Assignment-4

HANISH BHOGADI

2022-10-30

```
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 that 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")

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

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

```
peers(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(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:-*

a. We see that Facility 1, Facility 2, Facility 3 and Facility 4 are efficient.

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

c. Facility 5 is 97.75 % efficient leaving 2.25 % as inefficient

d.And Facility 6 is 86.75 % efficient leaving 13.25 % as inefficient.

**Calculating the Decreasing that returns to Scale (DRS)*

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

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

```
peers(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(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 can see that Facility 1, Facility 2, Facility 3 and Facility 4 are efficient.*
2. Also, we see that Facility 1, Facility 2 and Facility 4 are the peer members for Facility 5 and Facility 6 which are inefficient facilities.*
3. Facility 5 is 97.75 % efficient leaving 2.25 % as inefficient and Facility 6 is 86.75 % efficient i.e., leaving 13.25 % as inefficient.*

Calculating Increasing Returns to Scale (IRS)

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

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

```
peers(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(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 inefficient facility.*
3. Facility 6 is 89.63 % efficient leaving 10.37 % as inefficient.*

Calculating Variable Returns to Scale (VRS)

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

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

```
peers(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(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 the Free Disposability Hull (FDH)

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

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

```
peers(FDH)
```

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

```
lambda(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 due to the principal which FDH technique follows there by detecting even a small level of efficiency.

Calculating Free Replicability Hull (FRH)

