Assignment 6 - DEA and Goal Programming

Steve Spence

10/21/2019

Question 1 - Hope Valley Health Care Association

Problem Description – 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.

```
# This package is required for running the DEA functions in this program

require(Benchmarking)

## Loading required package: Benchmarking

## Warning: package 'Benchmarking' was built under R version 3.4.4

## Loading required package: lpSolveAPI

## Loading required package: ucminf
```

Next, the problem data will be loaded into the R environment.

```
# Create matrix for the two inputs

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

# Create matrix for the two outputs

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

# Name the columns of the inputs and outputs

colnames(X) <- c("Staff Hours per Day","Supplies per Day")

colnames(Y) <- c("Reimburse Patient-Days", "Privately Paid Patient-Days")

# Return the matrices for review

print(X)
```

```
##      Staff Hours per Day Supplies per Day
## [1,]                 150              0.2
## [2,]                 400              0.7
## [3,]                 320              1.2
## [4,]                 520              2.0
## [5,]                 350              1.2
## [6,]                 320              0.7
```

```r
print(Y)
```

```
##      Reimburse Patient-Days Privately Paid Patient-Days
## [1,]                  14000                        3500
## [2,]                  14000                       21000
## [3,]                  42000                       10500
## [4,]                  28000                       42000
## [5,]                  19000                       25000
## [6,]                  14000                       15000
```

The following chunk of code will return the results of DEA utilizing the FDH method.

```r
# DEA code utilizing the FDH method

FDH <- rep("FDH", times = 6)
Not_Applicable <- rep(NA, times = 6)
DEA_FDH <- dea(X, Y, RTS = "FDH")
DEA_FDH_Peers <- peers(DEA_FDH)
DEA_FDH_Lambda <- lambda(DEA_FDH)

print(DEA_FDH)
```

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

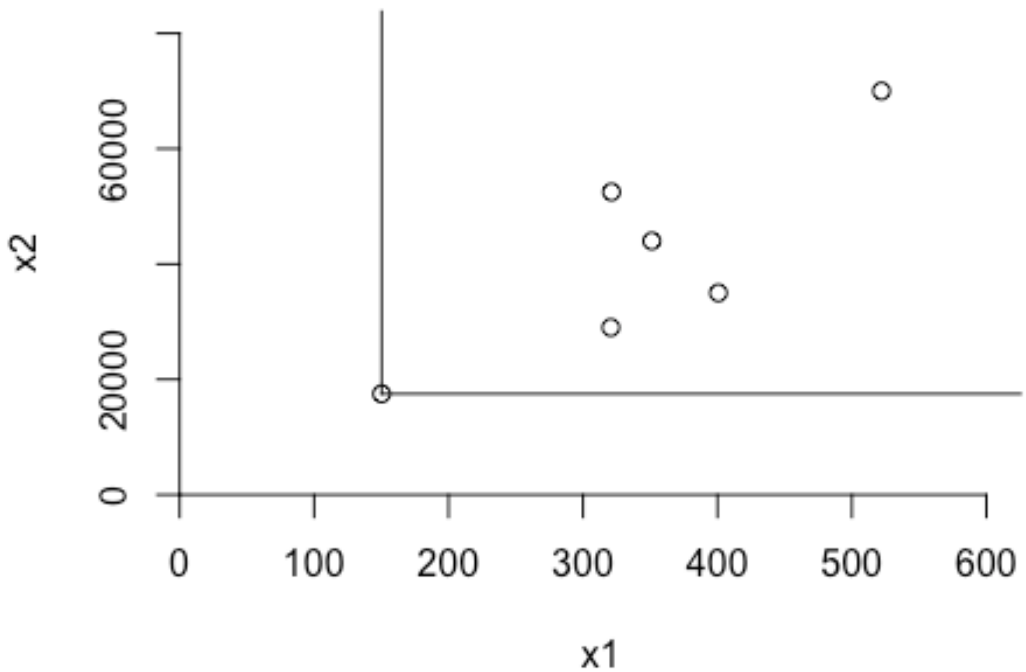
```r
print(DEA_FDH_Peers)
```

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

```r
print(DEA_FDH_Lambda)
```

```
##      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.isoquant(X, Y, RTS= "FDH")
```

