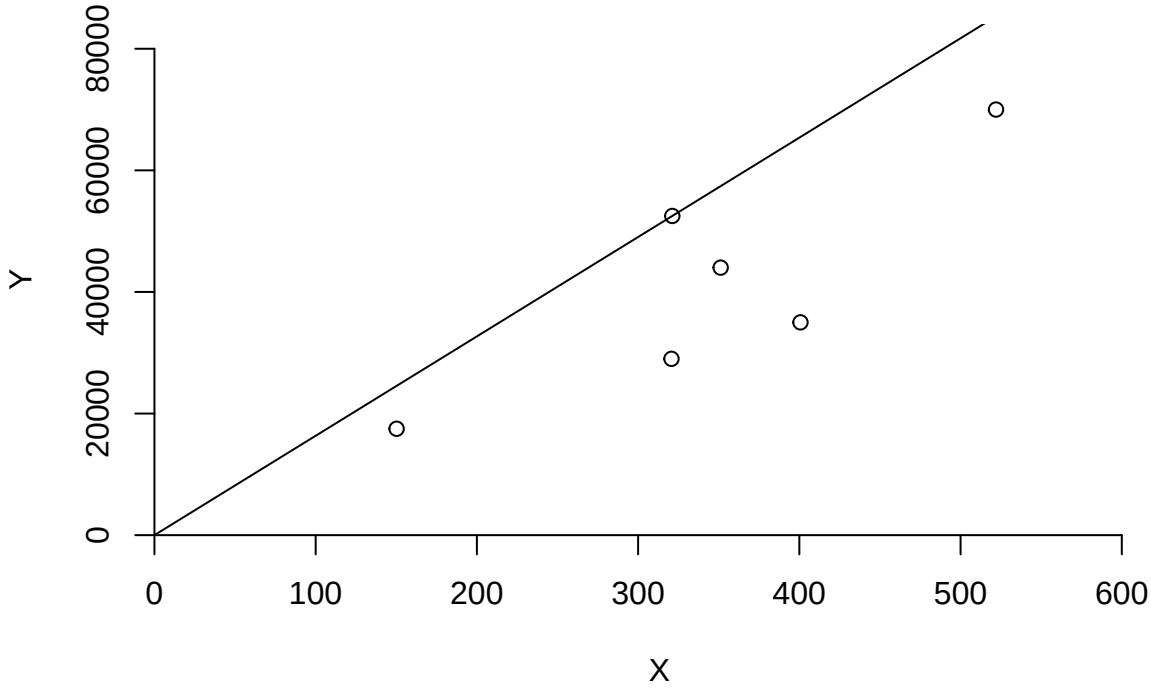
The results from the FRH indicate that all the DMUs are efficient. This is basically due to no convexity assumption and mostly this technique allows the scale to capture even the smallest level of efficiency which is free from replication and disposal.

Note - The peer values i.e. neighbors and lambda values i.e. weights of the peers would be only retrieved to the DMUs which are inefficient. Efficient DMUs don't have peers and lambda weights.

Conclusion
It is must to note that DEA is a very useful tool to any firm in deciding which is the best DMU i.e. which of the Decision Making Unit has to be maximized so that there would be an increase, decrease or any kind of variations to the output by feeding input into it.
Also, a company can decide upon which of the RTS it wants to employ i.e. Returns to Scale based on their requirements, each of these scales has it's own importance.

