Data pre-processing

```
library(nloptr)
library(stargazer)
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2022). stargazer: Well-Formatted Regression and Summary Statistics Tables.
```

```
##  R package version 5.2.3. https://CRAN.R-project.org/package=stargazer
```

```
library(lattice)
library(ggplot2)
```

```
# Set working directory

setwd('~/Library/CloudStorage/OneDrive-City,UniversityofLondon/term 2/SMM641 Revenue Management and Pri

# Import the data

data=read.csv('CongestionPricing.csv')

# Inspect data structures

head(data)
```

```
##   resp_id Peak_WTP Nonpeak_WTP
## 1       1        7           7
## 2       2        9           9
## 3       3        7           7
## 4       4        9           2
## 5       5        9           5
## 6       6       15           8
```

```
str(data)
```

```
## 'data.frame':    345 obs. of  3 variables:
##  $ resp_id    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Peak_WTP   : num  7 9 7 9 9 15 7 15 7 15 ...
##  $ Nonpeak_WTP: num  7 9 7 2 5 8 9 6 5 7 ...
```

```
# Get the number of data observations

N=nrow(data)
```

Scenario 1: Sinple Price Optimisation

If the programme's objective were solely to maximise revenue and a single congestion charge were to be applied across both peak and non-peak hours, which price would maximize the total revenue? With this price in effect, what is the total level of emissions?

To fine the optimal single price, we set a single price across all time slots and all car types to find the maximised revenue and compute the according emissions.

```
# Get the maximum WTP across 2 time slots

for (i in 1:N){data$maxWTP[i]=max(data[i,2:3])}

# Check the new column

head(data)
```

```
##   resp_id Peak_WTP Nonpeak_WTP maxWTP
## 1       1        7           7      7
## 2       2        9           9      9
## 3       3        7           7      7
## 4       4        9           2      9
## 5       5        9           5      9
## 6       6       15           8     15
```

```
# The maximum WTP in data, we can use this as the upper bound for our price search
# No need to consider a price if no one can afford it.

maxprice=max(data$maxWTP)

# Define empty array variables we will be introducing

demandSinglePrice = rep(NA,maxprice)
revenue           = rep(NA,maxprice)
averageSpeed      = rep(NA,maxprice)
emissionsPerCar   = rep(NA,maxprice)
totalEmissions    = rep(NA,maxprice)
demandLondon      = rep(NA,maxprice)
price             = c(1:maxprice)

# Find how many people willing to buy at different price levels

for (p in 1:maxprice){
    # demand at each price level
    demandSinglePrice[p] = sum(data$maxWTP>=p) # total number of TRUE value
    # convert demand to represent the population level (in thousands)
    demandLondon[p] = demandSinglePrice[p]*192/N
    # total revenue of congestion charge in London
    revenue[p]=p*demandLondon[p]*1000
}

task1a = data.frame(price, demandSinglePrice, demandLondon,
                    averageSpeed, emissionsPerCar, totalEmissions, revenue)
```