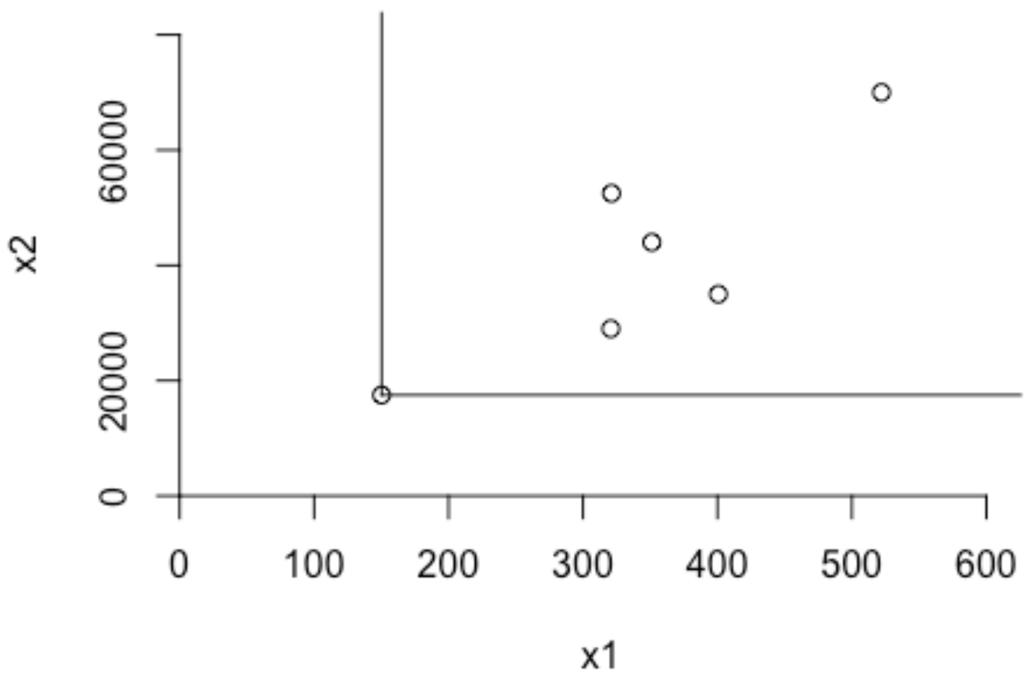



```
# Summarize the results for addition to a summary table

DEA_FDH_Peers <- cbind(DEA_FDH_Peers, Not_Applicable, Not_Applicable)

FDH_Summary <- cbind(FDH, DEA_FDH$eff, DEA_FDH_Peers, DEA_FDH_Lambda)
colnames(FDH_Summary) <- c("Method","Eff", "P1", "P2", "P3", "L1", "L2",
"L3", "L4", "L5", "L6")

print(FDH_Summary)

##      Method Eff P1  P2 P3 L1  L2  L3  L4  L5  L6
## [1,] "FDH"  "1" "1" NA NA "1" "0" "0" "0" "0" "0"
## [2,] "FDH"  "1" "2" NA NA "0" "1" "0" "0" "0" "0"
## [3,] "FDH"  "1" "3" NA NA "0" "0" "1" "0" "0" "0"
## [4,] "FDH"  "1" "4" NA NA "0" "0" "0" "1" "0" "0"
## [5,] "FDH"  "1" "5" NA NA "0" "0" "0" "0" "1" "0"
## [6,] "FDH"  "1" "6" NA NA "0" "0" "0" "0" "0" "1"
```

