(SMM641) - Revenue Management & Pricing. Pricing Optimization with Consumer Choice - R Supplement

Oben Ceryan

Contents

1 Required Packages and Data

1.1 Use Package nloptr

Type the following code to install and activate the **nloptr** package:

```
install.packages("nloptr",repos = "http://cran.us.r-project.org")
```

```
##
##   There is a binary version available but the source version is later:
##          binary  source needs_compilation
## nloptr 1.2.2.1 1.2.2.2              TRUE
```

```
library(nloptr)
```

1.2 Use Package stargazer

Type the following code to install and activate the **stargazer** package:

```
install.packages("stargazer",repos = "http://cran.us.r-project.org")
```

```
##
## The downloaded binary packages are in
##  /var/folders/rx/_gm3py093nd25flx8hwzm1bm0000gp/T//Rtmpt86jZz/downloaded_packages
```

```
library(stargazer)
```

1.3 Use Package lattice

Type the following code to install and activate the **lattice** package:

```
install.packages("lattice",repos = "http://cran.us.r-project.org")
```

```
##
##   There is a binary version available but the source version is later:
##           binary  source needs_compilation
## lattice 0.20-40 0.20-41              TRUE
```

```
library(lattice)
```

1.4 Data Files

Download and place in the R working directory the data files:

- "NYHCSurvey.csv"

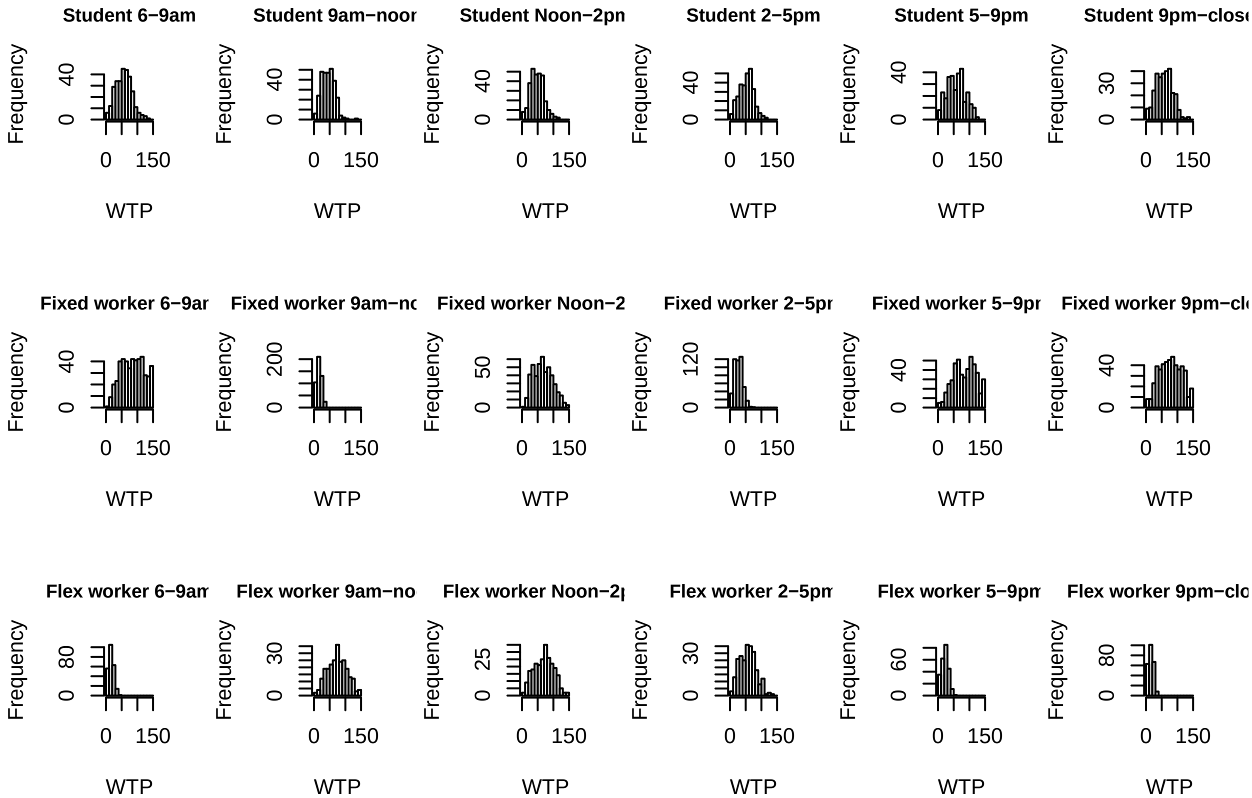
2 Case Study: New York Health Club Part A

2.1 Visualizing Data

```
# Read the Survey Data
nyhc = read.csv("NYHCSurvey.csv",header=T)
# Row count
N=nrow(nyhc)
# Print top few rows of data
head(nyhc)
```

```
##   Client Type TimeSlot1 TimeSlot2 TimeSlot3 TimeSlot4 TimeSlot5 TimeSlot6
## 1      1    1        18        50        41        76        69        41
## 2      2    1        66        14        86        62        71        46
## 3      3    1        60        43        43        26        91        58
## 4      4    1        59        48        62        15        91        15
## 5      5    1        21        24        59        45        81       131
## 6      6    1        66        69        29        80        37        79
```

```
# Generating list of labels for plotting later
list_client_type = c("Student", "Fixed worker", "Flex worker")
list_time_slots = c("6-9am", "9am-noon", "Noon-2pm","2-5pm","5-9pm","9pm-close")

# Histograms of WTPs
defpar = par() # saving default graphical parameters to reset later
par(mfrow=c(3,6))
for (i in 1:3){
  for (j in 3:8){
    temp=subset(nyhc, Type == i)
    hist(temp[,j], breaks=seq(0,150, by=10), main = paste(list_client_type[i],list_time_slots[j
  }
}
```




```r
# reset plot area
par(defpar)
temp=subset(nyhc, Type == 1)[3:8]
meanStudentWTP=colMeans(temp)
```

2.2 Setting a Single Price across all time slots and all client types

```r
# Compute maximum willingness to pay for each client across time slots and enter it as a new c
for (i in 1:N){
    nyhc$maxWTP[i]=max(nyhc[i,3:8])
}
# The above can also be achieved without a for loop by typing
# nyhc$maxWTP=apply(nyhc[,3:8],1,max)

#Displaying the first ten rows of data including the maxWTP for each client
nyhc[1:10,]
```

```
##    Client Type TimeSlot1 TimeSlot2 TimeSlot3 TimeSlot4 TimeSlot5 TimeSlot6
## 1       1    1        18        50        41        76        69        41
## 2       2    1        66        14        86        62        71        46
```