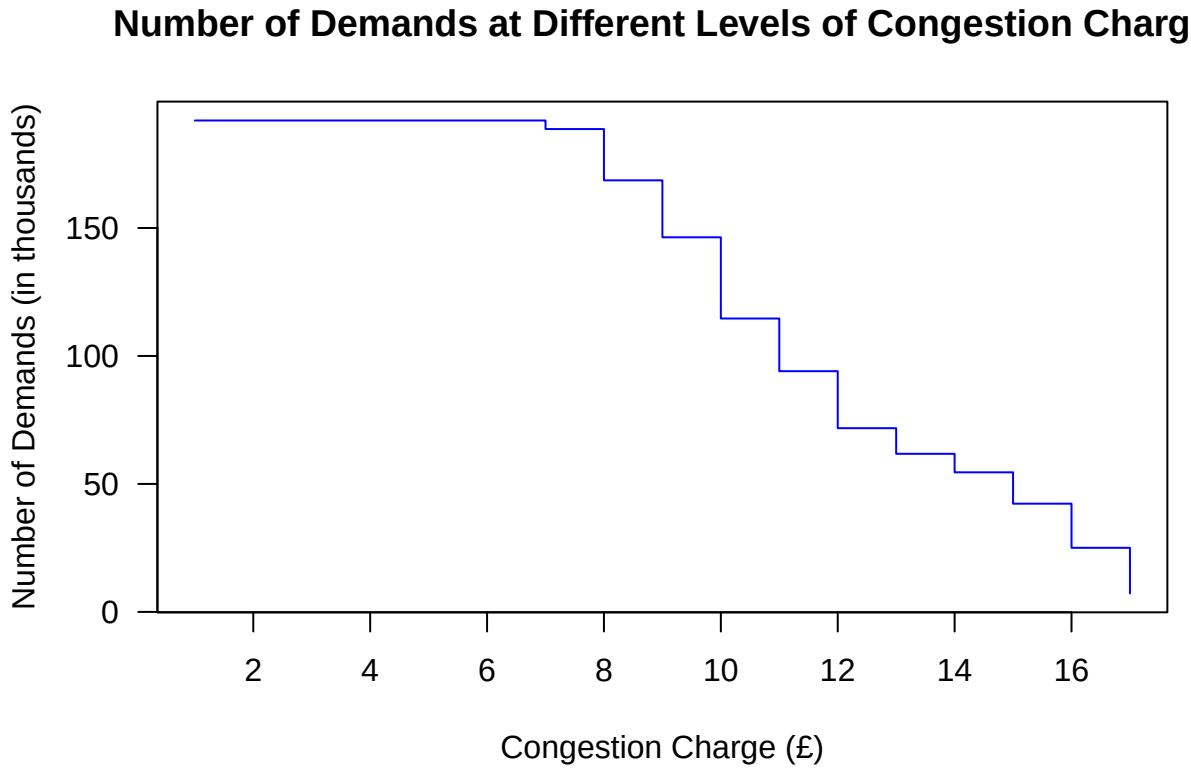
```r
# Identify values to maximise Revenue

revenueBest = max(revenue)
priceBest = which(revenue == revenueBest)
emissionsPerCarBest = emissionsPerCar[priceBest]
totalEmissionsBest = totalEmissions[priceBest]

# Print result

print(paste("If a single congestion charge were to be charged across both peak and non-peak hours, the
```

```
## [1] "If a single congestion charge were to be charged across both peak and non-peak hours, the optima
```

```r
# Set empty array

demandNonPeak_single = rep(NA,N)
demandPeak_single = rep(NA,N)

# Compute the consumer surplus and classify which hours driver will enter London

for (r in 1:N){
  surplusNonPeak_single = data[r,3] - priceBest
  surplusPeak_single = data[r,2] - priceBest
  demandNonPeak_single[r] = (surplusNonPeak_single>surplusPeak_single)*(surplusNonPeak_single>=0)
  demandPeak_single[r] = (surplusPeak_single>=surplusNonPeak_single)*(surplusPeak_single>=0)
}

# Demand of total population (in thousands)

demandNonPeak_single_total = sum(demandNonPeak_single) * (192/N)
demandPeak_single_total = sum(demandPeak_single) * (192/N)

# Compute average speed and emissions level at 2 periods

avg_speed_NonPeak = 30 - 0.0625*demandNonPeak_single_total
emissions_NonPeak = (ifelse(avg_speed_NonPeak<25,
                            617.5-16.7*avg_speed_NonPeak,
                            235.0-1.4*avg_speed_NonPeak))*demandNonPeak_single_total

avg_speed_Peak = 30 - 0.0625*demandPeak_single_total
emissions_Peak = (ifelse(avg_speed_Peak<25,
                            617.5-16.7*avg_speed_Peak,
                            235.0-1.4*avg_speed_Peak))*demandPeak_single_total

# Level of emssions all day

emissionsLevel = emissions_NonPeak + emissions_Peak

# Print result

print(paste("At", priceBest,", the total level of emissions is",
            round(emissionsLevel),"(g/km)"))
```

