DB Reading and Model

Joseph Keogh

10/14/2020

Get data from DB

Connect to the DB

```
# load credentials
username <- "jgk7uf@va-energy2"
hostname <- "va-energy2.postgres.database.azure.com"
password <- "0Tn5KBFbm2&6bG"
dbname <- "postgres"

# open credentials
db_driver <- dbDriver("PostgreSQL")
db <- dbConnect(db_driver, user=username, password=password, dbname=dbname, host=hostname)

# test connection if returns true the db is connected
print(dbExistsTable(db, "test_table"))
```

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

Grab data from DB

```
# drop existing data
response <- dbGetQuery(db, "SELECT * FROM test_table")
```

Understand Data

Get basic statistics on data

```
summary(response)
```

```
##     datetime                          date                 time
##  Min.   :2020-06-26 00:00:00   Min.   :2020-06-26   Length:2884
##  1st Qu.:2020-06-28 14:03:45   1st Qu.:2020-06-28   Class :character
##  Median :2020-07-01 05:12:30   Median :2020-07-01   Mode  :character
##  Mean   :2020-07-01 05:07:30   Mean   :2020-06-30
##  3rd Qu.:2020-07-03 19:31:15   3rd Qu.:2020-07-03
##  Max.   :2020-07-06 10:35:00   Max.   :2020-07-06
##     pnode_id         pnode_name         total_lmp_rt
##  Min.   :34964545   Length:2884        Min.   : -9.45
##  1st Qu.:34964545   Class :character   1st Qu.: 13.47
##  Median :34964545   Mode  :character   Median : 18.10
```

```
## Mean   :34964545                          Mean   : 20.94
## 3rd Qu.:34964545                          3rd Qu.: 22.48
## Max.   :34964545                          Max.   :389.66
```

The first data is back in June, through beginning of October

3000 Datum

```r
as.numeric(max(response$date) - min(response$date))
```

```
## [1] 10
```

datatypes

```r
typeof(response$datetime)
```

```
## [1] "double"
```

```r
typeof(response$date)
```

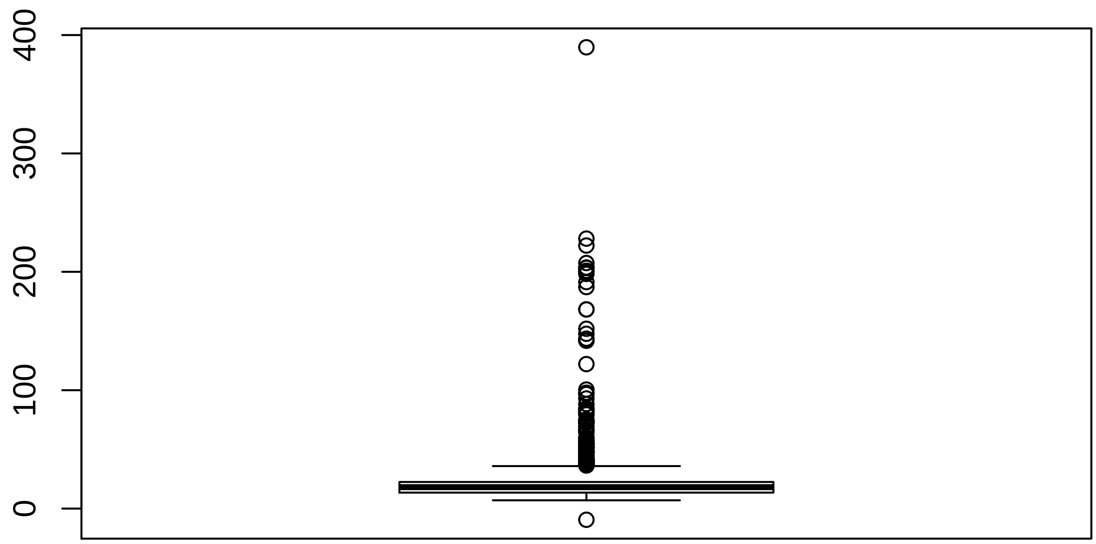
```
## [1] "double"
```

```r
typeof(response$time)
```

```
## [1] "character"
```

what does the rate look like

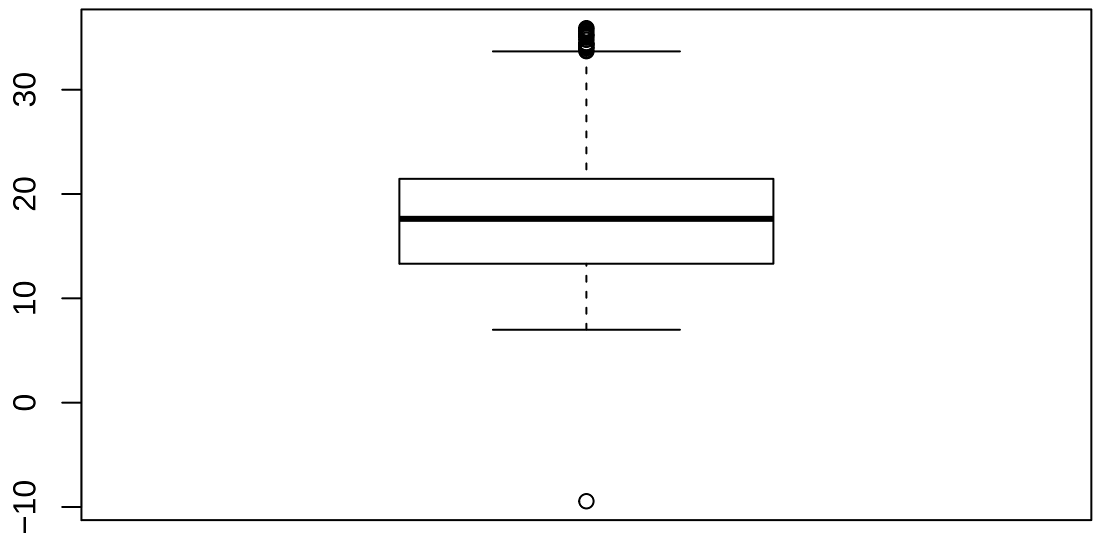
```r
lmp_boxplot <- boxplot(response$total_lmp_rt)
```



Can we make the data look better

Remove outliers

```
# remove outliers for boxplot
extreme_high <- lmp_boxplot$stats[5,]

lmp_noxtrm <- filter(response, total_lmp_rt < extreme_high)

boxplot(lmp_noxtrm$total_lmp_rt)
```