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

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

```
peers(FRH)
```

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

```
lambda(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 it ensures that the o/p is free from disposal and replication.

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", "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

1.Before interpretation knowing the variations in the scales (RTS) is crucial.**

2.Constant Returns to Scale (CRS) is regarded as the original scale and is widely used the majority of businesses.**

3.The Free Disposability and Free Replicability Hull (FDH & FRH) are regarded as a non-parametric way to evaluate the effectiveness of DMUs i.e no convexity assumption.*

4.The dispersion scales known as Decreasing, Increasing and Varying Returns to Scale (DRS, IRS, and VRS) help us in determining what to increase and decrease based on the input deployment.*

DRS - Decreasing Returns to Scale

1.The results show that DMUs 1, 2, 3, and 4 are effective. DMU(5) has a 97.75% efficiency and DMU(6) has an 86.7% efficiency.

2.This is what we discovered based on our preliminary research. Furthermore, DMU(4peer)'s units are 1, 2, and 4, with relative weights of 0.2, 0.08, and 0.54.

3.Similarly, the peer units for DMU(6) are 1, 2, and 4, with weights of 0.34, 0.4, and 0.13, respectively.

4.This scale tells us if there are any possible DMUs where we can scale the operations, for example, by looking at the inefficient DMUs in this case, DMUs 5 and 6. This is the base original scale, it can also be obtained by looking at the CRS values.

CRS - Constant Returns to Scale

1.The results show that DMUs 1, 2, 3, and 4 are productive. Only 97.75% of DMU(5) and 86.7% of DMU(6) are effectively used. On the basis of our initial analysis, we discovered this. In addition, DMU(4peer)'s units are 1, 2, and 4, with respective weights of 0.2, 0.08, and 0.54. The peer units for DMU(6) are 1, 2, and 4, with respective weights of 0.34, 0.4, and 0.13.*

3. Basically, CRS helps us to determine whether any potential DMUs may be scaled up or down, in this case, DMUs 1, 2, 3, and 4.

IRS - Increasing Returns to Scale

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

2.As name suggeets, this scale tells us if there are any firm DMUs where we can scale the operations, for example, by looking at the inefficient DMUs in this case, DMU 5 and 6. Because this is the base original scale, it can also be retrieved by looking at the CRS values.

VRS - Variable Returns to Scale

1.The results show that DMUs 1, 2, 3, 4, and 5 are effective. DMU(6) has an efficiency of only 89.63%. This is what we found based on our initial research.

2.Moreover, DMU(6) has peer units 1, 2, and 5 with relative weights of 0.4, 0.34, and 0.26, respectively.

3.Varying or Variable Returns to Scale helps us understand the scale of operations with variations in the input and output factor, either increasing or decreasing or utilizing both.*