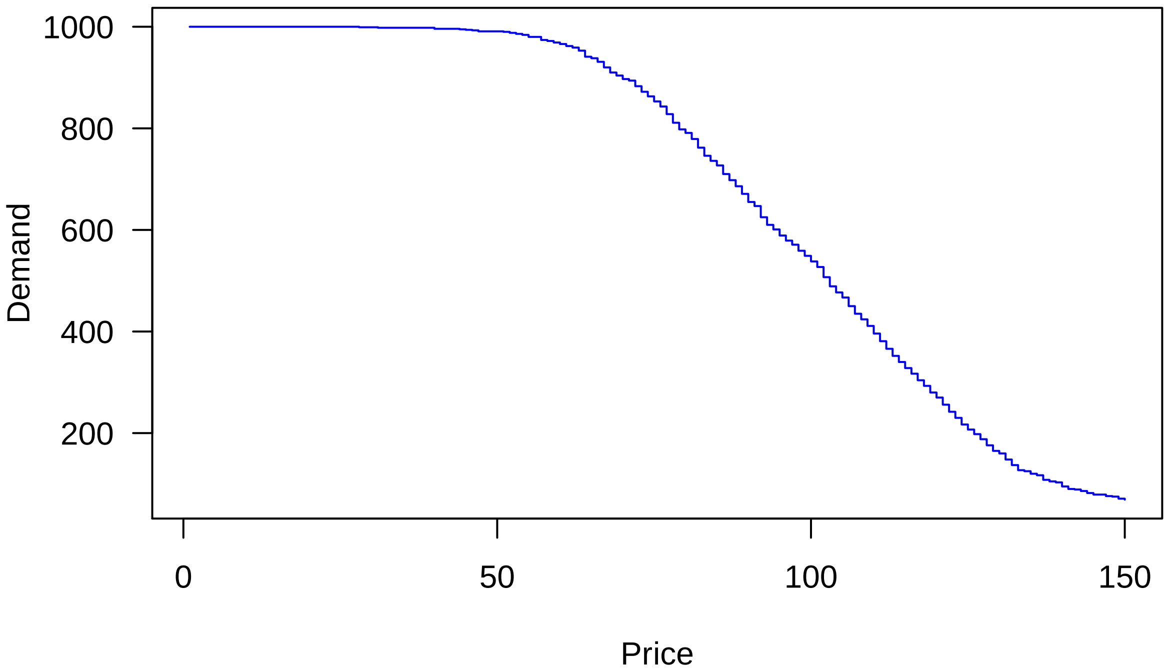
```
## 3       3   1        60        43        43        26        91        58
## 4       4   1        59        48        62        15        91        15
## 5       5   1        21        24        59        45        81       131
## 6       6   1        66        69        29        80        37        79
## 7       7   1       126        41        61        97       111        56
## 8       8   1        72        71        40        11        78       116
## 9       9   1        49        64        54        26        49       107
## 10     10   1        17        21        41        54        76        27
##    maxWTP
## 1      76
## 2      86
## 3      91
## 4      91
## 5     131
## 6      80
## 7     126
## 8     116
## 9     107
## 10     76
```

```r
# The maximium WTP in data, we can use this as the upper bound for our price search.
# No need to consider a price if noone can afford it.
maxprice=max(nyhc$maxWTP)

# Defining empty array variables we will be introducing
demand=rep(NA,maxprice)
revenue=rep(NA,maxprice)

# Find how many people buy at each price level
for (p in 1:maxprice){
    # For any price level, say for p=75,
    # nyhc$maxWTP>=75 assigns a true/false value for each element in nyhc max WTP column
    # (note there are N=1000 elements (rows), one for each client)
    # the value is TRUE if for that particular client has a maxWTP greater than or equal to 75
    # That means the client will buy if their maxWTP is greater than or equal to.
    # The value is FALSE if client's maxWTP is less than 75.
    # The sum() function adds up all the values.  It treats TRUEs as 1, and FALSEs as 0.
    # So if for a price, say 75, there are 853 TRUEs, the demand at that price level is 853.
    demand[p]=sum(nyhc$maxWTP>=p)
    revenue[p]=p*demand[p]
```

```
}

# Identifying the Best Price
revenueBest=max(revenue)
priceBest=which(revenue == revenueBest)
print(paste("If a single price is to be charged across all time slots, the optimal price is:",
```

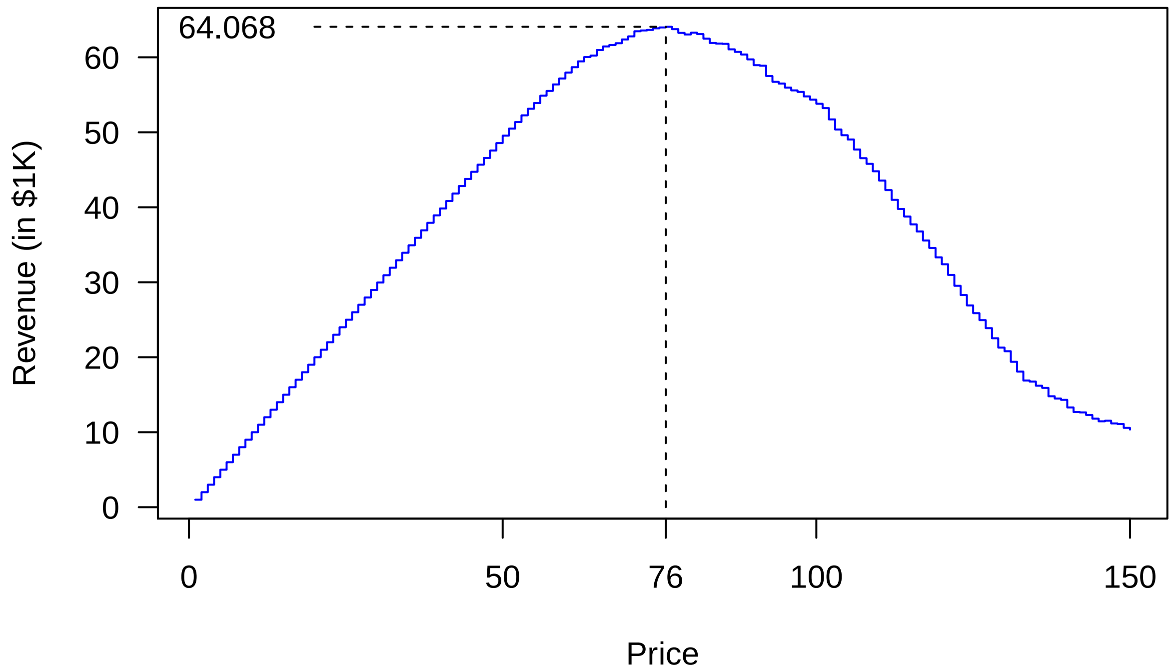
```
## [1] "If a single price is to be charged across all time slots, the optimal price is: 76"
```

```
# Plotting Demand vs Price
xaxis=1:maxprice
plot(xaxis,demand,pch = 16, type="s", col="blue", las=1, xaxt="n",
     xlab="Price",ylab="Demand")
xticks <- seq(0, maxprice, by=50)
axis(side = 1, at = xticks)
```




```
# Plotting Revenue vs Price
xaxis=1:maxprice
plot(xaxis,revenue/1000,pch = 16, type="s",col="blue",las=1, xaxt="n",
     xlab="Price",ylab="Revenue (in $1K)")
xticks <- seq(0, maxprice, by=50)
axis(side = 1, at = xticks)
axis(side = 1, at = priceBest)
lines(c(priceBest,priceBest),c(0, revenueBest/1000),lty=2)
axis(side = 2, at = round(revenueBest/1000,3),las=1,pos=20, tick=F)
```

```r
lines(c(20,priceBest),c(revenueBest/1000, revenueBest/1000),lty=2)
```



2.3 Setting a Peak Period Price (assuming a given NonPeak Period Price)

```r
# All Time Slots except TimeSlot5 are priced at 65
# We will pick a price for TimeSlot5

# Read the Survey Data
# nyhc = read.csv("NYHCSurvey.csv")
N=nrow(nyhc)

# The maximium WTP in data, we can use this as the upper bound for our price search.
# No need to consider a price if noone can afford it.
maxprice=max(nyhc[3:8])

# We will refer to all time slots except time slot 5 as NonPeak.
# Similarly timeslot 5 that is between 5-9pm is referred to as Peak

# Price for all time slots (i.e., nonpeak) except for time slot 5 (5-9pm) (peak)
basePrice=65

# Obtaining maximum surpluses among nonpeak time slots
# We will later introduce the following variables and so we define them first with correct dim
```

```r
# Note: Generally, we wouldn't know what to introduce or their sizes immediately.
# Start by writing the code.
# As we need to introduce variables, we do them first in the code.
# We might introduce additional temporary variables too if we would like to keep track of thin
# Once we are done with that part of the code, we go back and declare those variables with cor
# as done below:


demandNonPeak<-rep(0,maxprice)
demandPeak<-rep(0,maxprice)
revenue<-rep(0,maxprice)



# STEP 1:
# For each client we will obtain their maximum WTP and
# maximum Surplus among the Nonpeak time slots
# Columns 3 to 6 and Column 8 are the NonPeak time slots.

maxWTPNonPeak<-rep(0,N)
maxsurplusNonPeak<-rep(0,N)

for (i in 1:N){
    maxWTPNonPeak[i]=max(nyhc[i,c(3:6,8)])
    maxsurplusNonPeak[i]=max(nyhc[i,c(3:6,8)]-basePrice)

    # We can also generate new column(s) and add this information to our daya nyhc:
    nyhc$maxWTPNonPeak[i]=max(nyhc[i,c(3:6,8)])
    nyhc$maxsurplusNonPeak[i]=max(nyhc[i,c(3:6,8)]-basePrice)
}

# Viewing the first ten rows of data
nyhc[1:10,]
```

```
##    Client Type TimeSlot1 TimeSlot2 TimeSlot3 TimeSlot4 TimeSlot5 TimeSlot6
## 1       1    1        18        50        41        76        69        41
## 2       2    1        66        14        86        62        71        46
## 3       3    1        60        43        43        26        91        58
## 4       4    1        59        48        62        15        91        15
## 5       5    1        21        24        59        45        81       131
```

```
## 6         6    1          66         69         29         80         37         79
## 7         7    1         126         41         61         97        111         56
## 8         8    1          72         71         40         11         78        116
## 9         9    1          49         64         54         26         49        107
## 10       10    1          17         21         41         54         76         27
##     maxWTP maxWTPNonPeak maxsurplusNonPeak
## 1       76            76                11
## 2       86            86                21
## 3       91            60                -5
## 4       91            62                -3
## 5      131           131                66
## 6       80            80                15
## 7      126           126                61
## 8      116           116                51
## 9      107           107                42
## 10      76            54               -11
```