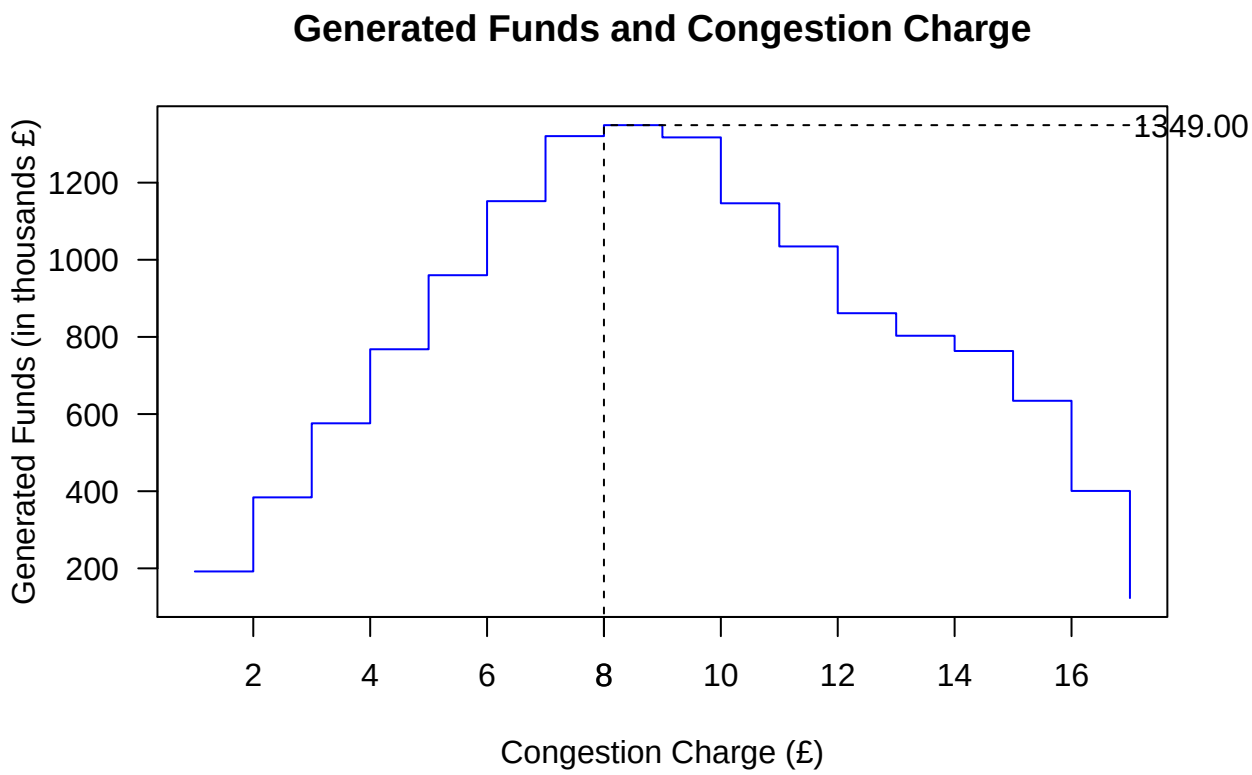
```
## [1] "At 8 , the total level of emissions is 45846 (g/km)"
```

```
# Plotting Number of cars vs Price

xaxis=1:maxprice

plot(xaxis, demandLondon,
     pch = 16, type="s", col="blue", las=1, xaxt="n",
     xlab="Congestion Charge (£)",ylab="Number of Demands (in thousands)",
     main = "Number of Demands at Different Levels of Congestion Charge")

xticks <- seq(0, maxprice, by=2)

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="Congestion Charge (£)",ylab="Generated Funds (in thousands £)",
     main = "Generated Funds and Congestion Charge")

xticks <- seq(0, maxprice, by=2)
```

```
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)

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



Scenario 2: Find Optimal Peak Price at Base Price Level £7

If the programme's objective were solely to maximise revenue and a peak period pricing strategy is to be implemented, with the price for the non-peak period set at £7, what price would we recommend for the peak period? Note the resulting revenue and emissions and compare the findings with those from scenario 1.

```
basePrice = 7

data$surplusNonPeak = data$Nonpeak_WTP-basePrice

data[1:10,]
```

```
##    resp_id Peak_WTP Nonpeak_WTP maxWTP surplusNonPeak
```

```
## 1        1        7           7      7             0
## 2        2        9           9      9             2
## 3        3        7           7      7             0
## 4        4        9           2      9            -5
## 5        5        9           5      9            -2
## 6        6       15           8     15             1
## 7        7        7           9      9             2
## 8        8       15           6     15            -1
## 9        9        7           5      7            -2
## 10      10       15           7     15             0
```

```
# Create a matrix of N rows, max price columns, fill with all 0 value
surplusPeak = matrix(0,N,maxprice)

# Compute Maximum Peak Surplus

for (p in 1:maxprice){
    for (i in 1:N){
        surplusPeak[i,p]=data[i,2]-p
    }
}