The following chunk of code will return the results of DEA utilizing the CRS method.

```
# DEA code utilizing the CRS method
```

```r
CRS <- rep("CRS", times = 6)
DEA_CRS <- dea(X, Y, RTS = "CRS")
DEA_CRS_Peers <- peers(DEA_CRS)
DEA_CRS_Lambda <- lambda(DEA_CRS)

print(DEA_CRS)

## [1] 1.0000 1.0000 1.0000 1.0000 0.9775 0.8675

print(DEA_CRS_Peers)

##      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

print(DEA_CRS_Lambda)

##             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.isoquant(X, Y, RTS= "CRS")
```

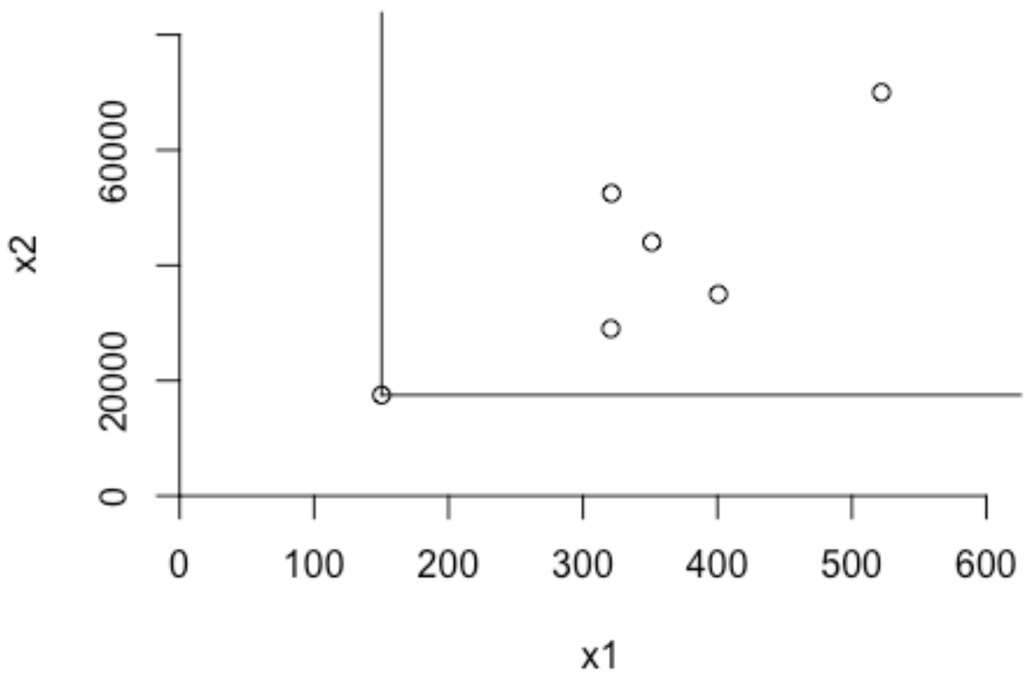



```
# Summarize the results for addition to a summary table

DEA_CRS_Lambda <- cbind(DEA_CRS_Lambda, Not_Applicable, Not_Applicable)

CRS_Summary <- cbind(CRS, DEA_CRS$eff, DEA_CRS_Peers, DEA_CRS_Lambda)
colnames(CRS_Summary) <- c("Method","Eff", "P1", "P2", "P3", "L1", "L2",
"L3", "L4", "L5", "L6")

CRS_Summary <- as.data.frame(CRS_Summary)
CRS_Summary

##   Method               Eff P1   P2   P3                L1
## 1    CRS                 1  1 <NA> <NA>                 1
## 2    CRS                 1  2 <NA> <NA>                 0
## 3    CRS                 1  3 <NA> <NA>                 0
## 4    CRS                 1  4 <NA> <NA>                 0
## 5    CRS 0.977498691784406  1    2    4               0.2
## 6    CRS 0.867452135493372  1    2    4 0.342857142857143
##                   L2 L3                L4   L5   L6
## 1                  0  0                 0 <NA> <NA>
## 2 0.999999999999999  0                 0 <NA> <NA>
## 3                  0  1                 0 <NA> <NA>
## 4                  0  0                 1 <NA> <NA>
```

```
## 5 0.080481423338566  0 0.538330716902146 <NA> <NA>
## 6 0.394992636229749  0 0.131075110456554 <NA> <NA>
```