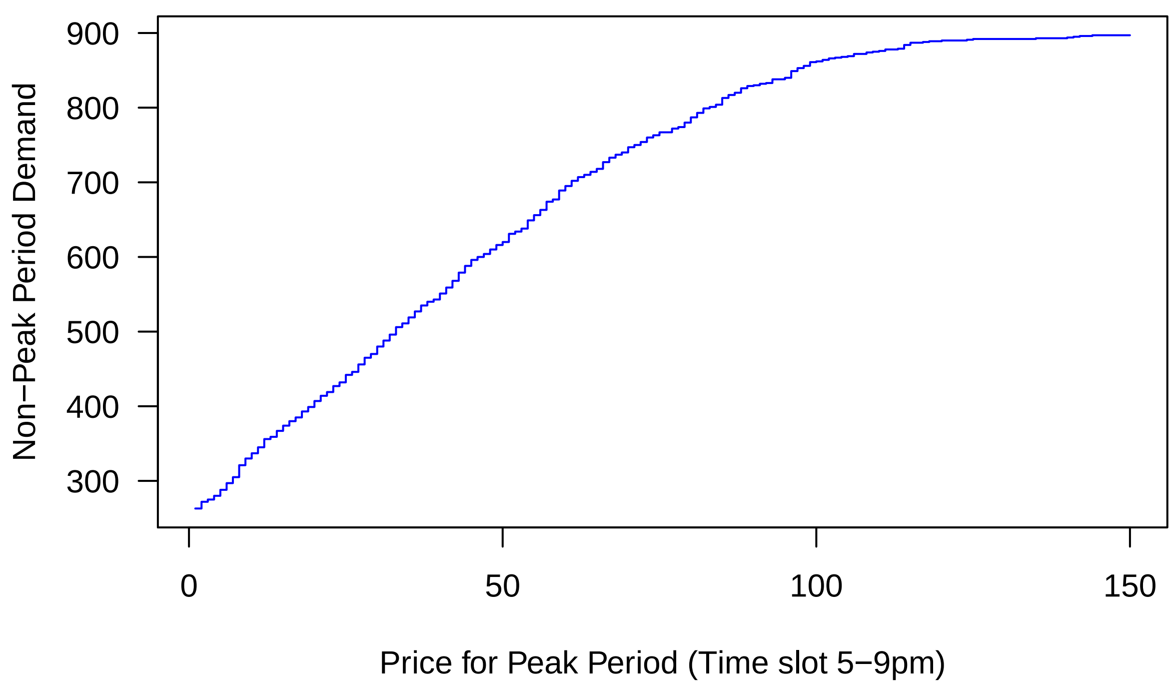
```r
# STEP 2:
# For each possible price point:
# and for all clients at the particular price point currently in consideration:

# Compare a client's surplus from NonPeak and Peak

# If a client's surplus from NonPeak is greater than their surplus for Peak
# and if the client's surplus from NonPeak is greater than 0,
# That client will purchase NonPeak.

# If a client's surplus from Peak is greater than their surplus for NonPeak
# and if the client's surplus from Peak is greater than 0,
# That client will purchase Peak.

# If both surpluses are less than 0, the customer will not buy.



# Let's first compute clients' surpluses for Peak across all possible Peak Price choices
# There are 1000 clients and 150 possible price choices
# So we will create a matrix of dimension: 1000 rows (for each client) and 150 Columns
# where for example, the element [10,50] is Client 10's surplus from Peak if Peak Price is 50.

surplusPeak<-matrix(0,N,maxprice)
```

```r
for (p in 1:maxprice){
    for (i in 1:N){
        surplusPeak[i,p]=nyhc[i,7]-p
    }
}

# Viewing a part of data in surplusPeak
# the first ten client's surpluses for Peak time slot if pPeak Price p=50, 51, 52, ..., 60.
colnames(surplusPeak)=paste0("p=",1:maxprice)
surplusPeak[1:10,c(65,70,75,80,85,90,95,100)]
```

```
##       p=65 p=70 p=75 p=80 p=85 p=90 p=95 p=100
##  [1,]    4   -1   -6  -11  -16  -21  -26   -31
##  [2,]    6    1   -4   -9  -14  -19  -24   -29
##  [3,]   26   21   16   11    6    1   -4    -9
##  [4,]   26   21   16   11    6    1   -4    -9
##  [5,]   16   11    6    1   -4   -9  -14   -19
##  [6,]  -28  -33  -38  -43  -48  -53  -58   -63
##  [7,]   46   41   36   31   26   21   16    11
##  [8,]   13    8    3   -2   -7  -12  -17   -22
##  [9,]  -16  -21  -26  -31  -36  -41  -46   -51
## [10,]   11    6    1   -4   -9  -14  -19   -24
```