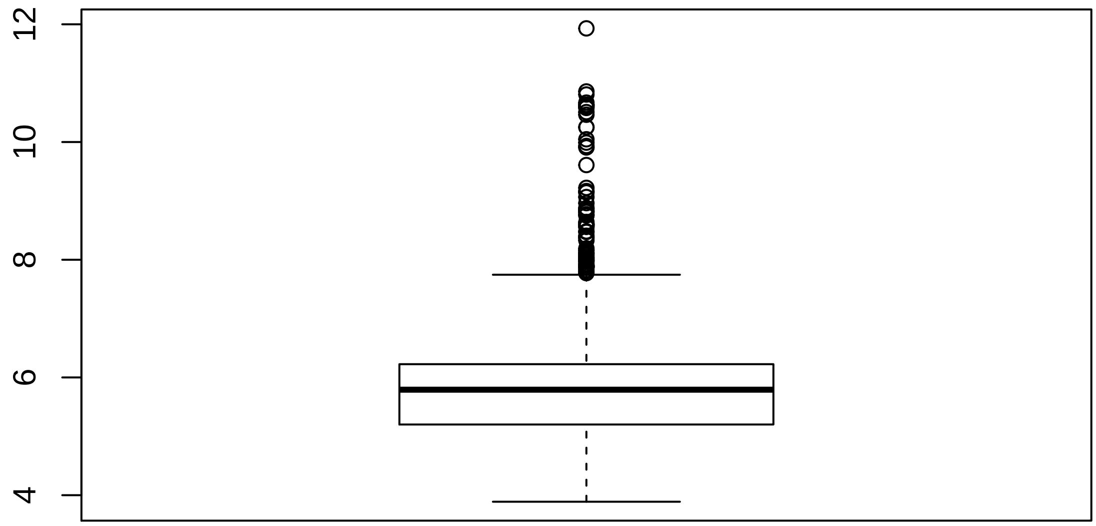
Removing extreme values does help the data look better

Log transform

```
lmp_log <- mutate(response, total_lmp_rt = log(total_lmp_rt^2+0.00000001))

summary(lmp_log$total_lmp_rt)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   3.889   5.201   5.792   5.825   6.225  11.931
```

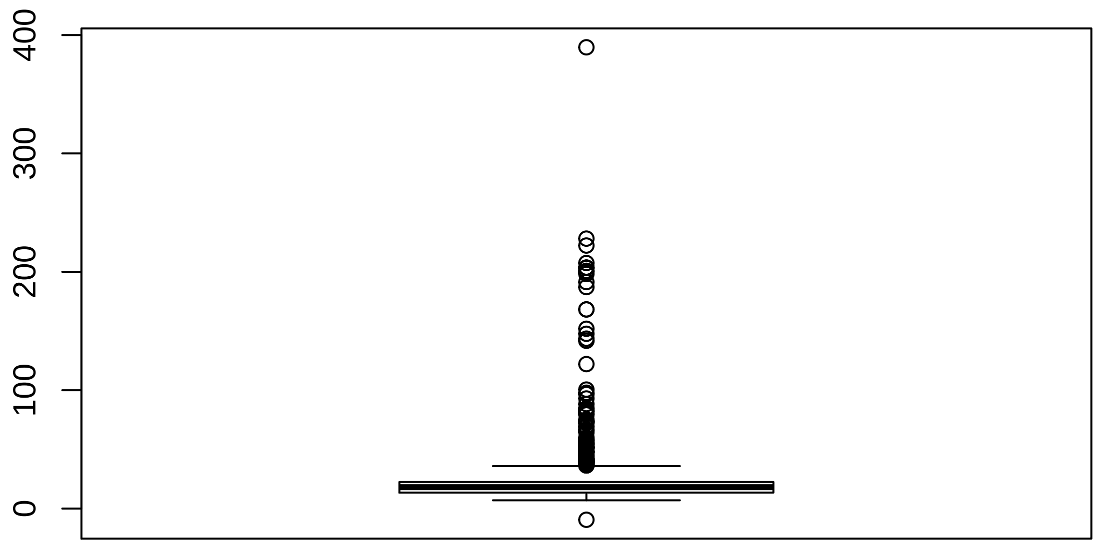
```
boxplot(lmp_log$total_lmp_rt)
```



Performing the log transform does help, but not as much as removing outliers

Decide on cleaned data

```
extrm_high <- boxplot(response$total_lmp_rt)$stats[5]
extrm_low <- boxplot(response$total_lmp_rt)$stats[1]
```




```r
electric <- response %>%
  filter(total_lmp_rt < extrm_high) %>%
  filter(total_lmp_rt > extrm_low) %>%
  mutate(datetime = ymd_hms(datetime)) %>%
  mutate(date = ymd(date)) %>%
  mutate(time = hms(time)) %>%
  na.omit()
```

```
## Warning: Problem with `mutate()` input `time`.
## i Some strings failed to parse, or all strings are NAs
## i Input `time` is `hms(time)`.
```

```
## Warning in .parse_hms(..., order = "HMS", quiet = quiet): Some strings
## failed to parse, or all strings are NAs
```