The following chunk of code will return the results of DEA utilizing the VRS method.

```
# DEA code utilizing the VRS method

VRS <- rep("VRS", times = 6)
DEA_VRS <- dea(X, Y, RTS = "VRS")
DEA_VRS_Peers <- peers(DEA_VRS)
DEA_VRS_Lambda <- lambda(DEA_VRS)

print(DEA_VRS)

## [1] 1.0000 1.0000 1.0000 1.0000 1.0000 0.8963

print(DEA_VRS_Peers)

##      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

print(DEA_VRS_Lambda)

##             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.isoquant(X, Y, RTS= "VRS")
```

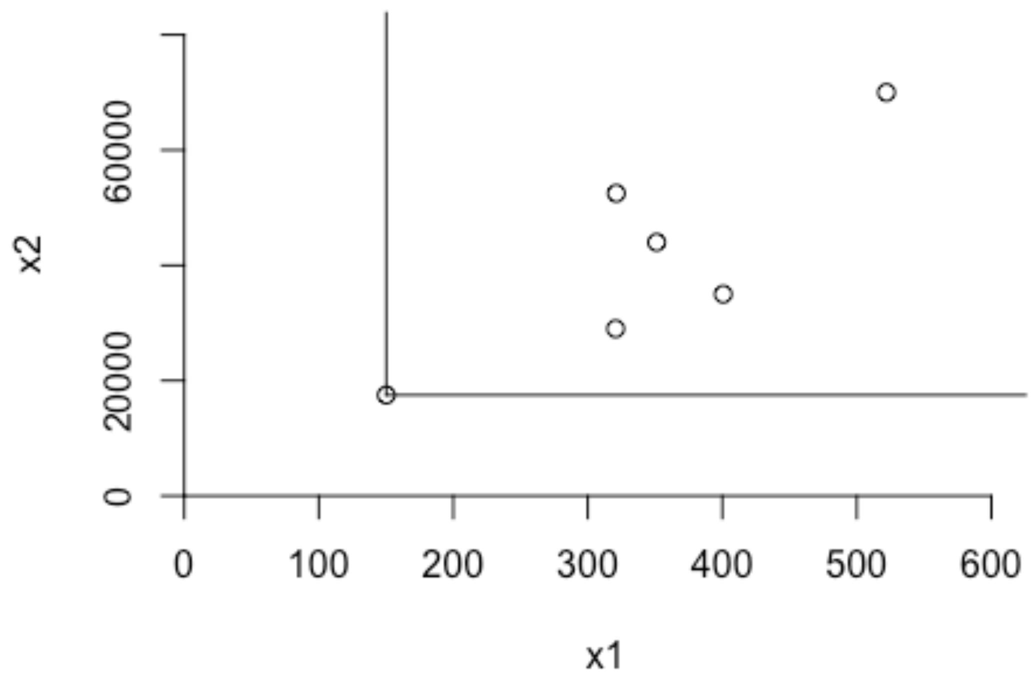



```
# Summarize the results for addition to a summary table

DEA_VRS_Lambda <- cbind(DEA_VRS_Lambda, Not_Applicable)

VRS_Summary <- cbind(VRS, DEA_VRS$eff, DEA_VRS_Peers, DEA_VRS_Lambda)
colnames(VRS_Summary) <- c("Method","Eff", "P1", "P2", "P3", "L1", "L2",
"L3", "L4", "L5", "L6")

VRS_Summary <- as.data.frame(VRS_Summary)
VRS_Summary

##   Method               Eff P1   P2   P3                L1
## 1    VRS                 1  1 <NA> <NA>                 1
## 2    VRS                 1  2 <NA> <NA>                 0
## 3    VRS                 1  3 <NA> <NA>                 0
## 4    VRS                 1  4 <NA> <NA>                 0
## 5    VRS                 1  5 <NA> <NA>                 0
## 6    VRS 0.896328293736501  1    2    5 0.401439884809215
##                  L2 L3 L4                L5   L6
## 1                 0  0  0                 0 <NA>
## 2                 1  0  0                 0 <NA>
## 3                 0  1  0                 0 <NA>
## 4                 0  0  1                 0 <NA>
```

```
## 5                  0  0  0                  1 <NA>
## 6 0.342260619150468  0  0 0.256299496040316 <NA>
```