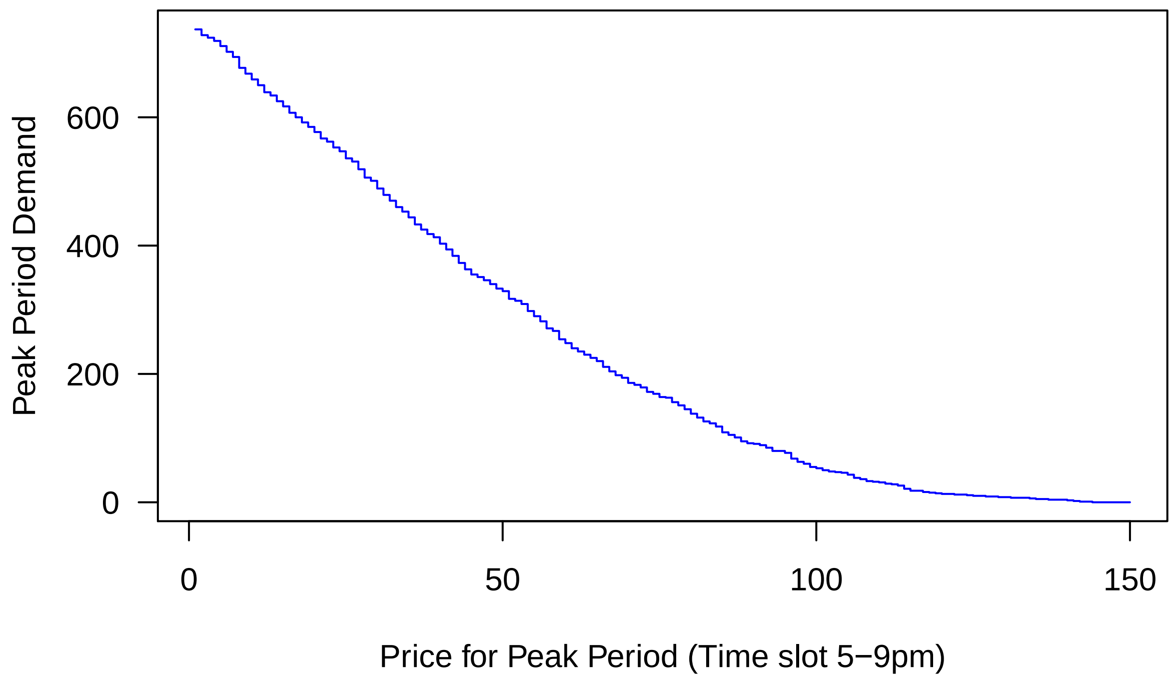
```r
# Next, let's compare each client's surplus from NonPeak and Peak for each Peak price point p
# And for each of these price pints p's, we will count how many clients will buy NonPeak and
# how many clients will buy Peak.

# surplusPeak[,p] corresponds to the surpluses across all clients at a particular price p
# (maxsurplusNonPeak>surplusPeak[,p]) returns an array of TRUEs and FALSEs  each element
# corresponding to each Client.
# If for a Client, their maximum surplus from Non Peak exceeds their surplus from Peak
# Then they will prefer NonPeak, indicated by TRUE.  Otherwise for that client we have FALSE.
# The Client also needs that their maximum surplus from Non Peak is >= 0 to buy.
# If everything is negative, they cannot be forced to buy.
# The (maxsurplusNonPeak>=0) returns TRUE if their maximum surplus from Non Peak is >= 0
# and returns FALSE otherwise.
# The multiplication of these two logical variables is TRUE only if both are TRUE.
# The sum function adds ip all TRUE values (treating them as 1s)
# and hence gives total NonPeak demand at price p.
```

```r
# The logic for demandPeak[p] is the same.



for (p in 1:maxprice){
  demandNonPeak[p]=sum((maxsurplusNonPeak>surplusPeak[,p])*(maxsurplusNonPeak>=0))
  demandPeak[p]=sum((surplusPeak[,p]>=maxsurplusNonPeak)*(surplusPeak[,p]>=0))
  revenue[p]=basePrice*demandNonPeak[p]+p*demandPeak[p]
}


# Plotting NonPeak Demand vs Peak Period Price
xaxis=1:maxprice
plot(xaxis,demandNonPeak,pch = 16, type="s",col="blue", las=1, xaxt="n",
     xlab="Price for Peak Period (Time slot 5-9pm)",ylab="Non-Peak Period Demand")
xticks <- seq(0, maxprice, by=50)
axis(side = 1, at = xticks)
```

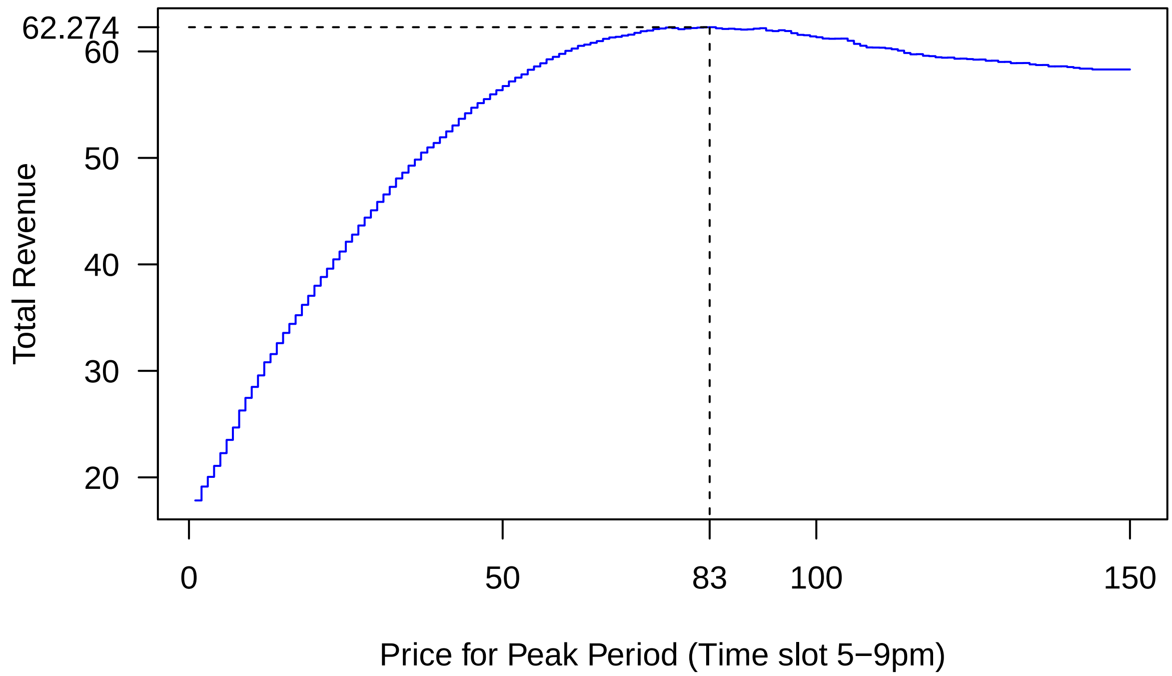



```r
# Plotting Peak Demand vs Peak Period Price
xaxis=1:maxprice
plot(xaxis,demandPeak,pch = 16, type="s",col="blue", las=1, xaxt="n",
     xlab="Price for Peak Period (Time slot 5-9pm)",ylab="Peak Period Demand")
xticks <- seq(0, maxprice, by=50)
```

```
axis(side = 1, at = xticks)
```




```
# Plotting Revenue vs Peak Period Price
xaxis=1:maxprice
plot(xaxis,revenue/1000,pch = 16, type="s",col="blue", las=1, xaxt="n",
     xlab="Price for Peak Period (Time slot 5-9pm)",ylab="Total Revenue")
xticks <- seq(0, maxprice, by=50)
axis(side = 1, at = xticks)
revenueBest=max(revenue[basePrice:maxprice])
priceBest=which(revenue == revenueBest)
axis(side = 1, at = priceBest)
lines(c(priceBest,priceBest),c(0, revenueBest/1000),lty=2)
axis(side = 2, at = round(revenueBest/1000,3),las=1)
lines(c(0,priceBest),c(revenueBest/1000, revenueBest/1000),lty=2)
```

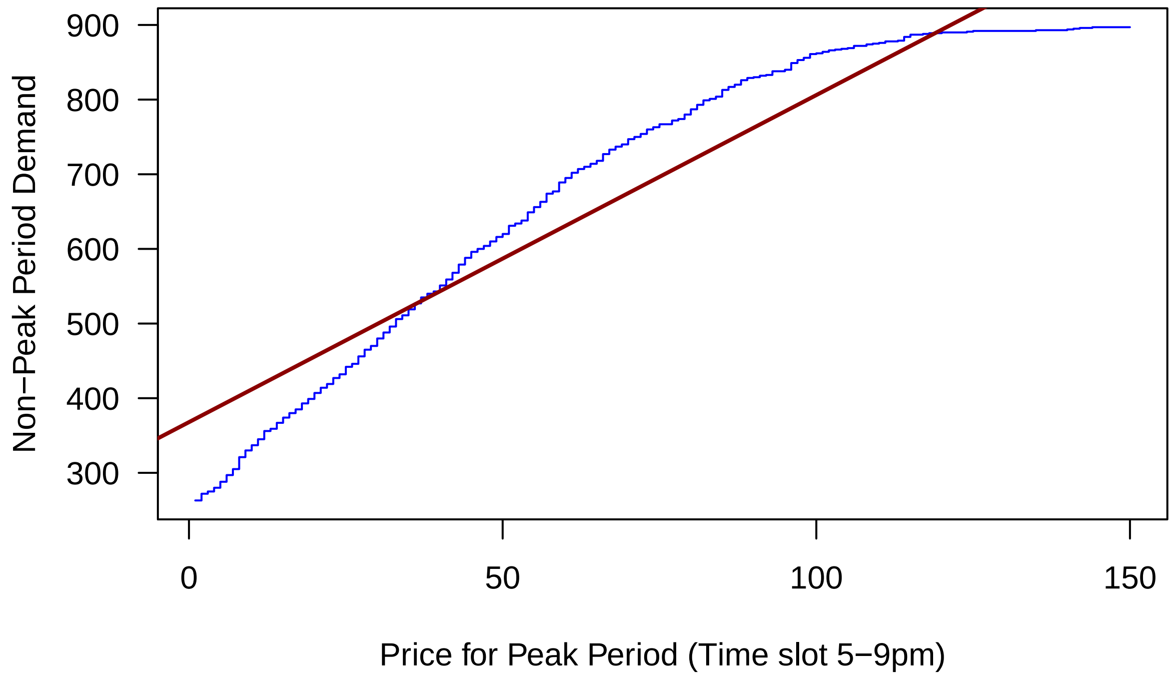



```
print(paste("When other periods have a base price of 65, the optimal price for the 5-9pm slot i
```