# Show results

colnames(surplusPeak)=paste0("p=",1:maxprice)

surplusPeak[1:10,]
```

```
##       p=1 p=2 p=3 p=4 p=5 p=6 p=7 p=8 p=9 p=10 p=11 p=12 p=13 p=14 p=15 p=16
##  [1,]   6   5   4   3   2   1   0  -1  -2   -3   -4   -5   -6   -7   -8   -9
##  [2,]   8   7   6   5   4   3   2   1   0   -1   -2   -3   -4   -5   -6   -7
##  [3,]   6   5   4   3   2   1   0  -1  -2   -3   -4   -5   -6   -7   -8   -9
##  [4,]   8   7   6   5   4   3   2   1   0   -1   -2   -3   -4   -5   -6   -7
##  [5,]   8   7   6   5   4   3   2   1   0   -1   -2   -3   -4   -5   -6   -7
##  [6,]  14  13  12  11  10   9   8   7   6    5    4    3    2    1    0   -1
##  [7,]   6   5   4   3   2   1   0  -1  -2   -3   -4   -5   -6   -7   -8   -9
##  [8,]  14  13  12  11  10   9   8   7   6    5    4    3    2    1    0   -1
##  [9,]   6   5   4   3   2   1   0  -1  -2   -3   -4   -5   -6   -7   -8   -9
## [10,]  14  13  12  11  10   9   8   7   6    5    4    3    2    1    0   -1
##       p=17
##  [1,]  -10
##  [2,]   -8
##  [3,]  -10
##  [4,]   -8
##  [5,]   -8
##  [6,]   -2
##  [7,]  -10
##  [8,]   -2
##  [9,]  -10
## [10,]   -2
```

```
# Create empty arrays
```

```r
demandNonPeak<-rep(0,maxprice)

demandPeak<-rep(0,maxprice)

revenue2<-rep(0,maxprice)

emissions1b<-rep(0,maxprice)

demand1b<-rep(0,maxprice)

# Compare consumer surplus and classify customer segments into peak and non-peak hours

for (p in 1:maxprice){

  surplusNonPeak = data$surplusNonPeak

  demandNonPeak[p]=sum((surplusNonPeak>surplusPeak[,p])*(surplusNonPeak>=0)) * (192/N)

  demandPeak[p]=sum((surplusPeak[,p]>=surplusNonPeak)*(surplusPeak[,p]>=0)) * (192/N)

  revenue2[p]=basePrice*demandNonPeak[p]*1000+p*demandPeak[p]*1000

  # Find level of emissions at each price point

  ## Non Peak hours

  avg_speed_NonPeak = 30 - 0.0625*demandNonPeak[p]

  emissions_NonPeak = (ifelse(avg_speed_NonPeak<25,
                              617.5-16.7*avg_speed_NonPeak,
                              235.0-1.4*avg_speed_NonPeak))*demandNonPeak[p]
  ## Peak hours

  avg_speed_Peak = 30 - 0.0625*demandPeak[p]

  emissions_Peak = (ifelse(avg_speed_Peak<25,
                              617.5-16.7*avg_speed_Peak,
                              235.0-1.4*avg_speed_Peak))*demandPeak[p]

  # Level of emissions all day

  emissions1b[p] = emissions_NonPeak + emissions_Peak

  emissions1b[p]
}

task1b = data.frame(peakPrice=c(1:maxprice),
                    demandNonPeak, demandPeak,
                    revenue2, emissions1b)
task1b
```

```
##    peakPrice demandNonPeak demandPeak  revenue2 emissions1b
## 1          1     0.0000000 192.000000  192000.0    60844.80
```