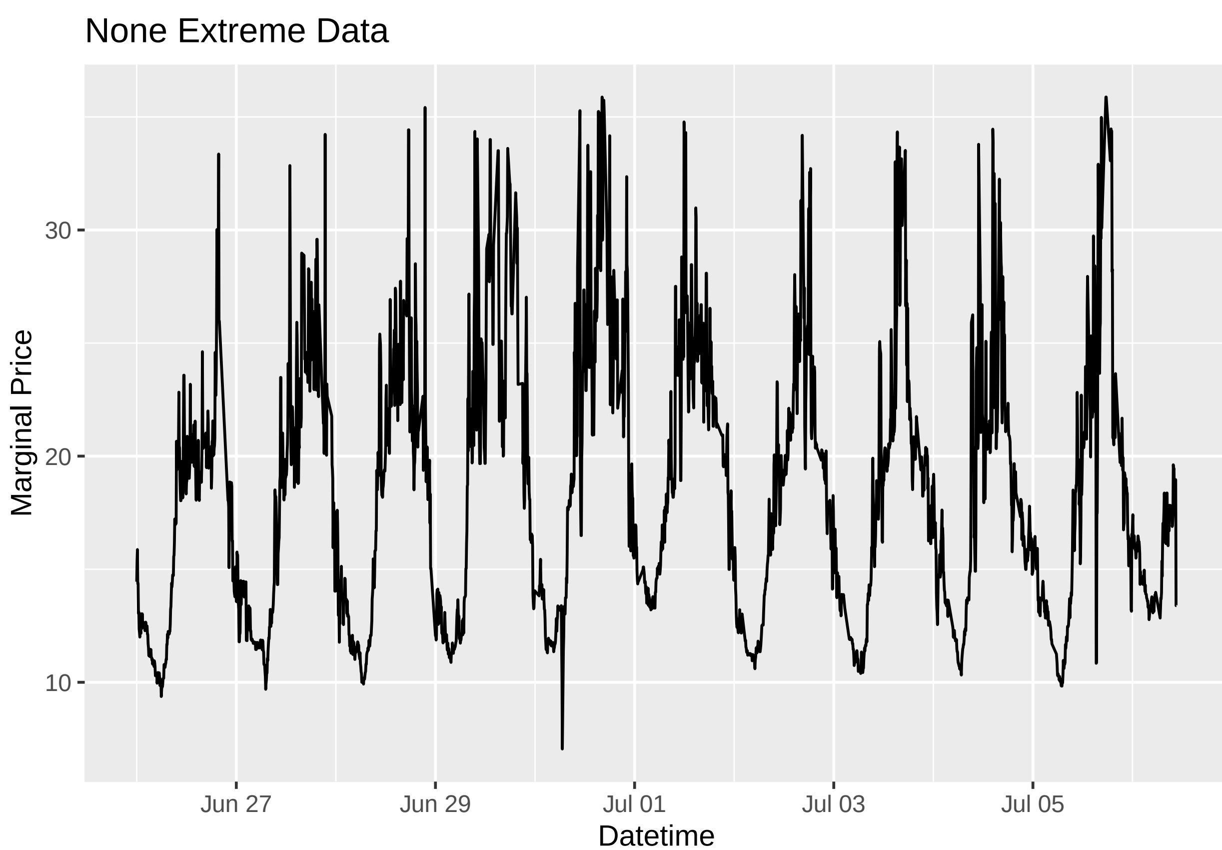
```r
summary(electric)
```

```
##     datetime                          date
##  Min.   :2020-06-26 00:00:00   Min.   :2020-06-26
##  1st Qu.:2020-06-28 11:16:15   1st Qu.:2020-06-28
##  Median :2020-07-01 06:17:30   Median :2020-07-01
##  Mean   :2020-07-01 04:05:14   Mean   :2020-06-30
##  3rd Qu.:2020-07-03 18:33:45   3rd Qu.:2020-07-03
##  Max.   :2020-07-06 10:35:00   Max.   :2020-07-06
```

```
##        time                                pnode_id         pnode_name
##  Min.   :1H 0M 0S                      Min.   :34964545   Length:2622
##  1st Qu.:6H 20M 0S                     1st Qu.:34964545   Class :character
##  Median :12H 5M 0S                     Median :34964545   Mode  :character
##  Mean   :12H 19M 25.6750572082383S     Mean   :34964545
##  3rd Qu.:18H 10M 0S                    3rd Qu.:34964545
##  Max.   :23H 55M 0S                    Max.   :34964545
##   total_lmp_rt
##  Min.   : 7.05
##  1st Qu.:13.41
##  Median :17.83
##  Mean   :18.21
##  3rd Qu.:21.73
##  Max.   :35.88
```

Visualize Cleaned data

```r
ggplot(electric, aes(x=datetime, y=total_lmp_rt)) +
  geom_line() +
  labs(title="None Extreme Data", y="Marginal Price", x="Datetime")
```



Create model

Remove trend

Create linear model and see statistical validity

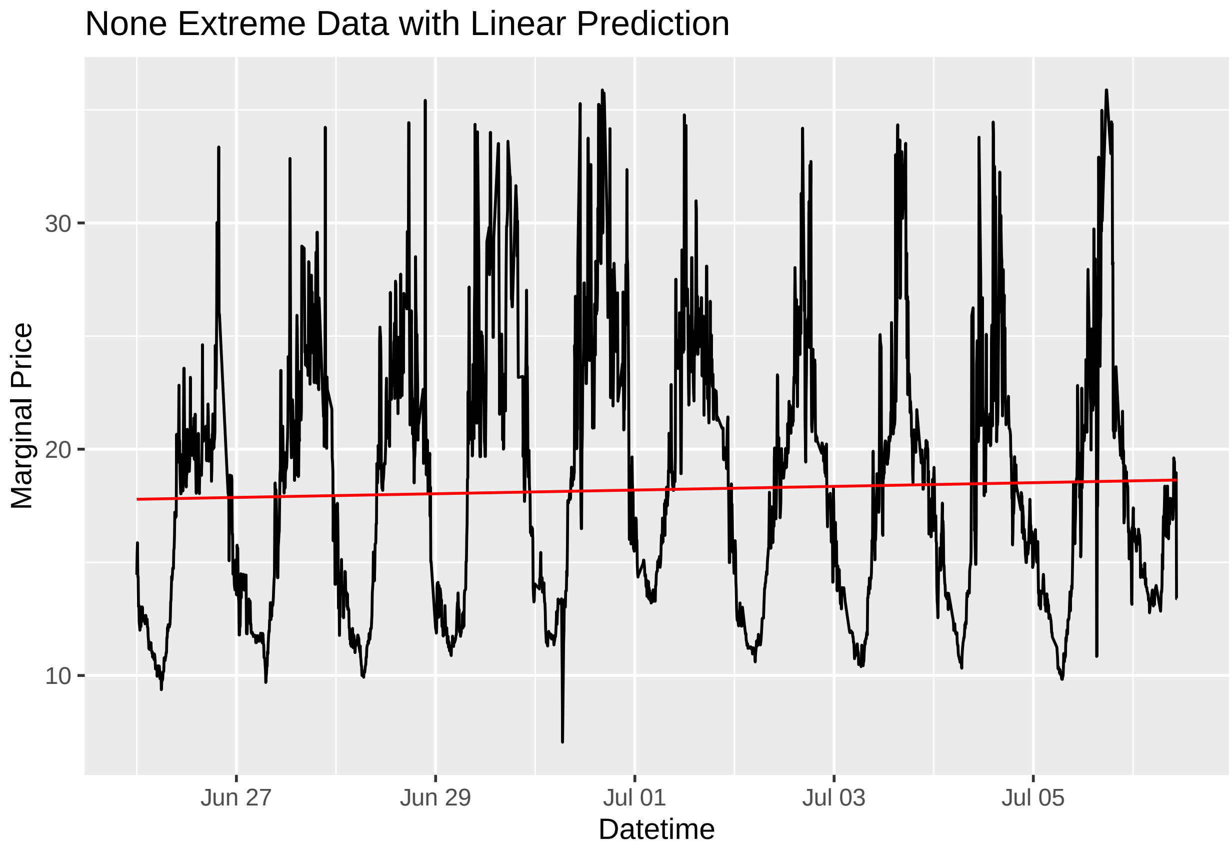
```
lm <- lm(electric$total_lmp_rt ~ electric$datetime)

summary(lm)
```

```
##
## Call:
## lm(formula = electric$total_lmp_rt ~ electric$datetime)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.0858  -4.8751  -0.4141   3.4873  17.7117
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -1.484e+03  6.685e+02  -2.219   0.0266 *
## electric$datetime  9.424e-07  4.195e-07   2.246   0.0248 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.637 on 2620 degrees of freedom
## Multiple R-squared:  0.001922,   Adjusted R-squared:  0.001541
## F-statistic: 5.046 on 1 and 2620 DF,  p-value: 0.02476
```