The following chunk of code will return the results of DEA utilizing the IRS method.

```
# DEA code utilizing the IRS method

IRS <- rep("IRS", times = 6)
DEA_IRS <- dea(X, Y, RTS = "IRS")
DEA_IRS_Peers <- peers(DEA_IRS)
DEA_IRS_Lambda <- lambda(DEA_IRS)

print(DEA_IRS)

## [1] 1.0000 1.0000 1.0000 1.0000 1.0000 0.8963

print(DEA_IRS_Peers)

##      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

print(DEA_IRS_Lambda)

##             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.isoquant(X, Y, RTS= "IRS")
```

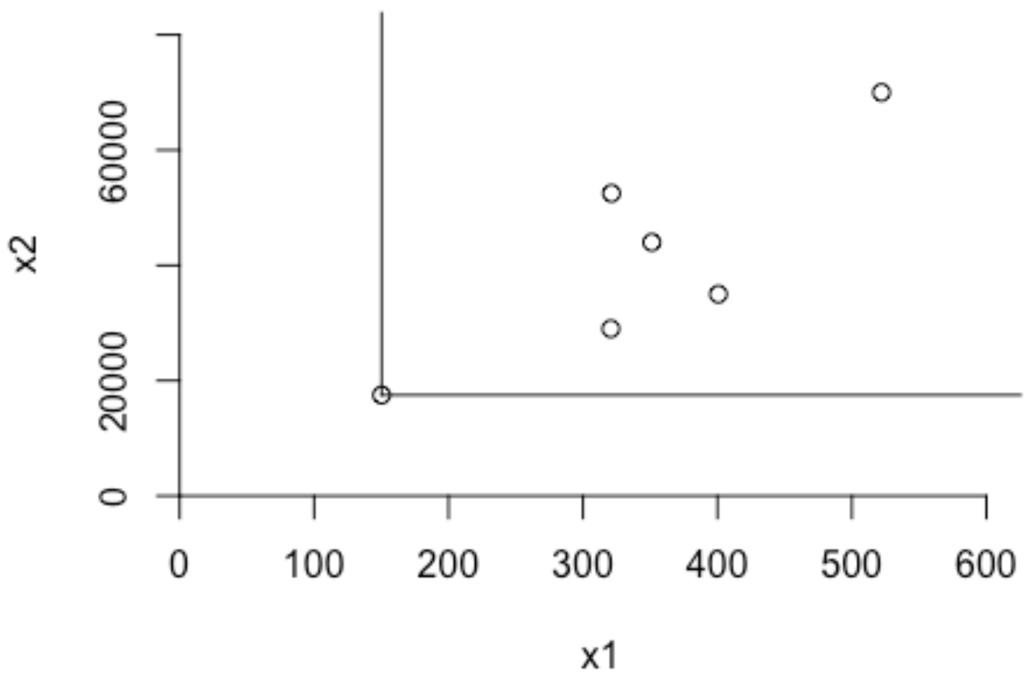



```
# Summarize the results for addition to a summary table

DEA_IRS_Lambda <- cbind(DEA_IRS_Lambda, Not_Applicable)

IRS_Summary <- cbind(IRS, DEA_IRS$eff, DEA_IRS_Peers, DEA_IRS_Lambda)
colnames(IRS_Summary) <- c("Method","Eff", "P1", "P2", "P3", "L1", "L2",
"L3", "L4", "L5", "L6")

IRS_Summary <- as.data.frame(IRS_Summary)
IRS_Summary

##   Method               Eff P1   P2   P3                L1
## 1    IRS                 1  1 <NA> <NA>                 1
## 2    IRS                 1  2 <NA> <NA>                 0
## 3    IRS                 1  3 <NA> <NA>                 0
## 4    IRS                 1  4 <NA> <NA>                 0
## 5    IRS                 1  5 <NA> <NA>                 0
## 6    IRS 0.896328293736501  1    2    5 0.401439884809215
##                  L2 L3 L4                 L5   L6
## 1                 0  0  0                  0 <NA>
## 2                 1  0  0                  0 <NA>
## 3                 0  1  0                  0 <NA>
## 4                 0  0  1                  0 <NA>
```

```
## 5                 0  0  0                 1 <NA>
## 6 0.342260619150468  0  0 0.256299496040316 <NA>
```