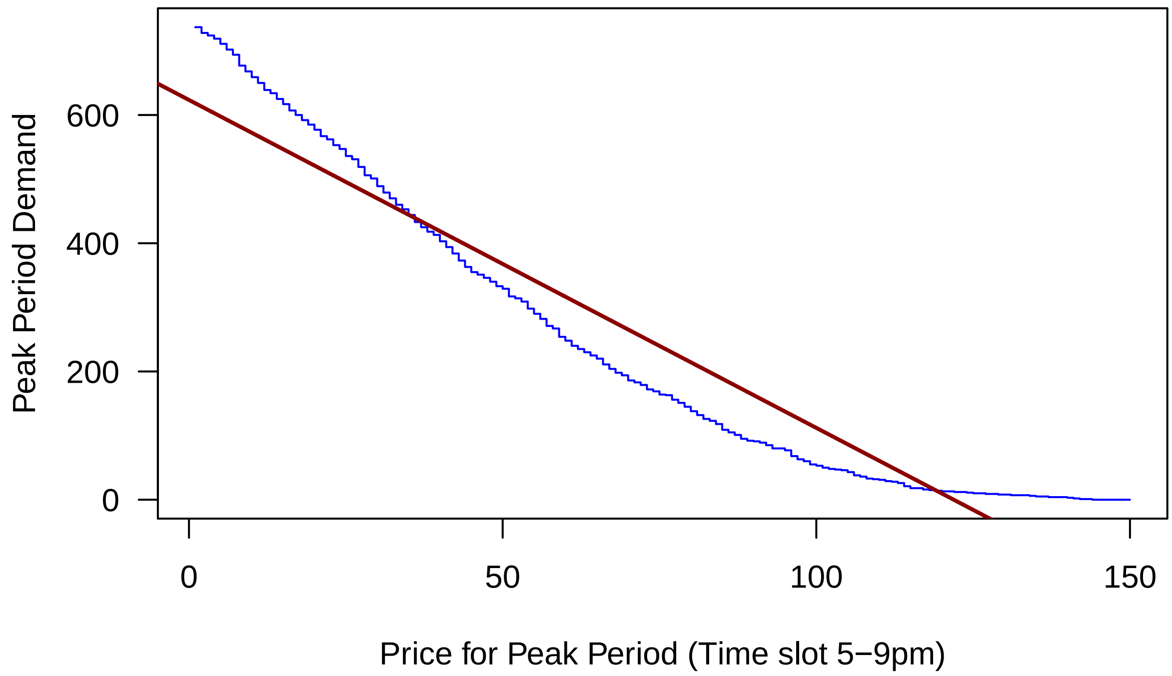
```
## [1] "When other periods have a base price of 65, the optimal price for the 5-9pm slot is: 83
```

2.4 Setting a Peak Period Price through Fitting Linear Demand Functions

```
# STEP 3: Fitting a linear demand model

PeakPrice=1:maxprice

# Fitting a Linear Demand Model for Non-Peak Period Demand
fitNonPeak <-lm(demandNonPeak ~ PeakPrice)
InterceptNonPeak=coef(fitNonPeak)[1]
CoefPriceNonPeak=coef(fitNonPeak)[2]




plot(xaxis,demandNonPeak,pch = 16, type="s",col="blue", las=1, xaxt="n",
     xlab="Price for Peak Period (Time slot 5-9pm)",ylab="Non-Peak Period Demand")
xticks <- seq(0, maxprice, by=50)
axis(side = 1, at = xticks)
abline(lm(demandNonPeak ~ PeakPrice), lwd=2, col="darkred")
```




```r
#summary(fitNonPeak)



# Fitting a Linear Demand Model for Peak Period Demand
fitPeak <-lm(demandPeak ~ PeakPrice)
InterceptPeak=coef(fitPeak)[1]
CoefPricePeak=coef(fitPeak)[2]



plot(xaxis,demandPeak,pch = 16, type="s",col="blue", las=1, xaxt="n",
     xlab="Price for Peak Period (Time slot 5-9pm)",ylab="Peak Period Demand")
xticks <- seq(0, maxprice, by=50)
axis(side = 1, at = xticks)
abline(lm(demandPeak ~ PeakPrice), lwd=2, col="darkred")
```

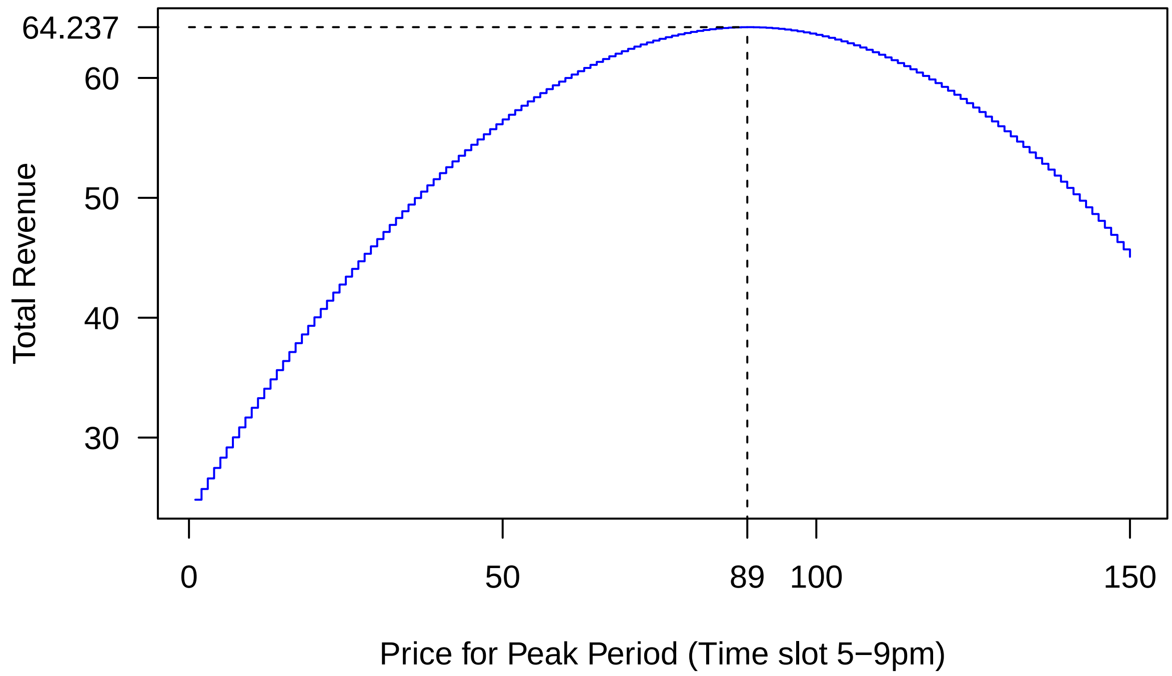



```r
#summary(fitNonPeak)

# To display the regression results in a Latex table that can be easily included in a report w
library(stargazer)
stargazer(fitNonPeak,fitPeak, type="text")
```

```
##
## ============================================================
##                                   Dependent variable:
##                               ----------------------------
##                               demandNonPeak   demandPeak
##                                    (1)            (2)
## ------------------------------------------------------------
## PeakPrice                        4.381***      -5.114***
##                                  (0.115)        (0.127)
##
## Constant                        367.910***    623.440***
##                                 (10.008)       (11.060)
##
## ------------------------------------------------------------
## Observations                       150            150
## R2                                0.908          0.916
## Adjusted R2                       0.907          0.916
## Residual Std. Error (df = 148)   60.978         67.387
```

```
## F Statistic (df = 1; 148)       1,452.017***  1,619.896***
## ============================================================
## Note:                           *p<0.1; **p<0.05; ***p<0.01
```

```r
# stargazer(fitNonPeak,fitPeak, type="latex", out="Outputexample.tex")


# Maximizing Revenue From the Linear Demand Model
demandNonPeakLinear<-rep(0,maxprice)
demandPeakLinear<-rep(0,maxprice)
revenueLinear<-rep(0,maxprice)

for (p in 1:maxprice){
  demandNonPeakLinear[p]=InterceptNonPeak+CoefPriceNonPeak*p
  demandPeakLinear[p]=InterceptPeak+CoefPricePeak*p
  revenueLinear[p]=basePrice*demandNonPeakLinear[p]+p*demandPeakLinear[p]
}

revenueLinearBest=max(revenueLinear)
priceBestLinear=which(revenueLinear == revenueLinearBest)


# Plotting Revenue vs Peak Period Price
xaxis=1:maxprice
plot(xaxis,revenueLinear/1000,pch = 16, type="s",col="blue", las=1, xaxt="n",
     xlab="Price for Peak Period (Time slot 5-9pm)",ylab="Total Revenue")
xticks <- seq(0, maxprice, by=50)
axis(side = 1, at = xticks)

axis(side = 1, at = priceBestLinear)
lines(c(priceBestLinear,priceBestLinear),c(0, revenueLinearBest/1000),lty=2)
axis(side = 2, at = round(revenueLinearBest/1000,3),las=1)
lines(c(0,priceBestLinear),c(revenueLinearBest/1000, revenueLinearBest/1000),lty=2)
```