Model is signficant, see what it looks like on the data

Plot the trendline

```
ggplot(electric, aes(x=datetime, y=total_lmp_rt)) +
  geom_line() +
  geom_line(data = fortify(lm), aes(x = electric$datetime, y = .fitted), color="red") +
  labs(title="None Extreme Data with Linear Prediction", y="Marginal Price", x="Datetime")
```

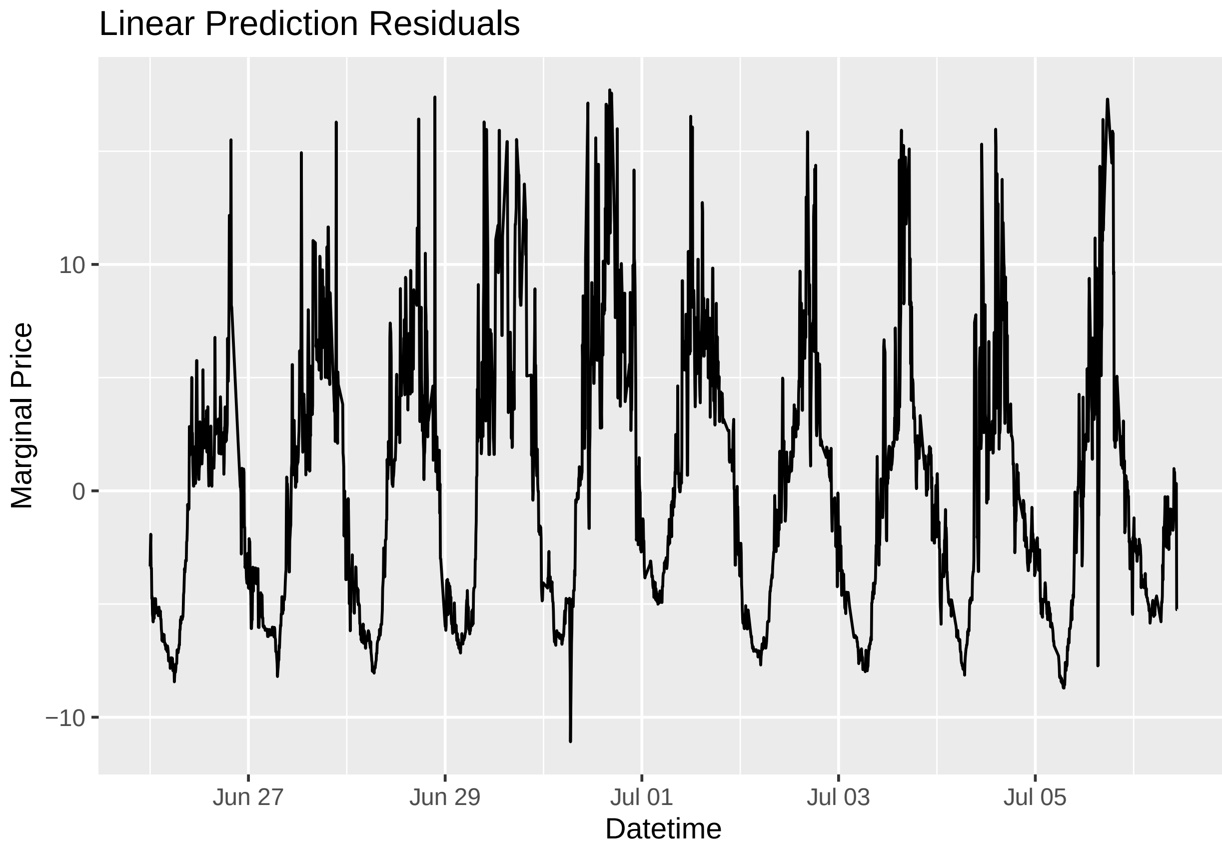


Use prediction in the future find residuals and save in dataframe

```
electric$lm_res <- lm$residuals
electric$lm_pred <- lm$fitted.values
```

Plot the residuals

```
ggplot(electric, aes(x=datetime, y=lm_res)) +
  geom_line() +
  labs(title="Linear Prediction Residuals", y="Marginal Price", x="Datetime")
```