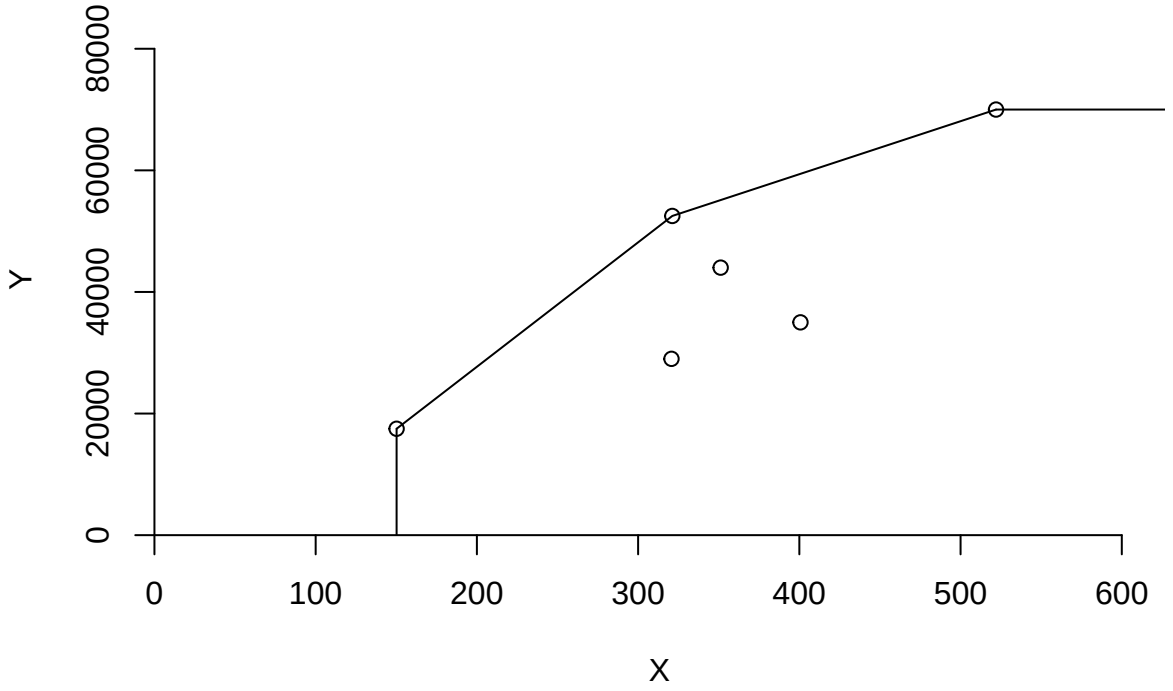
#Plotting the Graphs
 # CRS Plot

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



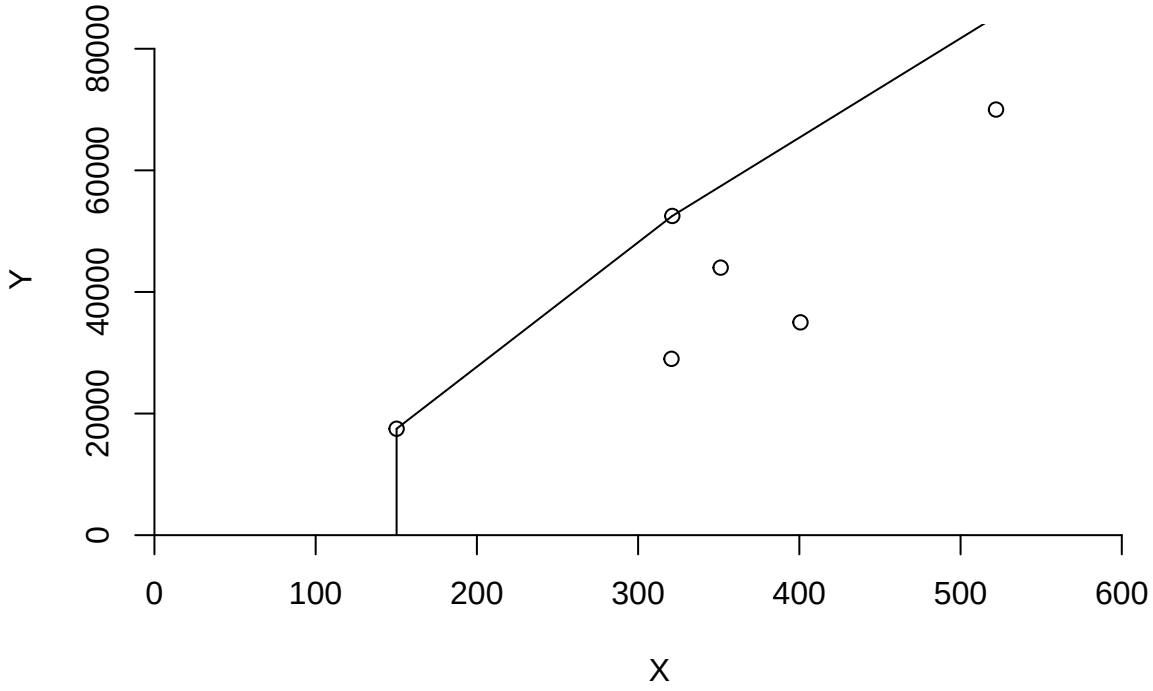
DRS Plot

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