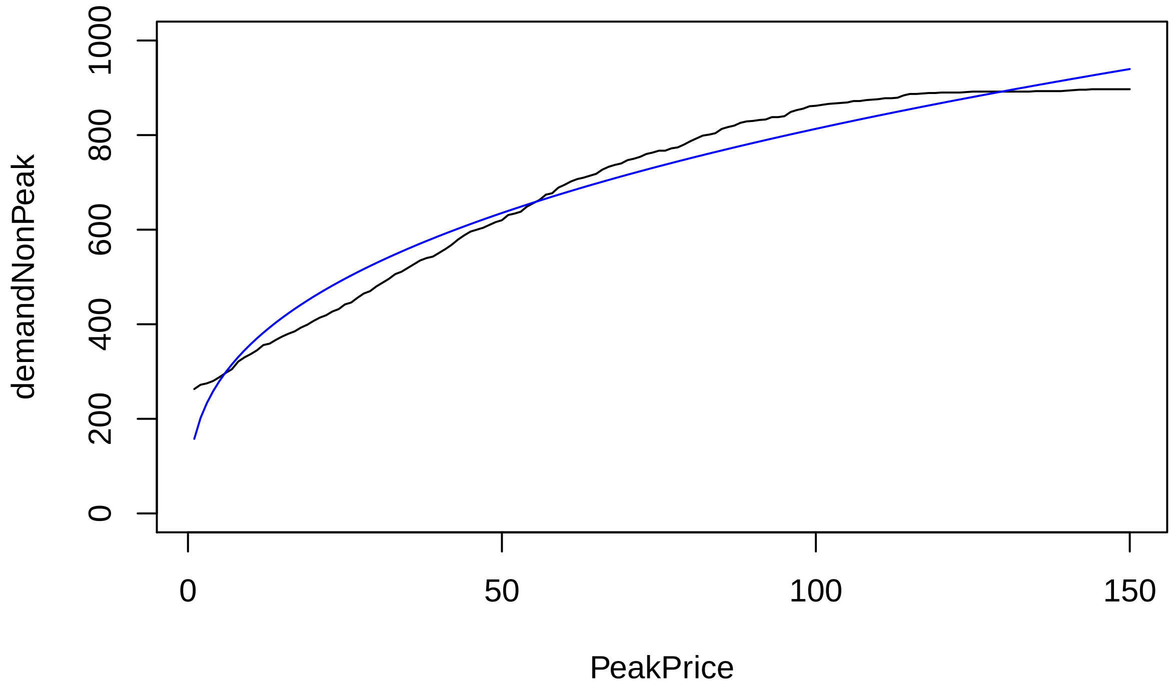



```
print(paste("When other periods have a base price of 65, the optimal price for the 5-9pm slot
```

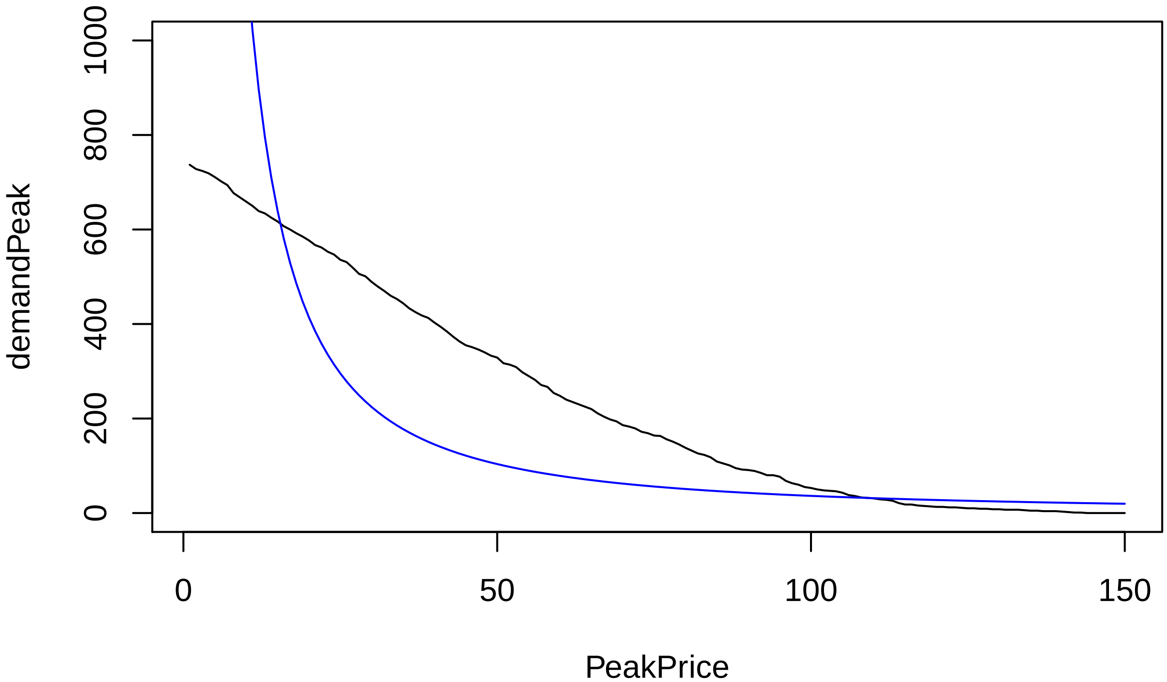
```
## [1] "When other periods have a base price of 65, the optimal price for the 5-9pm slot (usin
```