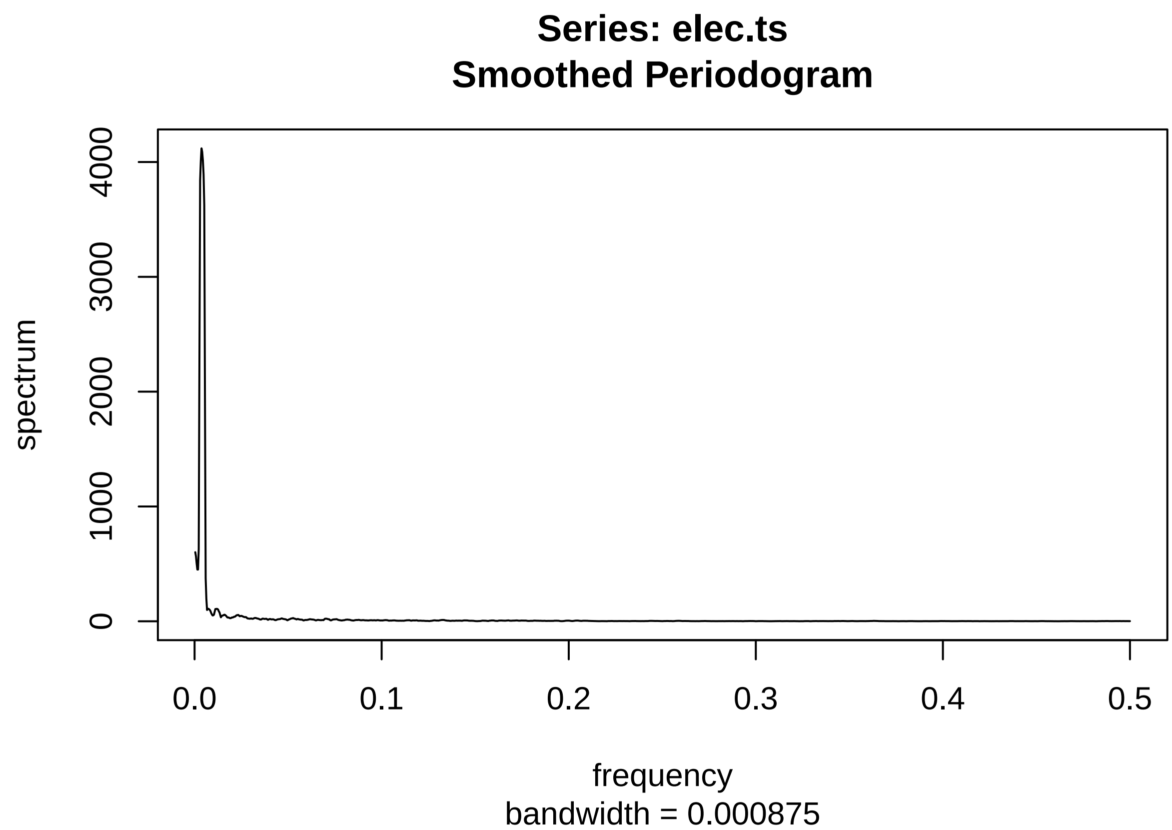


This data looks stationary so we are good to move on

Remove sinusoidal movement

Find any sinusoidal movement

```
# create time series
elec.ts <- ts(electric$lm_res)

# find frequencies of high influence
pgram <- spec.pgram(elec.ts, spans=9, demean=T, log='no')
```




```r
# sort the frequencies based on influence
sorted.spec <- sort(pgram$spec, decreasing=T, index.return=T)

# convert to periods
sorted.omegas <- pgram$freq[sorted.spec$ix]
sorted.Ts <- 1/pgram$freq[sorted.spec$ix]

# the cutoff for influential
pgram.cutoff <- 5

# the sampling rate per day
print('sampling rate')
```

```
## [1] "sampling rate"
```

```r
nrow(electric)/as.numeric(max(electric$date)-min(electric$date))
```

```
## [1] 262.2
```

```r
# the top periods
print('top periods')
```

```
## [1] "top periods"
```

```r
sorted.Ts[1:pgram.cutoff]
```

```
## [1] 270.0000 245.4545 225.0000 300.0000 207.6923
```

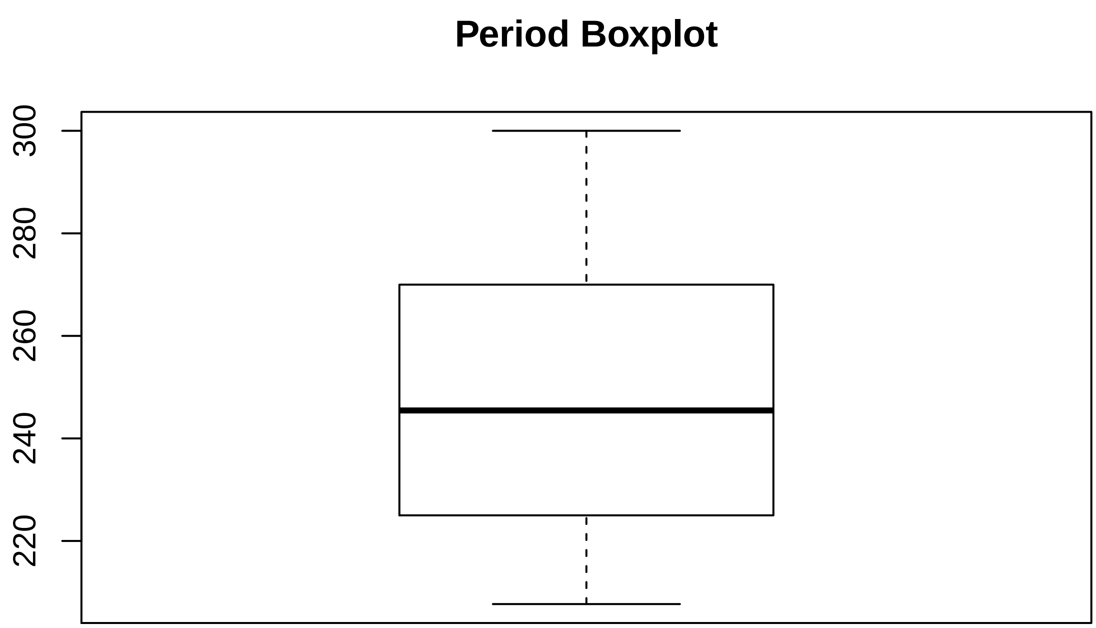
```r
# top frequencies
## to double check that this makes sense based on periodogram
print('top frequencies')
```

```
## [1] "top frequencies"
```

```r
sorted.omegas[1:pgram.cutoff]
```

```
## [1] 0.003703704 0.004074074 0.004444444 0.003333333 0.004814815
```

```r
# visual
pgram.box <- boxplot(sorted.Ts[1:pgram.cutoff], main="Period Boxplot")
```