FRH - Free Replicability Hull

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

Note - The peer values, i.e. neighbors, and lambda values, i.e. weights of the peers, would only be retrieved to the inefficient DMUs. Efficient DMUs lack peers and lambda weights.

FDH - Free Disposability Hull

*The results show that all of the DMUs are efficient. This is primarily due to the assumption of no convexity, and this technique enables the scale to capture even the smallest level of efficiency.

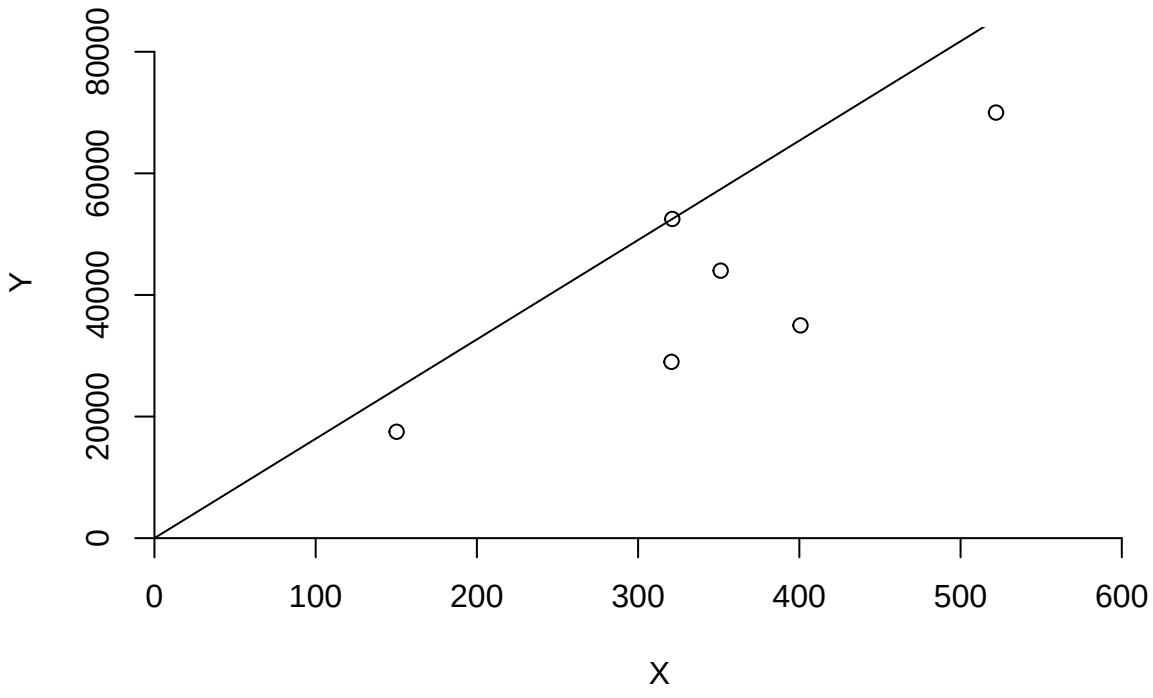
Conclusion

1. It is must to note that DEA is a very useful tool for any firm in deciding which DMU is the best, i.e. which of the Decision Making Units should be maximized so that there is an increase, decrease, or any kind of variation in the output by feeding input into it.
2. Also, a company can decide which RTS it wants to employ i.e. Returns to Scale, to use based on their needs; each of these scales has its own importance.*

Plotting the Graphs

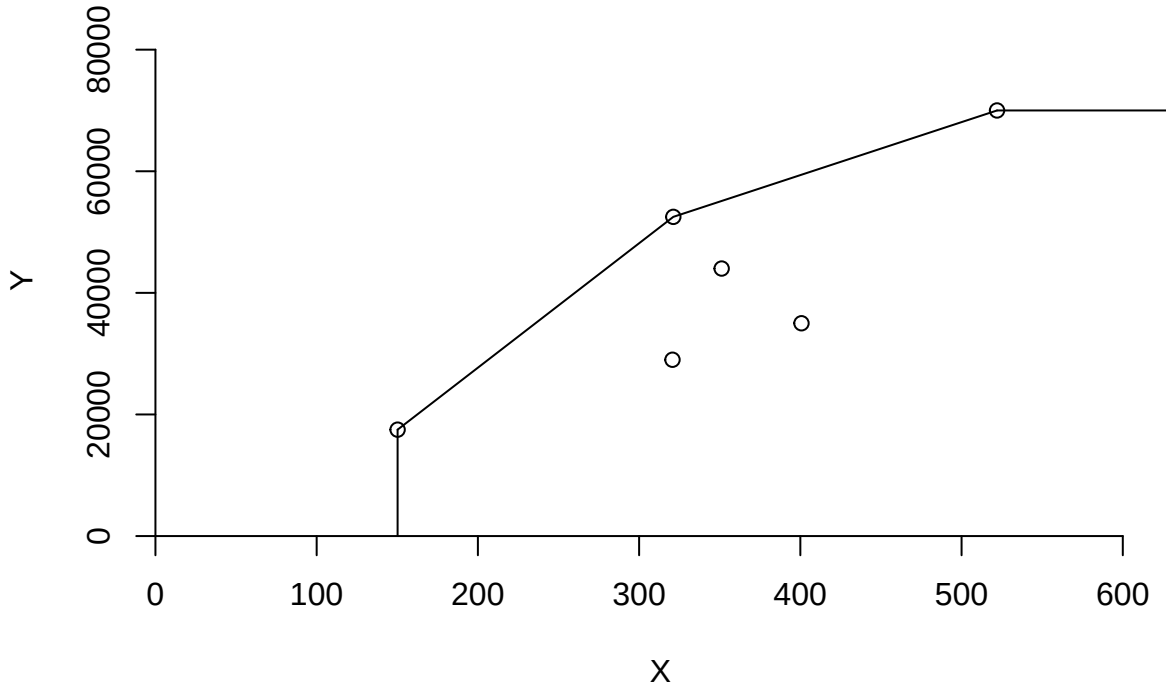
CRS Plot

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



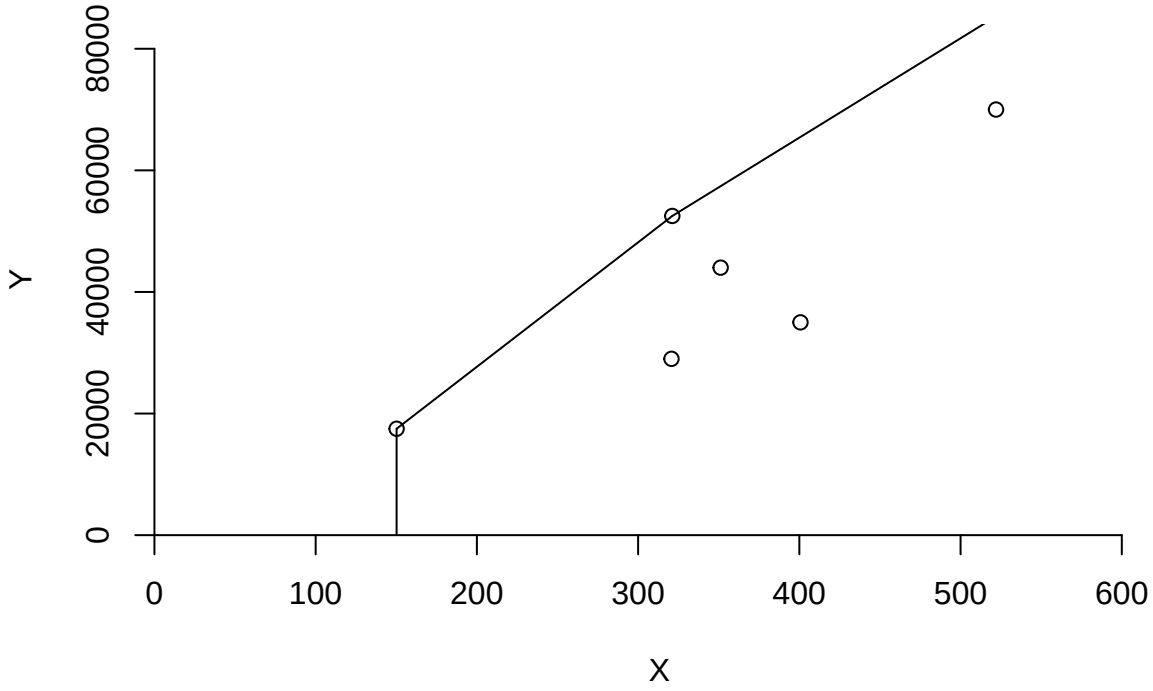
DRS Plot

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



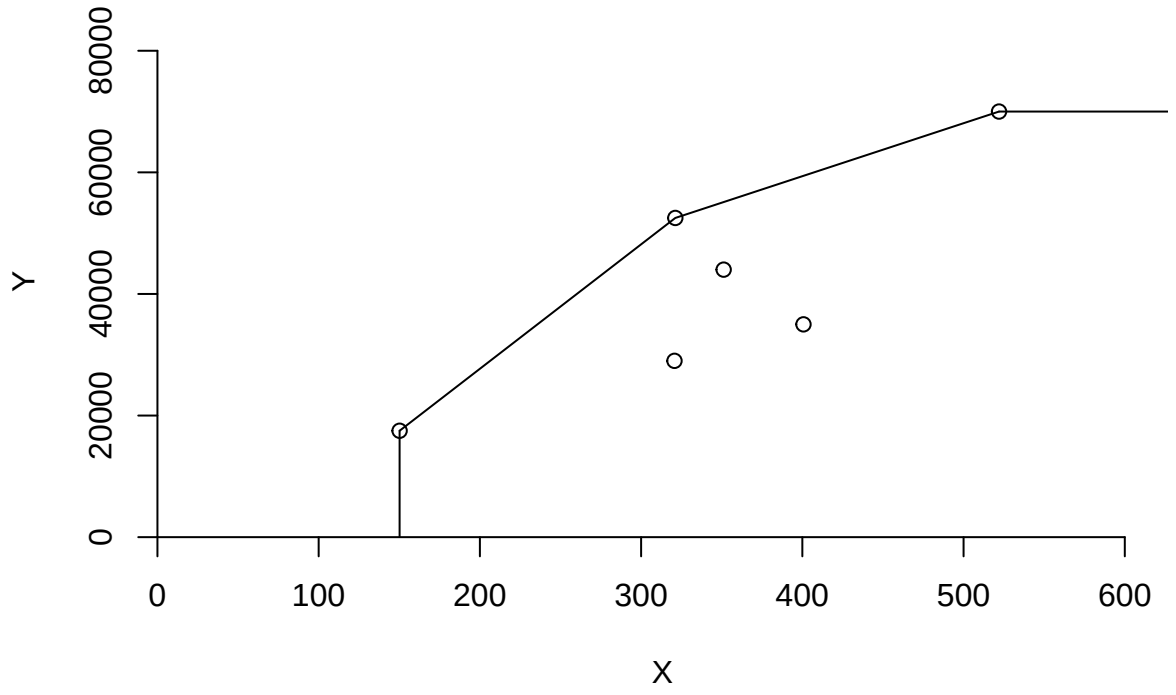
IRS Plot

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



VRS Plot

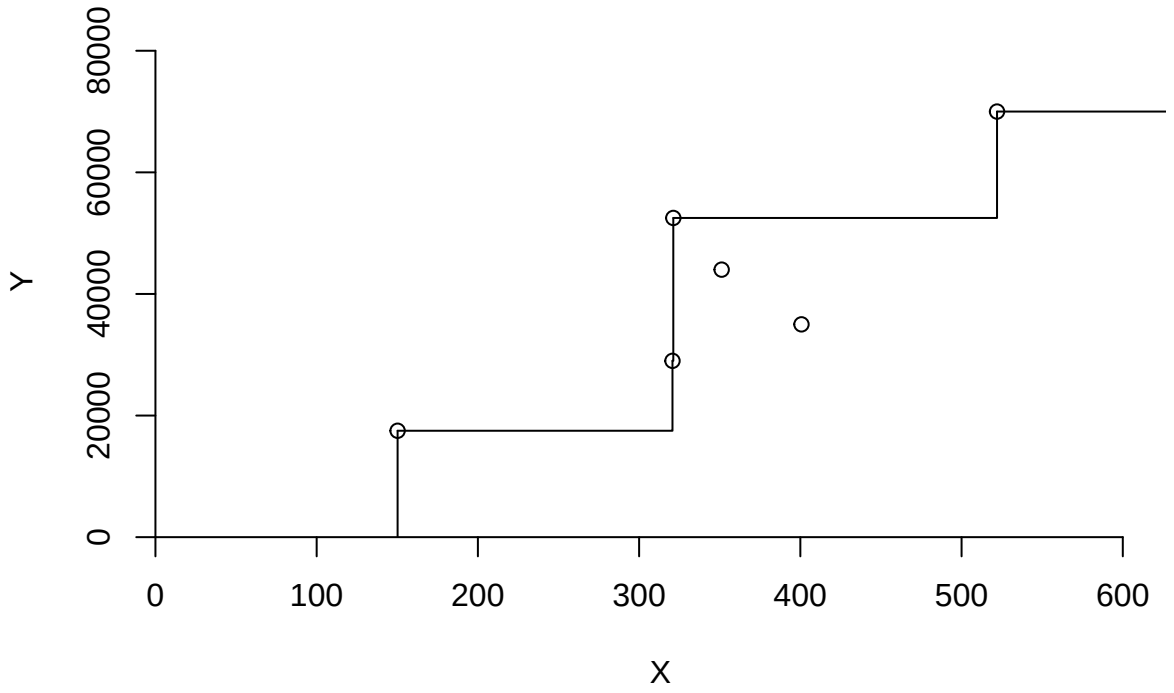
```
dea.plot(x,y,RTS="vrs")
```