2.5 Aside: Fitting Constant Elasticity and Log Linear Demand Functions

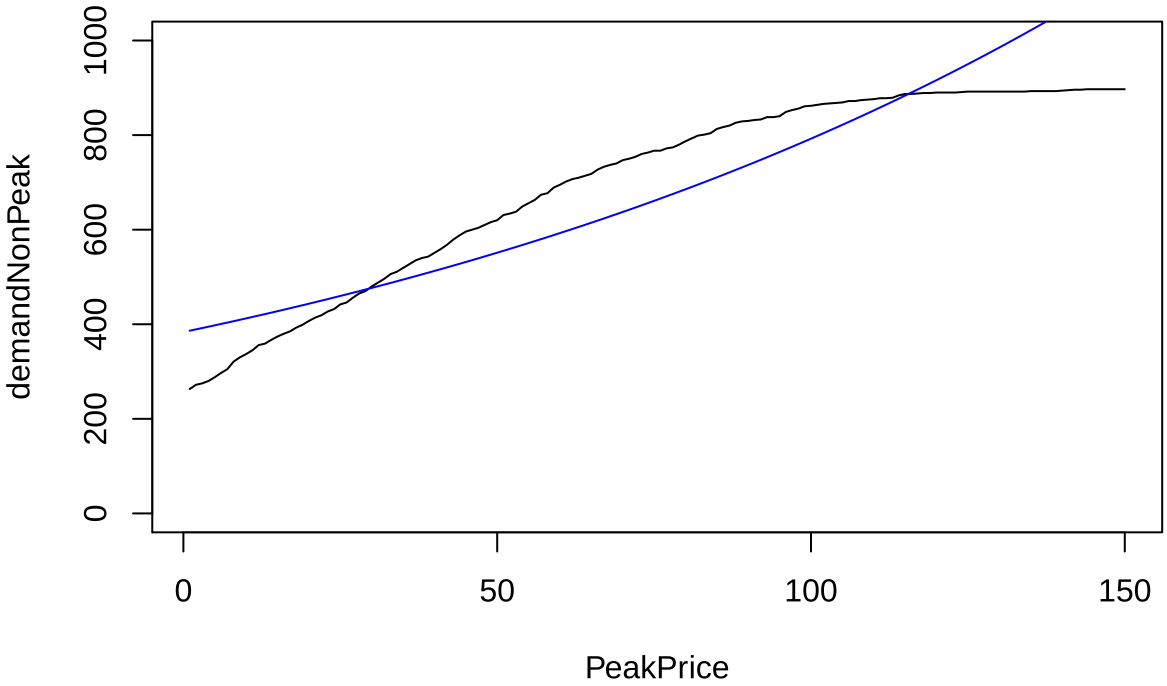
```
# Fitting a Constant Elasticity Demand Model
PeakPrice=1:maxprice
# for Non-Peak Period Demand
logdemandNonPeak=log(demandNonPeak)
logdemandNonPeak=ifelse(demandNonPeak > 0, log(demandNonPeak), 0)
logPeakPrice=log(PeakPrice)
fitIsoNonPeak <-lm(logdemandNonPeak ~ logPeakPrice)
#summary(fitIsoPeak)
InterceptIsoNonPeak=coef(fitIsoNonPeak)[1]
CoefPriceIsoNonPeak=coef(fitIsoNonPeak)[2]
D=exp(InterceptIsoNonPeak)
plot(PeakPrice,demandNonPeak,type="l",ylim=c(0, N))
lines(PeakPrice,D*PeakPrice^(CoefPriceIsoNonPeak), col="blue")
```



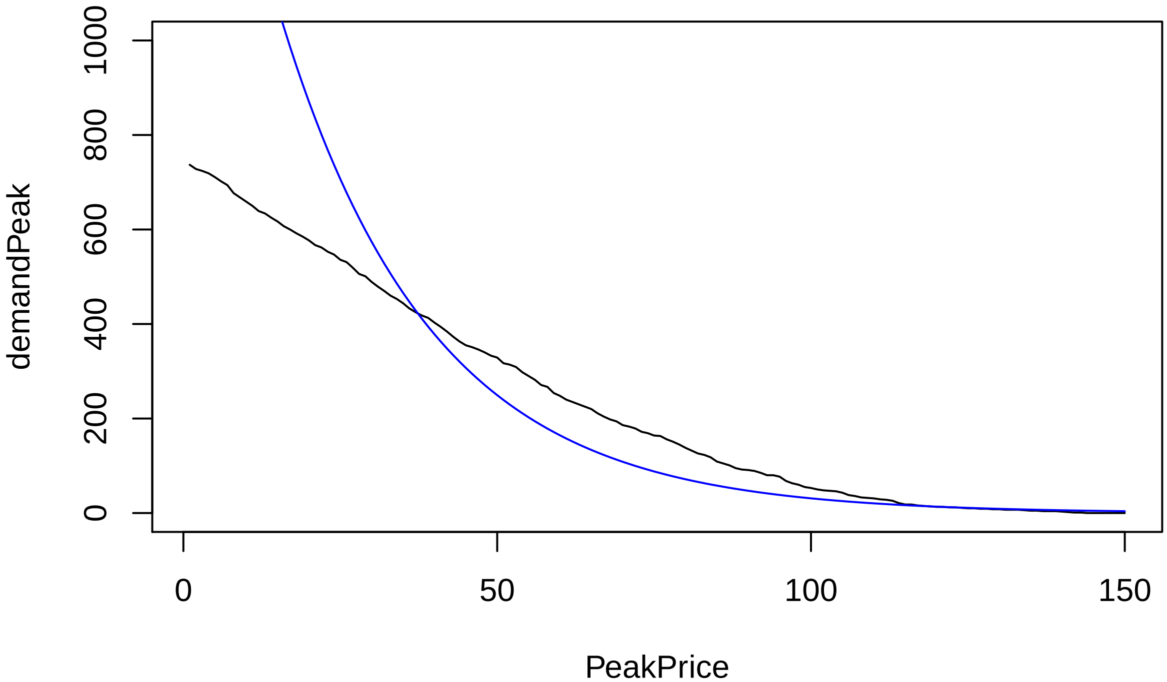

```
# for Peak Period Demand
logdemandPeak=log(demandPeak)
logdemandPeak=ifelse(demandPeak > 0, log(demandPeak), 0)
logPeakPrice=log(PeakPrice)
fitIsoPeak <-lm(logdemandPeak ~ logPeakPrice)
#summary(fitIsoPeak)
InterceptIsoPeak=coef(fitIsoPeak)[1]
CoefPriceIsoPeak=coef(fitIsoPeak)[2]
D=exp(InterceptIsoPeak)
plot(PeakPrice,demandPeak,type="l",ylim=c(0, N))
lines(PeakPrice,D*PeakPrice^(CoefPriceIsoPeak),col="blue")
```




```
# Fitting a Log Linear (Exponential) Demand Model
PeakPrice=1:maxprice
# for Non-Peak Period Demand
logdemandNonPeak=log(demandNonPeak)
logdemandNonPeak=ifelse(demandNonPeak > 0, log(demandNonPeak), 0)
fitExpNonPeak <-lm(logdemandNonPeak ~ PeakPrice)
#summary(fitExpPeak)
InterceptExpNonPeak=coef(fitExpNonPeak)[1]
CoefPriceExpNonPeak=coef(fitExpNonPeak)[2]
D=exp(InterceptExpNonPeak)
plot(PeakPrice,demandNonPeak,type="l",ylim=c(0, N))
lines(PeakPrice,D*exp(CoefPriceExpNonPeak*PeakPrice),col="blue")
```