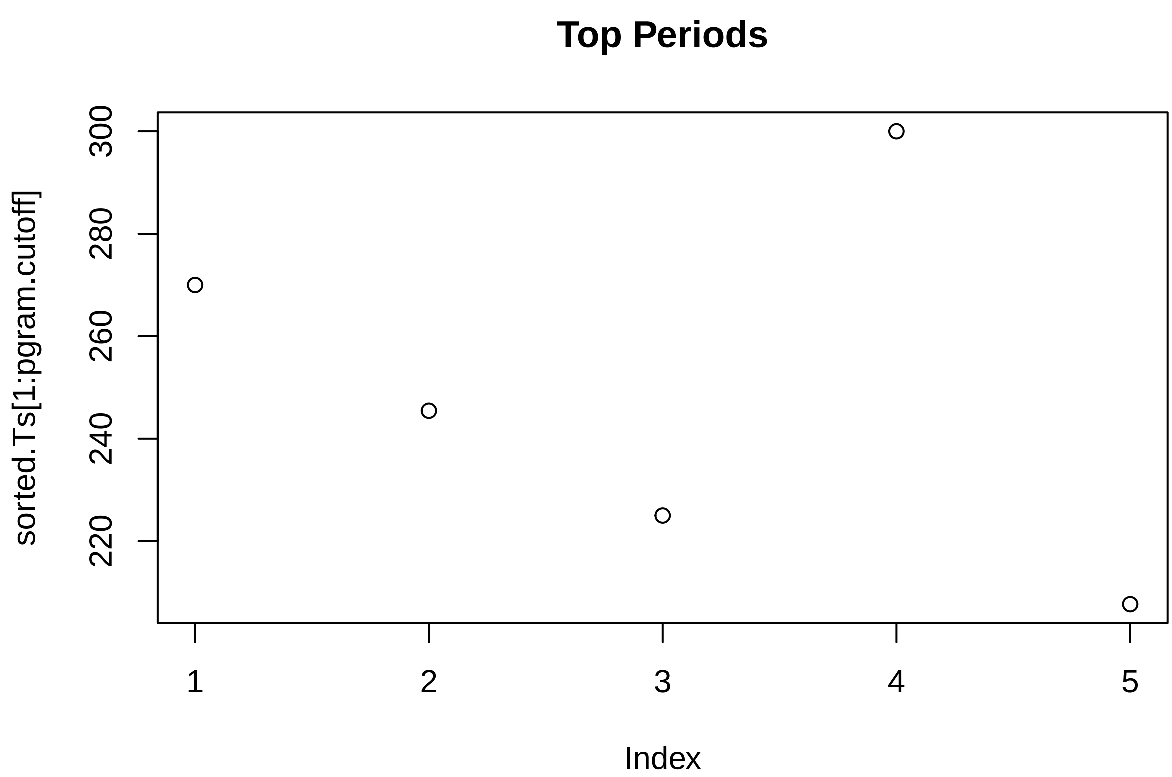

```r
# the average influential period
print('mean of top periods')
```

```
## [1] "mean of top periods"
```

```r
pgram.box.mean <- pgram.box$stats[3]
print(pgram.box.mean)
```

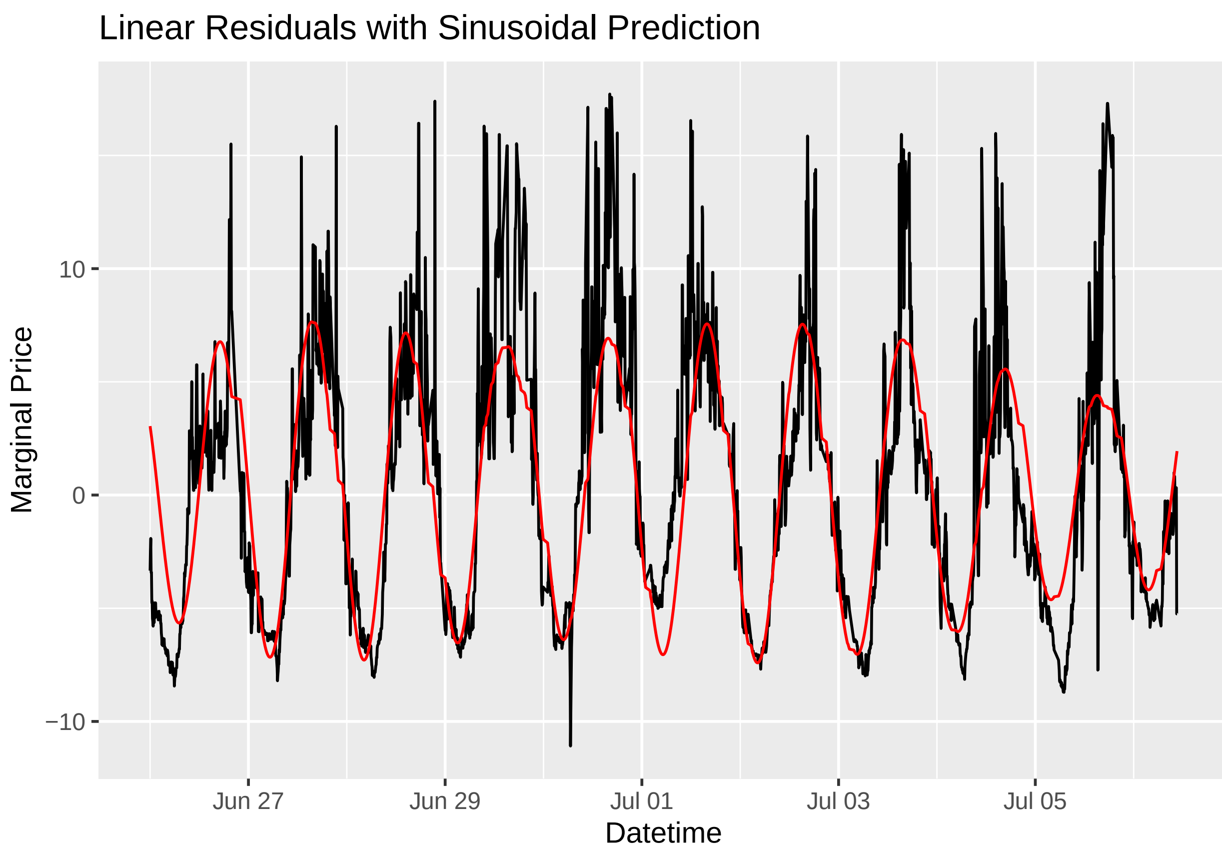
```
## [1] 245.4545
```

```r
# plot top periods
plot(sorted.Ts[1:pgram.cutoff], main = "Top Periods")
```




```r
### create a model for the seasonality
# assign potential periods to variables
p1 <- sorted.Ts[1]
p2 <- sorted.Ts[2]
p3 <- sorted.Ts[3]
p4 <- sorted.Ts[4]
p5 <- sorted.Ts[4]

# create time variable
time<-c(1:length(elec.ts))

# model
sin_mov <- lm(elec.ts ~
                   sin(2*pi*time/p1) +
                   cos(2*pi*time/p1) +
                   sin(2*pi*time/p2) +
```

```
                     cos(2*pi*time/p2) +
                     sin(2*pi*time/p3) +
                     cos(2*pi*time/p3) +
                     sin(2*pi*time/p4) +
                     cos(2*pi*time/p4) +
                     sin(2*pi*time/p5) +
                     cos(2*pi*time/p5)
                     )

### visualize
ggplot(electric, aes(x=datetime, y=electric$lm_res)) +
  geom_line() +
  geom_line(data = fortify(lm), aes(x = electric$datetime, y = sin_mov$fitted.values), color="red") +
  labs(title="Linear Residuals with Sinusoidal Prediction", y="Marginal Price", x="Datetime")
```