The following chunk of code will return the results of DEA utilizing the DRS method.

```
# DEA code utilizing the DRS method

DRS <- rep("DRS", times = 6)
DEA_DRS <- dea(X, Y, RTS = "DRS")
DEA_DRS_Peers <- peers(DEA_DRS)
DEA_DRS_Lambda <- lambda(DEA_DRS)

print(DEA_DRS)

## [1] 1.0000 1.0000 1.0000 1.0000 0.9775 0.8675

print(DEA_DRS_Peers)

##      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

print(DEA_DRS_Lambda)

##             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.isoquant(X, Y, RTS= "DRS")
```




```
# Summarize the results for addition to a summary table

DEA_DRS_Lambda <- cbind(DEA_DRS_Lambda, Not_Applicable, Not_Applicable)

DRS_Summary <- cbind(DRS, DEA_DRS$eff, DEA_DRS_Peers, DEA_DRS_Lambda)
colnames(DRS_Summary) <- c("Method","Eff", "P1", "P2", "P3", "L1", "L2",
"L3", "L4", "L5", "L6")

DRS_Summary <- as.data.frame(DRS_Summary)
DRS_Summary

##   Method               Eff P1   P2   P3                L1
## 1    DRS                 1  1 <NA> <NA>                 1
## 2    DRS                 1  2 <NA> <NA>                 0
## 3    DRS                 1  3 <NA> <NA>                 0
## 4    DRS                 1  4 <NA> <NA>                 0
## 5    DRS 0.977498691784406  1    2    4               0.2
## 6    DRS 0.867452135493373  1    2    4 0.342857142857143
##                   L2 L3                L4   L5   L6
## 1                  0  0                 0 <NA> <NA>
## 2                  1  0                 0 <NA> <NA>
## 3                  0  1                 0 <NA> <NA>
## 4                  0  0                 1 <NA> <NA>
```

```
## 5 0.0804814233385653  0 0.538330716902146 <NA> <NA>
## 6   0.39499263622975  0 0.131075110456554 <NA> <NA>
```