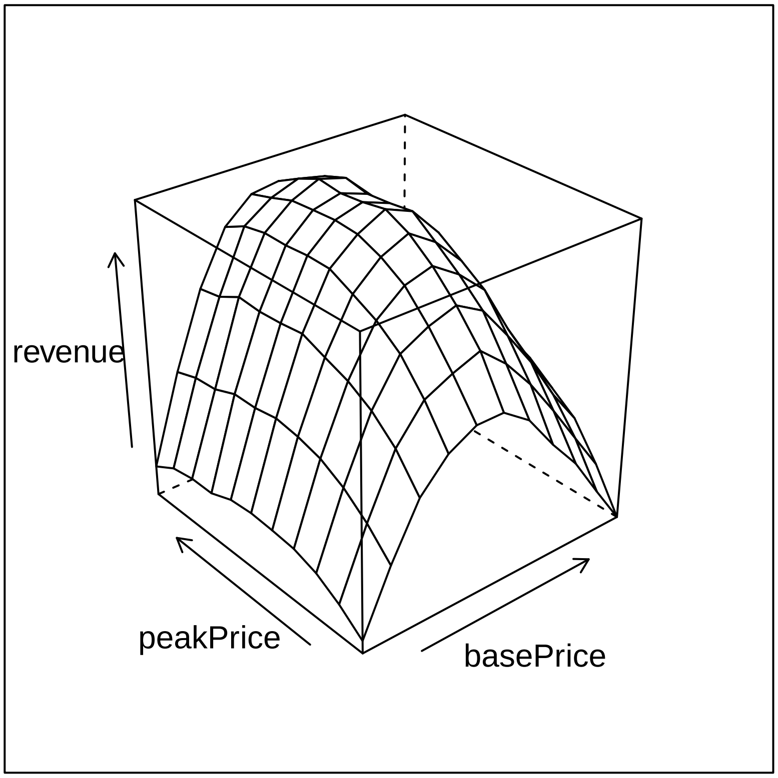

```r
# for Peak Period Demand
logdemandPeak=log(demandPeak)
logdemandPeak=ifelse(demandPeak > 0, log(demandPeak), 0)
fitExpPeak <-lm(logdemandPeak ~ PeakPrice)
#summary(fitExpPeak)
InterceptExpPeak=coef(fitExpPeak)[1]
CoefPriceExpPeak=coef(fitExpPeak)[2]
D=exp(InterceptExpPeak)
plot(PeakPrice,demandPeak,type="l",ylim=c(0, N))
lines(PeakPrice,D*exp(CoefPriceExpPeak*PeakPrice),col="blue")
```



2.6 Setting NonPeak and Peak Period Prices Simultaneously

```
# PART A3. Setting Both NonPeak and PeakPrices

# Read the Survey Data
nyhc = read.csv("NYHCSurvey.csv",header=T)

# Row count
N=nrow(nyhc)

# We would like to create new data that has four columns,
# the first two columns will be the prices for Nonpeak and Peak time slots
# The remaining columns will count how many people buy NonPeak and Peak at that price combinat
# Lets search prices from 25 to 125 with an increment of 5.
# This gives us 21 price points for basePrice (i.e., 25, 30, ..., 125) and similarly
# it will lead to 21 price points for Peak price (i.e., 25, 30, ..., 125).
# We will not require that Peak Price exceeds basePrice now because our goal at this stage
# is to model the demand in terms of prices.
# The total number of price combinations is 21*21=441
# We will keep track of this through an index variable called index.
```

```r
# Price for all time slots (i.e., nonpeak) except for time slot 5 (5-9pm) (peak)
surplusNonPeak<-rep(0,N)
surplusPeak<-rep(0,N)
demandNonPeak<-rep(0,121)
demandPeak<-rep(0,121)

index=1
for (basePrice in seq(from = 50, to = 100, by = 5)){
  for (peakPrice in seq(from = 50, to = 100, by = 5)){
    for (i in 1:N){
      surplusNonPeak[i]=max(nyhc[i,c(3:6,8)]-basePrice)
      surplusPeak[i]=nyhc[i,7]-peakPrice
    }
    demandNonPeak[index]=sum((surplusNonPeak>surplusPeak)*(surplusNonPeak>=0))
    demandPeak[index]=sum((surplusPeak>=surplusNonPeak)*(surplusPeak>=0))
    index=index+1
  }
}

# Create a data table which we will use to run the two regressions:
newdata<-data.frame(matrix(nrow=121,ncol = 5))
colnames(newdata)=c("index","basePrice","peakPrice","NonPeakDemand", "PeakDemand")
index=1
for (basePrice in seq(from = 50, to = 100, by = 5)){
  for (peakPrice in seq(from = 50, to = 100, by = 5)){
    newdata[index,1]=index
    newdata[index,2]=basePrice
    newdata[index,3]=peakPrice
    newdata[index,4]=demandNonPeak[index]
    newdata[index,5]=demandPeak[index]
    index=index+1
  }
}

# Visualizing Revenue as a Function of Base and Peak Price
newdata$revenue=newdata$basePrice*newdata$NonPeakDemand+newdata$peakPrice*newdata$PeakDemand

library(lattice)
wireframe(revenue ~ basePrice * peakPrice, data=newdata)
```




```r
# Run Regressions:

# Regression for the dependent variable NonPeakDemand

fit2NonPeak <-lm(NonPeakDemand ~ basePrice+peakPrice, data=newdata)
summary(fit2NonPeak)
```

```
##
## Call:
## lm(formula = NonPeakDemand ~ basePrice + peakPrice, data = newdata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -53.008 -13.117   2.064  17.293  35.683
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1202.8430    12.5167   96.10   <2e-16 ***
## basePrice    -11.8540     0.1167 -101.56   <2e-16 ***
## peakPrice      4.0587     0.1167   34.77   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.3 on 118 degrees of freedom
## Multiple R-squared:  0.9899, Adjusted R-squared:  0.9897
## F-statistic:  5762 on 2 and 118 DF,  p-value: < 2.2e-16
```

```r
a1=coef(fit2NonPeak)[1]
b11=coef(fit2NonPeak)[2]
b12=coef(fit2NonPeak)[3]

# Regression for the dependent variable NonPeakDemand

fit2Peak <-lm(PeakDemand ~ basePrice+peakPrice, data=newdata)
a2=coef(fit2Peak)[1]
b21=coef(fit2Peak)[2]
b22=coef(fit2Peak)[3]

stargazer(fit2NonPeak,fit2Peak, type="text")
```

```
##
## ===========================================================
##                                   Dependent variable:
##                               ----------------------------
##                               NonPeakDemand   PeakDemand
##                                    (1)            (2)
## -----------------------------------------------------------
## basePrice                      -11.854***     4.168***
##                                 (0.117)        (0.119)
##
## peakPrice                       4.059***      -5.949***
##                                 (0.117)        (0.119)
##
## Constant                      1,202.843***   348.190***
##                                 (12.517)      (12.771)
##
## -----------------------------------------------------------
## Observations                      121            121
## R2                               0.990          0.969
## Adjusted R2                      0.990          0.969
## Residual Std. Error (df = 118)   20.300         20.712
```

```
## F Statistic (df = 2; 118)       5,761.901***  1,860.117***
## =============================================================
## Note:                            *p<0.1; **p<0.05; ***p<0.01
```

```r
#stargazer(fit2NonPeak,fit2Peak, type="latex", out="Outputexample.tex")


# Finding optimal revenue by optimization
library("nloptr")

# Differentiated Prices

eval_f <- function(x){
  basePrice=x[1]
  peakPrice=x[2]
  NonPeakDemand=max(0,a1+b11*basePrice+b12*peakPrice)
  PeakDemand=max(0,a2+b21*basePrice+b22*peakPrice)
  revenue=basePrice*NonPeakDemand+peakPrice*PeakDemand
  objfunction=-revenue
  return(objfunction)
}

eval_g_ineq <- function(x) {
  basePrice=x[1]
  peakPrice=x[2]
  NonPeakDemand=max(0,a1+b11*basePrice+b12*peakPrice)
  PeakDemand=max(0,a2+b21*basePrice+b22*peakPrice)
  constraint <- c(-NonPeakDemand,
                  -PeakDemand,
                  x[1]-x[2])
  return(constraint)
}

# initial values
x0 <- c(70,90)
# lower and upper bounds of control
lb <- c(50,50)
ub <- c(100,100)
opts <- list( "algorithm" = "NLOPT_LN_COBYLA",
              "xtol_rel"  = 1.0e-9,
```

```r
                 "maxeval"    = 1000)
result <- nloptr(x0=x0,eval_f=eval_f,lb=lb,ub=ub,
                     eval_g_ineq=eval_g_ineq,opts=opts)
# print(result)

priceOpt<-result$solution
RevenueOpt<- -result$objective

print(paste("Optimal Base Price:",priceOpt[1]))
```

```
## [1] "Optimal Base Price: 80.1121177652542"
```

```r
print(paste("Optimal Peak Price:",priceOpt[2]))
```

```
## [1] "Optimal Peak Price: 84.6597709195169"
```

```r
print(paste("Optimal Revenue:",RevenueOpt))
```

```
## [1] "Optimal Revenue: 62919.9944037216"
```