```
## 2          2     0.0000000 192.000000  384000.0    60844.80
## 3          3     0.0000000 192.000000  576000.0    60844.80
## 4          4     0.5565217 191.443478  769669.6    60664.67
## 5          5     1.1130435 190.886957  962226.1    60485.24
## 6          6     6.6782609 185.321739 1158678.3    58729.49
## 7          7    13.3565217 175.304348 1320626.1    55092.50
## 8          8    29.4956522 145.252174 1368487.0    44711.90
## 9          9    42.8521739 119.095652 1371826.1    37110.10
## 10        10    50.0869565  94.608696 1296695.7    30250.61
## 11        11    58.4347826  78.469565 1272208.7    27260.10
## 12        12    69.0086957  62.886957 1237704.3    26218.60
## 13        13    78.4695652  48.417391 1178713.0    25233.08
## 14        14    89.6000000  33.947826 1102469.6    25470.56
## 15        15    95.1652174  22.260870 1000069.6    24879.09
## 16        16   101.2869565  11.130435  887095.7    24666.83
## 17        17   106.2956522   2.226087  781913.0    24606.60
```

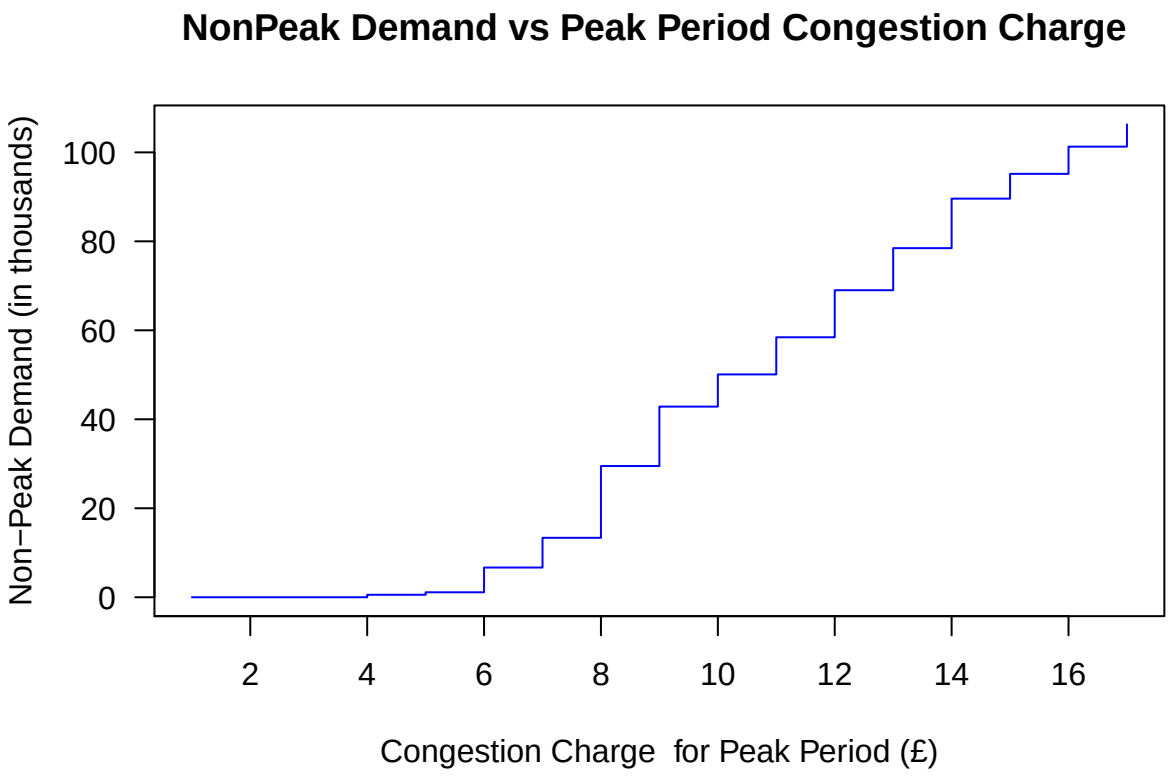
```
# Identify values to maximise Revenue

revenueBest2 = max(revenue2)

priceBest2 = which(revenue2 == revenueBest2)

emissionsBest2 = emissions1b[priceBest2]

# Print result

print(paste("When Non-peak periods have a base price of 7 (£), the optimal revenue is", round(revenueBes
```

```
## [1] "When Non-peak periods have a base price of 7 (£), the optimal revenue is 1371826 (£) at the opti
```

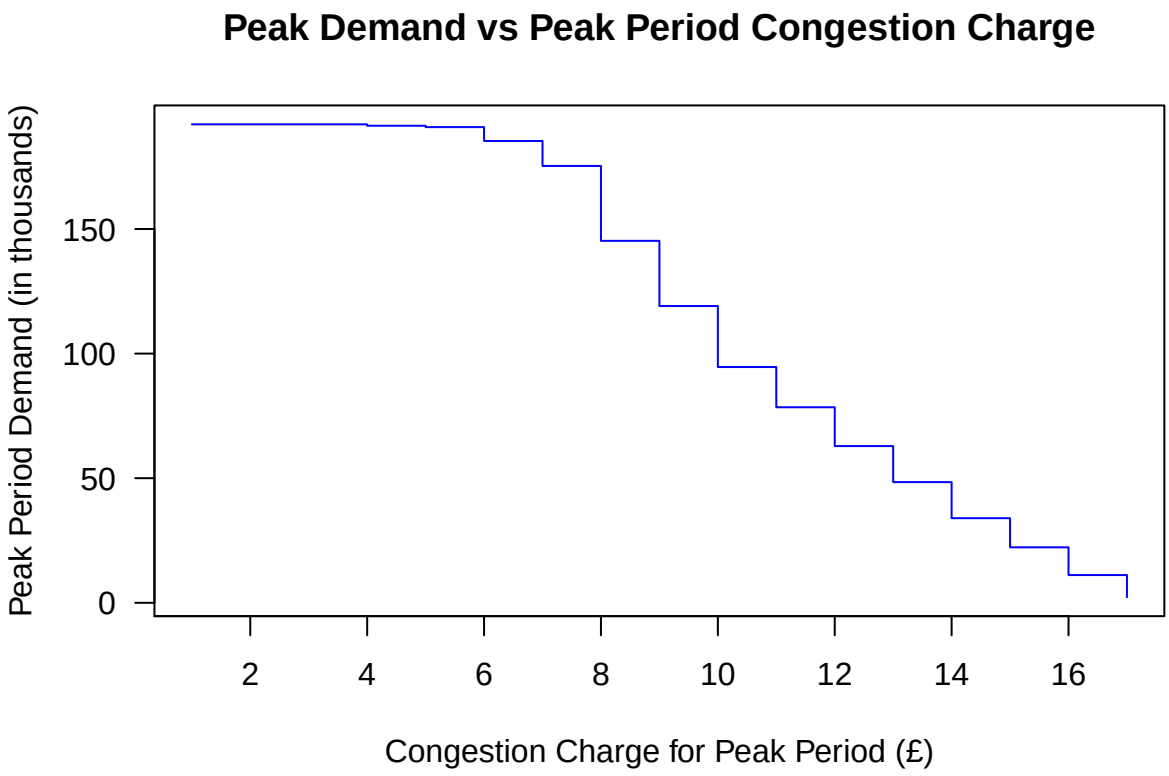
```
print(paste("Level of emssisions for this price point is", round(emissionsBest2), "(g/km)"))
```