The following chunk of code will return the results of DEA utilizing the FRH/ADD method.

```r
# DEA code utilizing the ADD method

ADD <- rep("ADD", times = 6)
DEA_ADD <- dea(X, Y, RTS = "ADD")
DEA_ADD_Peers <- peers(DEA_ADD)
DEA_ADD_Lambda <- lambda(DEA_ADD)

print(DEA_ADD)

## [1] 1 1 1 1 1 1

print(DEA_ADD_Peers)

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

print(DEA_ADD_Lambda)

##      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

# Summarize the results for addition to a summary table

DEA_ADD_Peers <- cbind(DEA_ADD_Peers, Not_Applicable, Not_Applicable)

ADD_Summary <- cbind(ADD, DEA_ADD$eff, DEA_ADD_Peers, DEA_ADD_Lambda)
colnames(ADD_Summary) <- c("Method","Eff", "P1", "P2", "P3", "L1", "L2",
"L3", "L4", "L5", "L6")

ADD_Summary <- as.data.frame(ADD_Summary)
ADD_Summary

##   Method Eff P1   P2   P3 L1 L2 L3 L4 L5 L6
## 1    ADD   1  1 <NA> <NA>  1  0  0  0  0  0
## 2    ADD   1  2 <NA> <NA>  0  1  0  0  0  0
## 3    ADD   1  3 <NA> <NA>  0  0  1  0  0  0
```

```
## 4    ADD   1  4 <NA> <NA>  0  0  0  1  0  0
## 5    ADD   1  5 <NA> <NA>  0  0  0  0  1  0
## 6    ADD   1  6 <NA> <NA>  0  0  0  0  0  1
```

```r
# Combine all of the method summary tables into one large summary table for
each method

Summary_Table <- rbind(FDH_Summary, CRS_Summary, VRS_Summary, IRS_Summary,
DRS_Summary, ADD_Summary)

# Return the summary table for review

print(Summary_Table)
```

```
##    Method               Eff P1   P2   P3                L1
## 1     FDH                 1  1 <NA> <NA>                 1
## 2     FDH                 1  2 <NA> <NA>                 0
## 3     FDH                 1  3 <NA> <NA>                 0
## 4     FDH                 1  4 <NA> <NA>                 0
## 5     FDH                 1  5 <NA> <NA>                 0
## 6     FDH                 1  6 <NA> <NA>                 0
## 7     CRS                 1  1 <NA> <NA>                 1
## 8     CRS                 1  2 <NA> <NA>                 0
## 9     CRS                 1  3 <NA> <NA>                 0
## 10    CRS                 1  4 <NA> <NA>                 0
## 11    CRS 0.977498691784406  1    2    4               0.2
## 12    CRS 0.867452135493372  1    2    4 0.342857142857143
## 13    VRS                 1  1 <NA> <NA>                 1
## 14    VRS                 1  2 <NA> <NA>                 0
## 15    VRS                 1  3 <NA> <NA>                 0
## 16    VRS                 1  4 <NA> <NA>                 0
## 17    VRS                 1  5 <NA> <NA>                 0
## 18    VRS 0.896328293736501  1    2    5 0.401439884809215
## 19    IRS                 1  1 <NA> <NA>                 1
## 20    IRS                 1  2 <NA> <NA>                 0
## 21    IRS                 1  3 <NA> <NA>                 0
## 22    IRS                 1  4 <NA> <NA>                 0
## 23    IRS                 1  5 <NA> <NA>                 0
## 24    IRS 0.896328293736501  1    2    5 0.401439884809215
## 25    DRS                 1  1 <NA> <NA>                 1
## 26    DRS                 1  2 <NA> <NA>                 0
## 27    DRS                 1  3 <NA> <NA>                 0
## 28    DRS                 1  4 <NA> <NA>                 0
## 29    DRS 0.977498691784406  1    2    4               0.2
## 30    DRS 0.867452135493373  1    2    4 0.342857142857143
## 31    ADD                 1  1 <NA> <NA>                 1
## 32    ADD                 1  2 <NA> <NA>                 0
## 33    ADD                 1  3 <NA> <NA>                 0
## 34    ADD                 1  4 <NA> <NA>                 0
## 35    ADD                 1  5 <NA> <NA>                 0
```

```
## 36    ADD                 1  6 <NA> <NA>                 0
##                     L2 L3                 L4                L5   L6
## 1                    0  0                  0                 0    0
## 2                    1  0                  0                 0    0
## 3                    0  1                  0                 0    0
## 4                    0  0                  1                 0    0
## 5                    0  0                  0                 1    0
## 6                    0  0                  0                 0    1
## 7                    0  0                  0              <NA> <NA>
## 8   0.999999999999999  0                  0              <NA> <NA>
## 9                    0  1                  0              <NA> <NA>
## 10                   0  0                  1              <NA> <NA>
## 11  0.080481423338566  0 0.538330716902146              <NA> <NA>
## 12  0.394992636229749  0 0.131075110456554              <NA> <NA>
## 13                   0  0                  0                 0 <NA>
## 14                   1  0                  0                 0 <NA>
## 15                   0  1                  0                 0 <NA>
## 16                   0  0                  1                 0 <NA>
## 17                   0  0                  0                 1 <NA>
## 18  0.342260619150468  0                  0 0.256299496040316 <NA>
## 19                   0  0                  0                 0 <NA>
## 20                   1  0                  0                 0 <NA>
## 21                   0  1                  0                 0 <NA>
## 22                   0  0                  1                 0 <NA>
## 23                   0  0                  0                 1 <NA>
## 24  0.342260619150468  0                  0 0.256299496040316 <NA>
## 25                   0  0                  0              <NA> <NA>
## 26                   1  0                  0              <NA> <NA>
## 27                   0  1                  0              <NA> <NA>
## 28                   0  0                  1              <NA> <NA>
## 29 0.0804814233385653  0 0.538330716902146              <NA> <NA>
## 30   0.39499263622975  0 0.131075110456554              <NA> <NA>
## 31                   0  0                  0                 0    0
## 32                   1  0                  0                 0    0
## 33                   0  1                  0                 0    0
## 34                   0  0                  1                 0    0
## 35                   0  0                  0                 1    0
## 36                   0  0                  0                 0    1
```