```
#tinytex::install_tinytex()
```

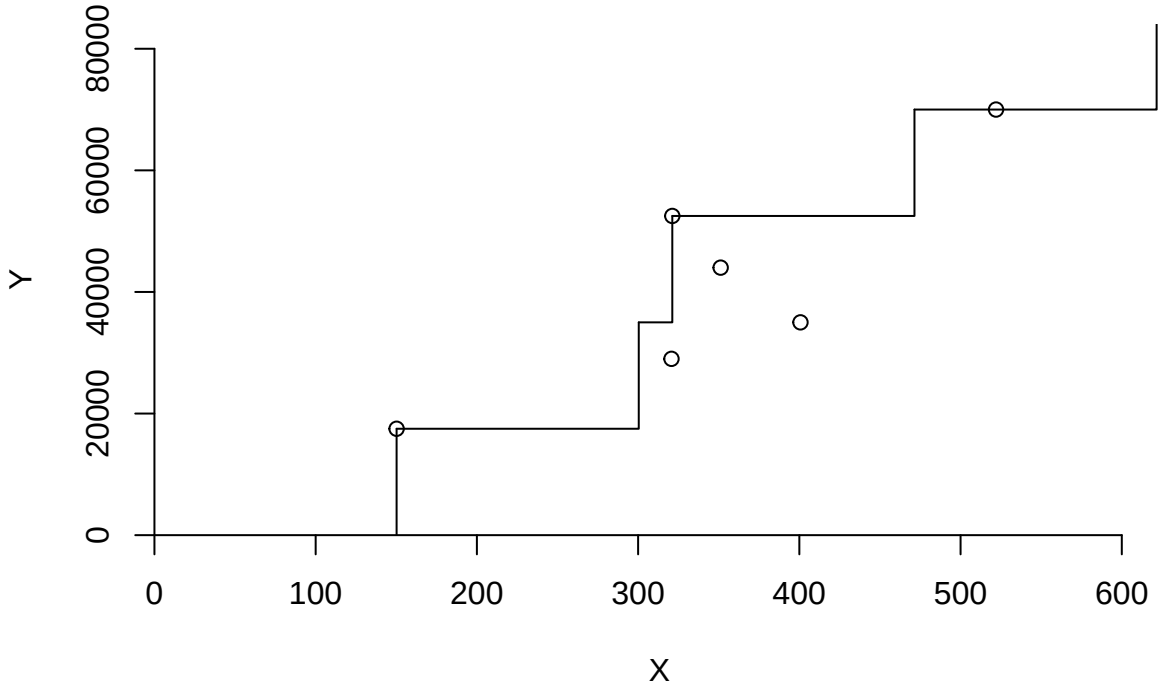
FDH Plot

```
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