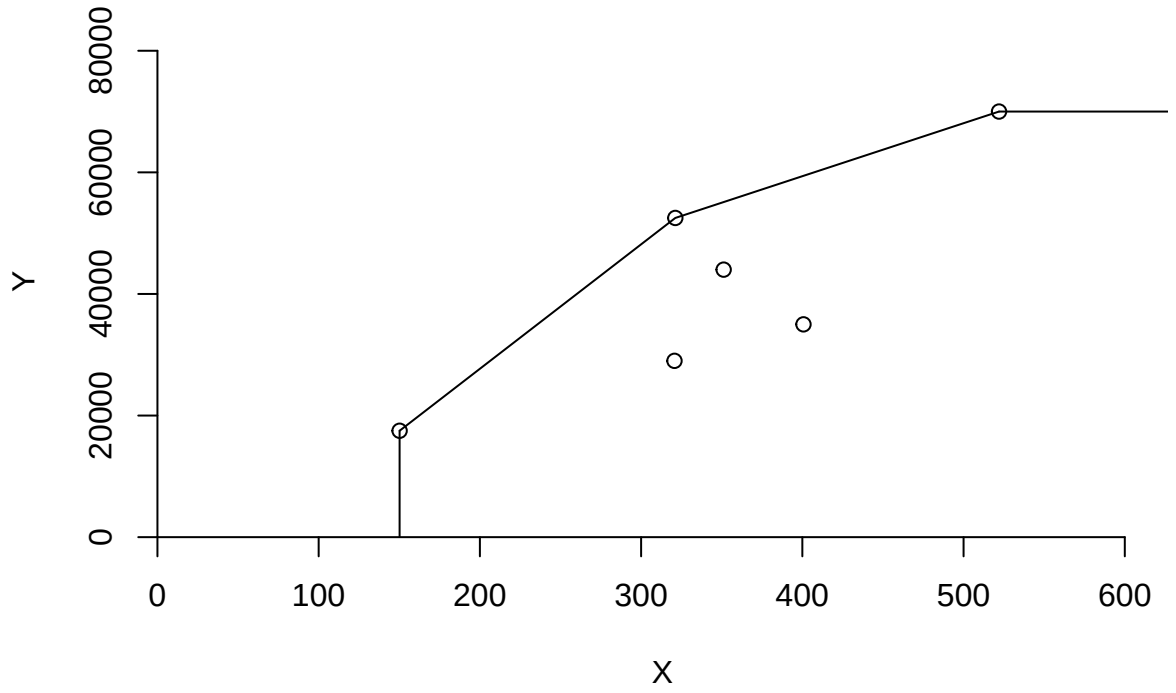
IRS Plot

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



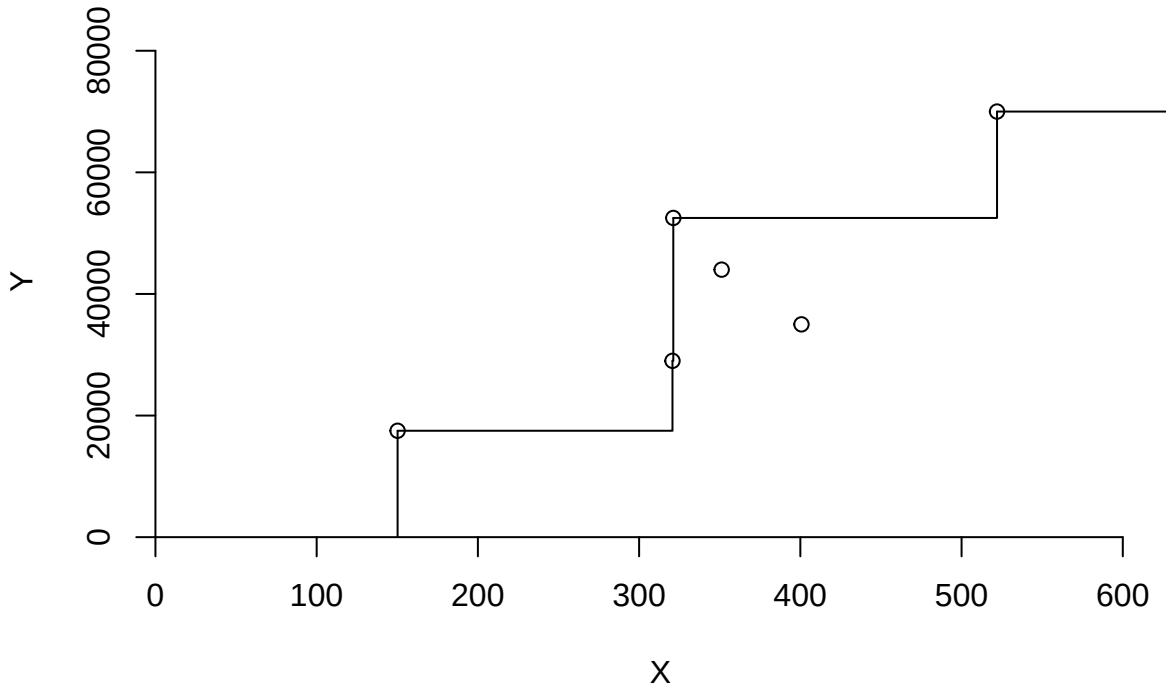
VRS Plot

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