```
## [1] "Level of emssisions for this price point is 37110 (g/km)"
```

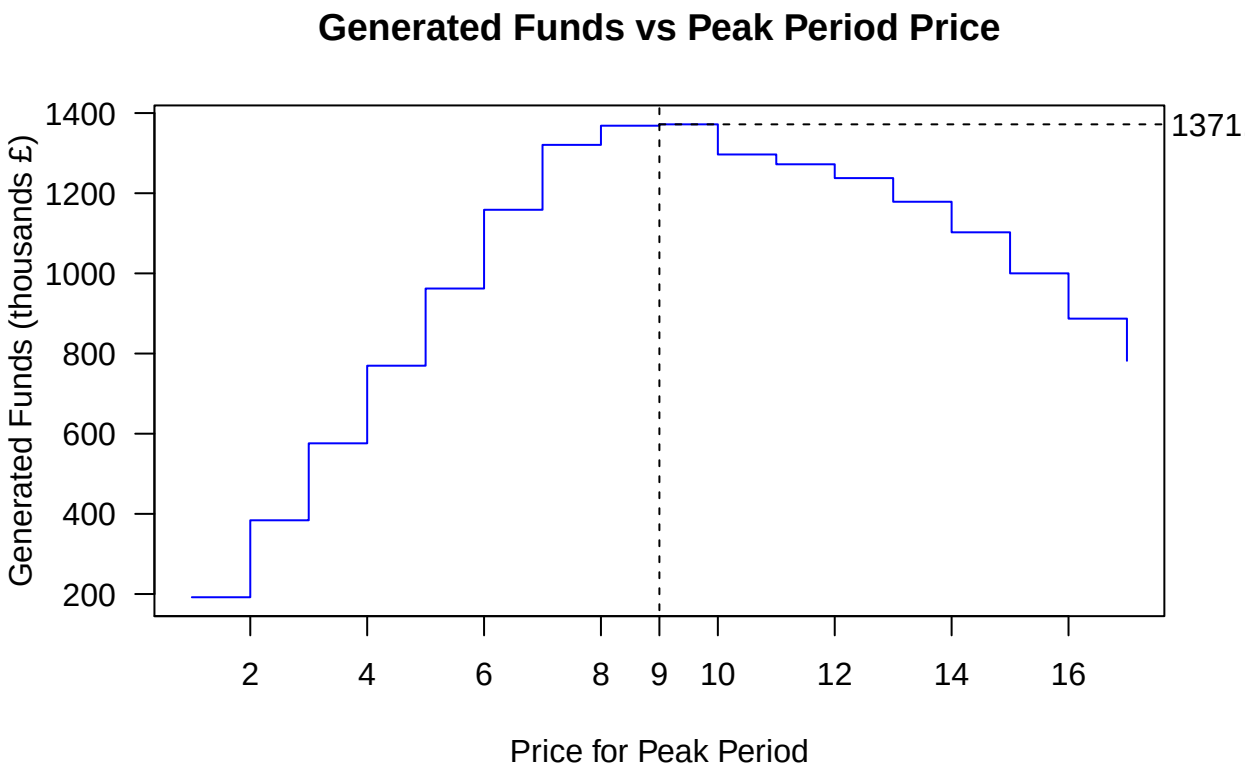
```
# Plotting NonPeak Demand vs Peak Period Price

xaxis=1:maxprice

plot(xaxis,task1b$demandNonPeak,pch = 16, type="s",col="blue", las=1, xaxt="n",
     xlab="Congestion Charge  for Peak Period (£)",ylab="Non-Peak Demand (in thousands)",
     main = "NonPeak Demand vs Peak Period Congestion Charge ")

xticks <- seq(0, maxprice, by=2)