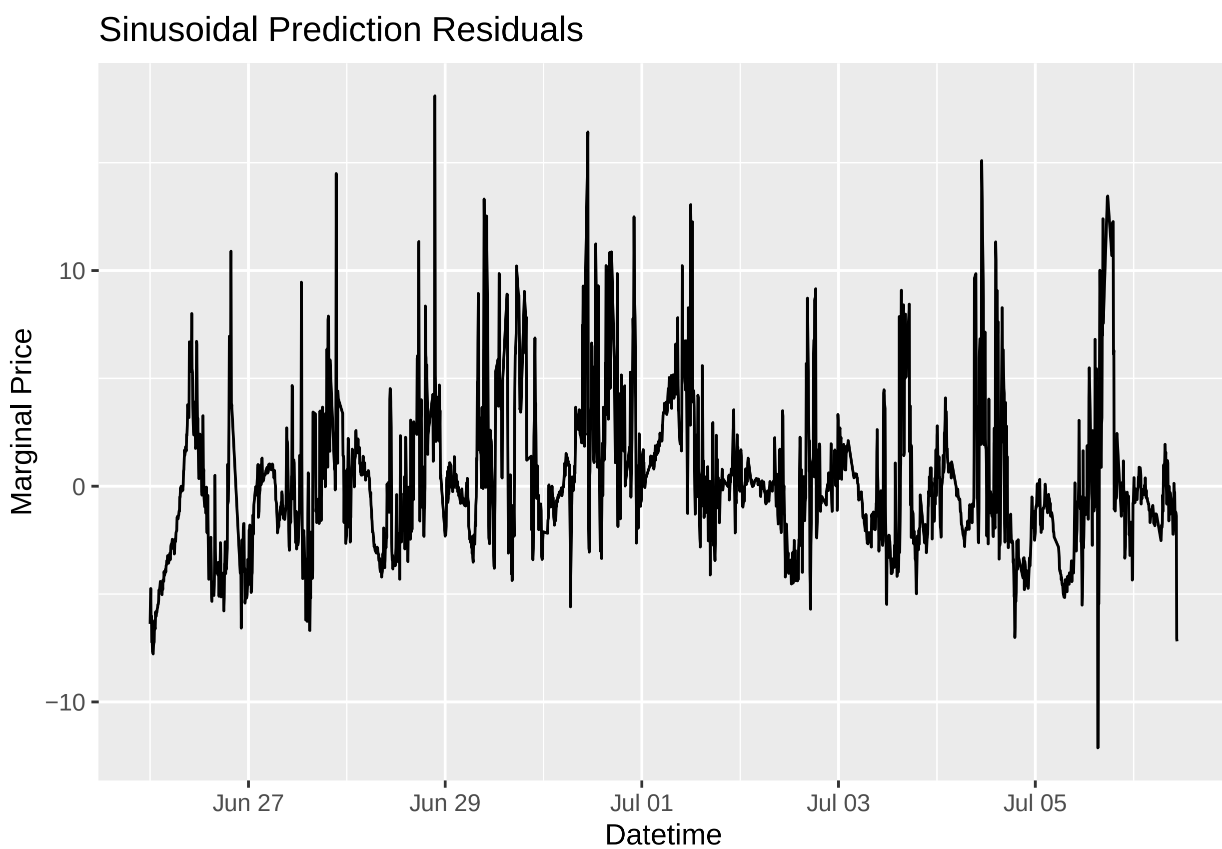


Use model to store residuals

```
electric$sin_res <- sin_mov$residuals
electric$sin_pred <- sin_mov$fitted.values
```

Plot residuals

```r
ggplot(electric, aes(x=datetime, y=sin_res)) +
  geom_line() +
  labs(title="Sinusoidal Prediction Residuals", y="Marginal Price", x="Datetime")
```