After reviewing the summary table, it can be seen that te FRH and FDH methods both return efficiencies of 1.0, as well as identical peer and lambda values, for all six DMUs. The CRS method found DMU[1:4] to be efficient at 1.0. The VRS method found DMU[1:5] to be efficient at 1.0. IRS found DMU[1:5] to be efficient at 1.0, and the DRS method found DMU[1:4] to be efficient at 1.0. All of the less efficient DMUs had a Peer[1] and Peer [2] value of 1 and 2, respectively; however, the Peer[3] value was either 4 or 5, depending on the method. Additionaly, the relative weights (lambdas) for the same DMU across all methods were relatively close.

Question 2 - Research and Development Division of Emax Corporation

Based on the problem statement, the goal is to:

Maximize $Z = P - 6C - 3D$

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.

Subject to:

Total Profit: Maximize $P = 20X1 + 15X2 + 25*X3$

Employment Level: $6X1 + 4X2 + 5*X3 = 50$

Earnings Next Year: $8X1 + 7X2 + 5*X3 >= 75$

As a result, the auxillery variables become:

$Y1 = 6X1 + 4X2 + 5X3 - 50$ $Y2 = 8X1 + 7X2 + 5X3 - 75$

Which becomes:

$(Y1P - Y1M) = 6X1 + 4X2 + 5X3 - 50$ $(Y2P - Y2M) = 8X1 + 7X2 + 5X3 - 75$

Therefore, the final setup of the problem statement is:

Maximize $Z = 20X1 + 15X2 + 25X3 - 6Y1P - 6Y1M - 3Y2M$

Subject to:

$6X1 + 4X2 + 5X3 - (Y1P - Y1M) = 50$ $8X1 + 7X2 + 5X3 - (Y2P - Y2M) = 75$

And:

$X1, X2, X3 >= 0$ $Y1P, Y1M, Y2P, Y2M >= 0$

Lastly, we will run this problem in R as a linear programming model and discuss the results.

```r
# This problem will require the "lpSolveAPI" library

require(lpSolveAPI)

# Import the .lp file for this problem

lpm <- read.lp("emax.lp")

# Return the linear programming model

lpm
```

```
## Model name:
##             X1    X2    X3   Y1P   Y1M   Y2M   Y2P
## Maximize    20    15    25    -6    -6    -3     0
## R1           6     4     5    -1     1     0     0  =  50
## R2           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
# Solve the linear programming model

solve(lpm)

## [1] 0

get.objective(lpm)

## [1] 225

get.variables(lpm)

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

Based on the output of the linear programming model, we can conclude several things.

X1 = 0 X2 = 0 X3 = 15 Y1P = 25 Y1M = 0 Y2M = 0 Y2P = 0

Therefore, we can conclude that the product mix should only contain product 3. With this mix, there would be an object value of 225 units. The goal for earnings next year is fully met; however, the employment level goal will be exceeded by 25 units, which correlates to 2,500 employees and a penalty of 150 units to the objective function.