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




```
# Plotting Peak Demand vs Peak Period Price

xaxis=1:maxprice

plot(xaxis,task1b$demandPeak,pch = 16, type="s",col="blue", las=1, xaxt="n",
     xlab="Congestion Charge for Peak Period (£)",ylab="Peak Period Demand (in thousands)",
     main = "Peak Demand vs Peak Period Congestion Charge")

xticks <- seq(0, maxprice, by=2)

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




```r
# Plotting Revenue vs Peak Period Price

xaxis=1:maxprice

plot(xaxis,task1b$revenue2/1000,pch = 16, type="s",col="blue", las=1, xaxt="n",
     xlab="Price for Peak Period",ylab="Generated Funds (thousands £)",
     main = "Generated Funds vs Peak Period Price")

xticks <- seq(0, maxprice, by=2)

axis(side = 1, at = xticks)

revenueBest2 = max(revenue2[basePrice:maxprice])

priceBest2 = which(revenue2/1000 == revenueBest2/1000)

axis(side = 1, at = priceBest2)

lines(c(priceBest2,priceBest2),c(0, revenueBest2),lty=2)

axis(side = 4, at = round(revenueBest2/1000,3),las=1, pos=17.1, tick=F)

lines(c(priceBest2,20),c(revenueBest2/1000, revenueBest2/1000),lty=2)
```



Scenrio 3: Optimisation

Suppose now that the programme's objective is to minimize emissions rather than maximize revenue. However, the City would like to ensure that the programme can self-sustain its operation and that a sufficient portion of the revenue is allocated to reinvest in the public transportation infrastructure. Overall, the City requires that the revenue should not fall below £1.1 million per day.

Assuming a non-peak period price of £7, what price would we recommend for the peak period? Compare the resulting revenue and emissions level with that of part of Scenario 2.

```
# Fit linear model for multiple price

fitNonPeak = lm(demandNonPeak~peakPrice, data=task1b)

a1=coef(fitNonPeak)[1]

b1=coef(fitNonPeak)[2]

summary(fitNonPeak)
```

```
##
## Call:
## lm(formula = demandNonPeak ~ peakPrice, data = task1b)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.7124  -4.1098   0.2182   3.5956  18.7362
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -26.5371     4.4747   -5.93 2.76e-05 ***
## peakPrice     7.8009     0.4367   17.86 1.61e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.821 on 15 degrees of freedom
## Multiple R-squared:  0.9551, Adjusted R-squared:  0.9521
## F-statistic: 319.1 on 1 and 15 DF,  p-value: 1.61e-11
```

```
fitPeak = lm(demandPeak~peakPrice, data=task1b)

a2=coef(fitPeak)[1]

b2=coef(fitPeak)[2]

summary(fitPeak)
```

```
##
## Call:
## lm(formula = demandPeak ~ peakPrice, data = task1b)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -35.126  -8.630  -5.201   6.756  33.056
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 241.2726     9.0869   26.55 5.03e-14 ***
## peakPrice   -14.1463     0.8868  -15.95 8.12e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.91 on 15 degrees of freedom
## Multiple R-squared:  0.9443, Adjusted R-squared:  0.9406
## F-statistic: 254.5 on 1 and 15 DF,  p-value: 8.118e-11
```

The results show a great fit to the data points with significantly high R-squared.

```
demandNonPeak=rep(0,maxprice)

demandPeak=rep(0,maxprice)

demandTotal=rep(0,maxprice)

emissionsNonPeak=rep(0,maxprice)

emissionsPeak=rep(0,maxprice)
```

```
emissionsTotal=rep(0,maxprice)

revenue1c=rep(0,maxprice)

for (p in (1:maxprice)) {

  demandNonPeak[p]=a1+b1*p

  demandPeak[p]=a2+b2*p

  demandTotal[p]=demandNonPeak[p]+demandPeak[p] # in thousand

  averageSpeed_NonPeak=30-0.0624*demandNonPeak[p]

  emissionsNonPeak[p] = (ifelse(avg_speed_NonPeak<25,
                           617.5-16.7*avg_speed_NonPeak,
                           235.0-1.4*avg_speed_NonPeak))*demandNonPeak[p]