There is still some cyclic movement, we can address this later

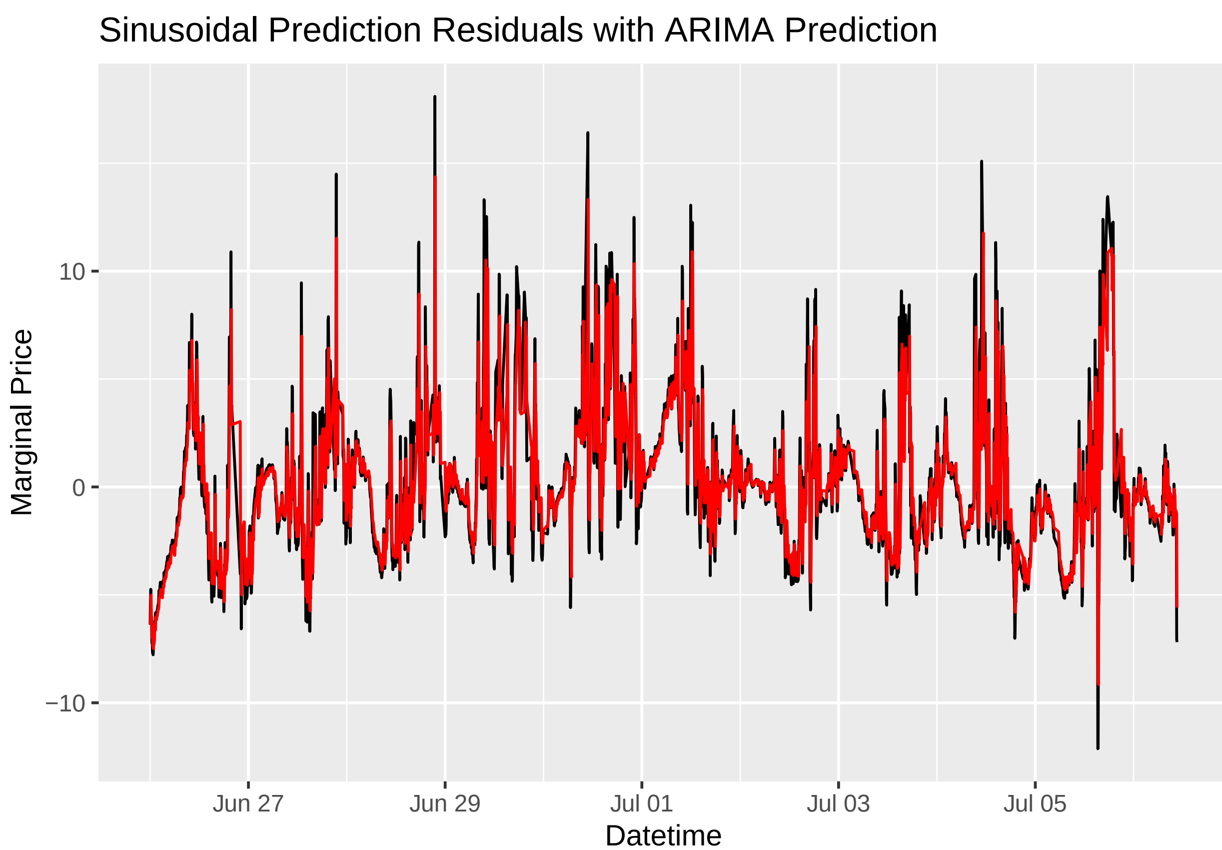
Model Residuals

```r
auto <- auto.arima(electric$sin_res, approximation = FALSE)

summary(auto)
```

```
## Series: electric$sin_res
## ARIMA(1,1,1)
##
## Coefficients:
##          ar1      ma1
##       0.6957  -0.9354
## s.e.  0.0242   0.0134
##
## sigma^2 estimated as 3.114:  log likelihood=-5207.1
## AIC=10420.21   AICc=10420.22   BIC=10437.82
##
```

```
## Training set error measures:
##                       ME     RMSE      MAE       MPE     MAPE     MASE
## Training set 0.006779134 1.763738 1.016539 -19.17687 329.0944 1.028765
##                     ACF1
## Training set 0.006981379
```

```
ggplot(electric, aes(x=datetime, y=sin_res)) +
  geom_line() +
  geom_line(aes(x=datetime, y=auto$fitted), colour="red") +
  labs(title="Sinusoidal Prediction Residuals with ARIMA Prediction", y="Marginal Price", x="Datetime")
```



Save residual prediction

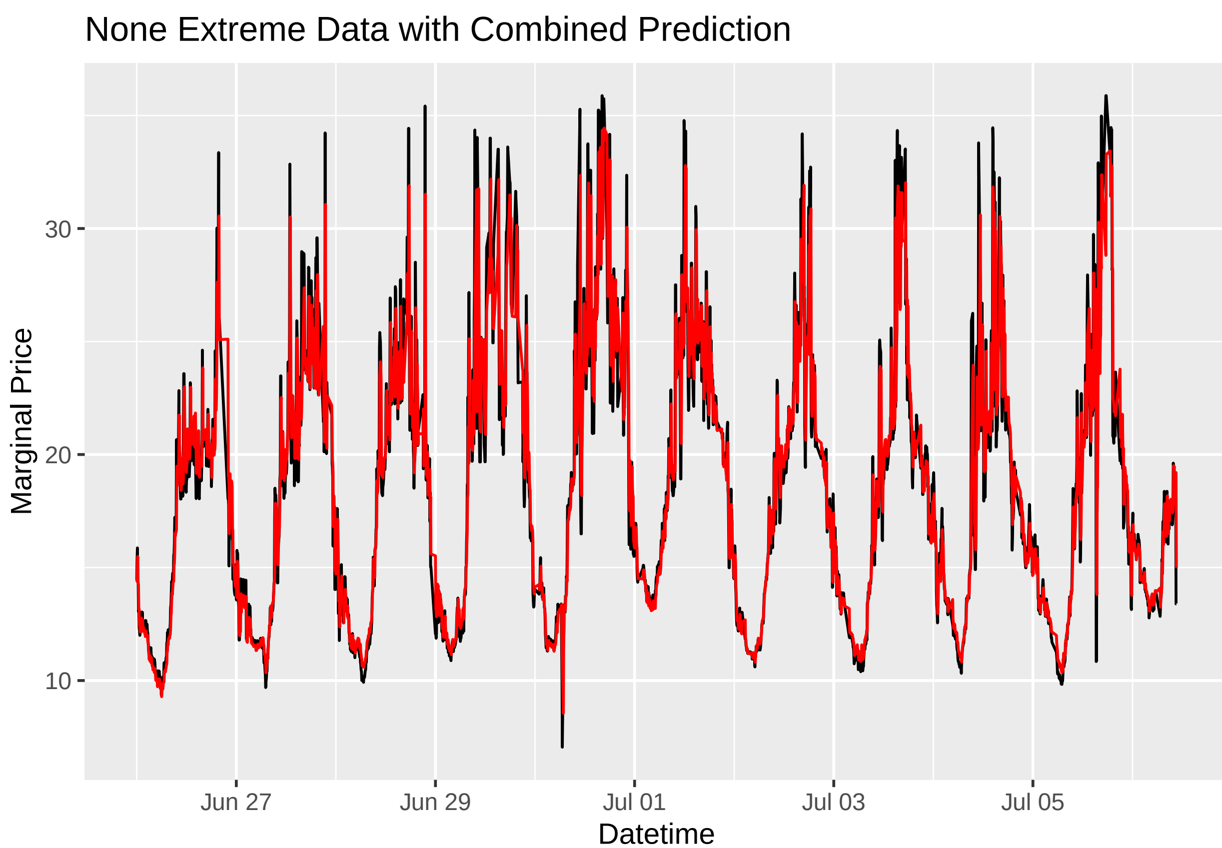
```
electric$ar_pred <- auto$fitted
```

Combine all models

```
electric$model_final <- electric$ar_pred + electric$lm_pred + electric$sin_pred
```

Plot the final model

```r
ggplot(electric, aes(x=datetime, y=total_lmp_rt)) +
  geom_line() +
  geom_line(aes(x=datetime, y=model_final), colour="red") +
  labs(title="None Extreme Data with Combined Prediction", y="Marginal Price", x="Datetime")
```



Model validity and statistics

End of file

```r
"End of file"
```

```
## [1] "End of file"
```