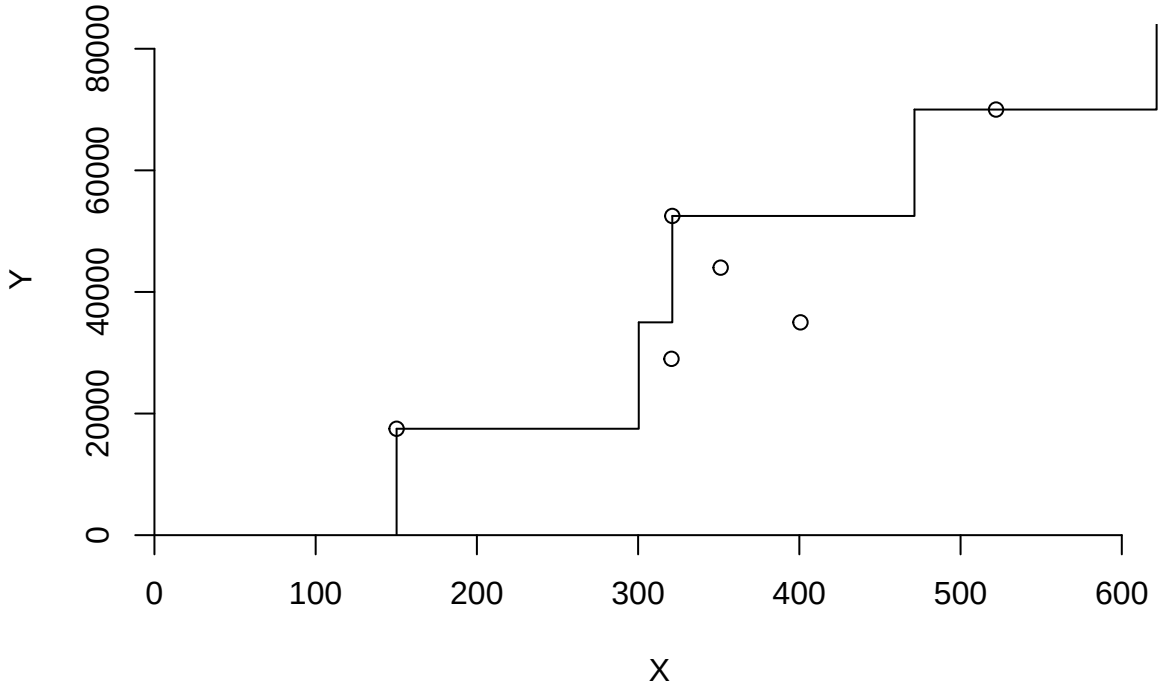
FDH Plot

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



FRH Plot

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


Y
0
20000
40000
60000
80000
0
100
200
300
400
500
600
X