  averageSpeed_Peak=30-0.0624*demandPeak[p]

  emissionsPeak[p] = (ifelse(averageSpeed_Peak<25,
                           617.5-16.7*averageSpeed_Peak,
                           235.0-1.4*averageSpeed_Peak))*demandPeak[p]

  emissionsTotal[p]=emissionsNonPeak[p] + emissionsPeak[p]

  revenue1c[p]=7*demandNonPeak[p]+p*demandPeak[p]
}

task1c = data.frame(c(1:17), demandNonPeak, demandPeak, demandTotal,
                    emissionsNonPeak, emissionsPeak,
                    emissionsTotal, revenue1c)
task1c
```

```
##    c.1.17. demandNonPeak  demandPeak demandTotal emissionsNonPeak emissionsPeak
## 1        1    -18.736232 227.1263427    208.3901       -4261.4826    80217.3492
## 2        2    -10.935379 212.9800512    202.0447       -2487.2092    72081.4481
## 3        3     -3.134527 198.8337596    195.6992        -712.9359    64362.6240
## 4        4      4.666326 184.6874680    189.3538        1061.3375    57060.8770
## 5        5     12.467178 170.5411765    183.0084        2835.6109    50176.2070
## 6        6     20.268031 156.3948849    176.6629        4609.8843    43708.6140
## 7        7     28.068883 142.2485934    170.3175        6384.1577    37658.0980
## 8        8     35.869736 128.1023018    163.9720        8158.4310    32024.6591
## 9        9     43.670588 113.9560102    157.6266        9932.7044    26808.2972
## 10      10     51.471441  99.8097187    151.2812       11706.9778    22009.0123
## 11      11     59.272293  85.6634271    144.9357       13481.2512    17626.8044
## 12      12     67.073146  71.5171355    138.5903       15255.5245    14249.6274
## 13      13     74.873998  57.3708440    132.2448       17029.7979    11360.1108
## 14      14     82.674851  43.2245524    125.8994       18804.0713     8505.5587
## 15      15     90.475703  29.0782609    119.5540       20578.3447     5685.9712
## 16      16     98.276556  14.9319693    113.2085       22352.6181     2901.3482
## 17      17    106.077408   0.7856777    106.8631       24126.8914      151.6897
##    emissionsTotal  revenue1c
```

```
## 1        75955.87   95.97272
## 2        69594.24  349.41245
## 3        63649.69  574.55959
## 4        58122.21  771.41415
## 5        53011.82  939.97613
## 6        48318.50 1080.24552
## 7        44042.26 1192.22234
## 8        40183.09 1275.90656
## 9        36741.00 1331.29821
## 10       33715.99 1358.39727
## 11       31108.06 1357.20375
## 12       29505.15 1327.71765
## 13       28389.91 1269.93896
## 14       27309.63 1183.86769
## 15       26264.32 1069.50384
## 16       25253.97  926.84740
## 17       24278.58  755.89838
```

```
task1c=task1c[task1c$revenue1c>=1100,]
task1c
```

```
##    c.1.17. demandNonPeak demandPeak demandTotal emissionsNonPeak emissionsPeak
## 7        7      28.06888  142.24859    170.3175         6384.158     37658.098
## 8        8      35.86974  128.10230    163.9720         8158.431     32024.659
## 9        9      43.67059  113.95601    157.6266         9932.704     26808.297
## 10      10      51.47144   99.80972    151.2812        11706.978     22009.012
## 11      11      59.27229   85.66343    144.9357        13481.251     17626.804
## 12      12      67.07315   71.51714    138.5903        15255.525     14249.627
## 13      13      74.87400   57.37084    132.2448        17029.798     11360.111
## 14      14      82.67485   43.22455    125.8994        18804.071      8505.559
##    emissionsTotal revenue1c
## 7        44042.26  1192.222
## 8        40183.09  1275.907
## 9        36741.00  1331.298
## 10       33715.99  1358.397
## 11       31108.06  1357.204
## 12       29505.15  1327.718
## 13       28389.91  1269.939
## 14       27309.63  1183.868
```

```
# Identify the optimal result

emissionsBest1c = min(task1c$emissionsTotal)

priceBest1c = which(emissionsTotal == emissionsBest1c)

revenueBest1c = revenue1c[priceBest1c] # in thousand

print(paste("Minimum emissions level of (g/km)", round(emissionsBest1c), "is achieved at price (£)", rou
)
```

```
## [1] "Minimum emissions level of (g/km) 27310 is achieved at price (£) 14 with the generated refund o
```