Time series modelling

Data Analysis

Energy Consumption data from 2010-01-01 to 2015-05-31. Each data point represents the daily energy consumption. specifically, we are using the log-transformed time series to deal with fluctuating variance/heteroskedasticity.

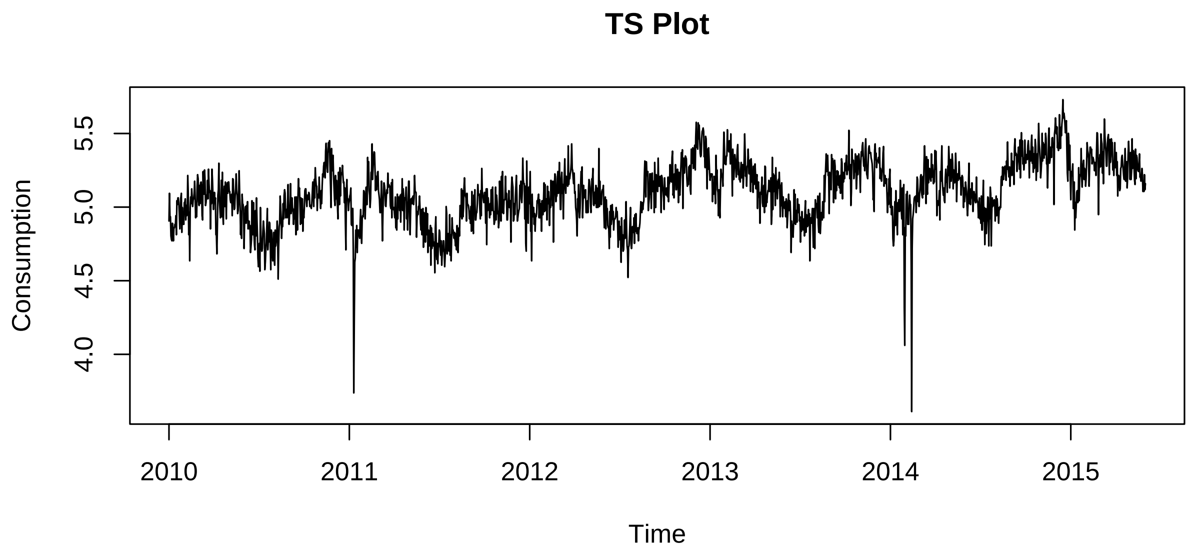
```
library(zoo)
library(lubridate)
library(mgcv)
library(TSA)
library(dynlm)
```

Instructions on reading the data

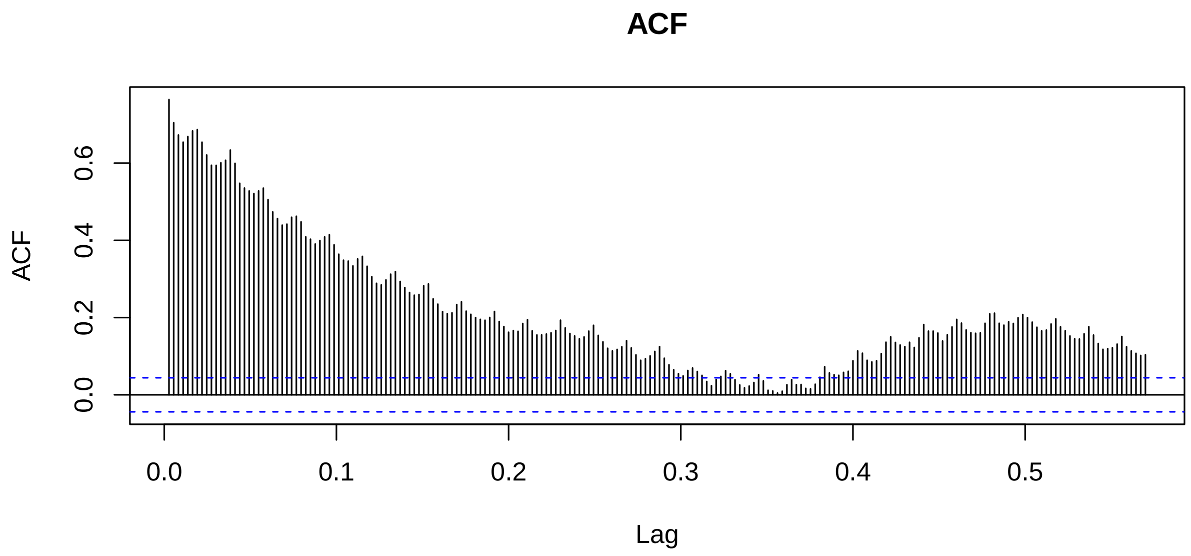
```
#Read in data
Input_Data<-read.csv("Data.csv")
year = Input_Data$Year
month = Input_Data$Month
day = Input_Data$Day
datemat = cbind(as.character(day),as.character(month),as.character(year))
paste.dates = function(date){
    day = date[1]; month=date[2]; year = date[3]
    return(paste(day,month,year,sep="/"))
}
dates = apply(datemat,1,paste.dates)
dates = as.Date(dates, format="%d/%m/%Y")
Input_Data = cbind(dates,Input_Data)
attach(Input_Data)
tmp = log(Volume) # log-transform the data

# Extract Catagorical Month and Week Indicators
month = as.factor(format(dates,"%b"))
week = as.factor(weekdays(dates))

# Convert original data into time series (ts)
Consumption = ts(tmp,start=c(2010,1,1),frequency=365) # Org data
```

Part 1: Trend and Seasonality fitting

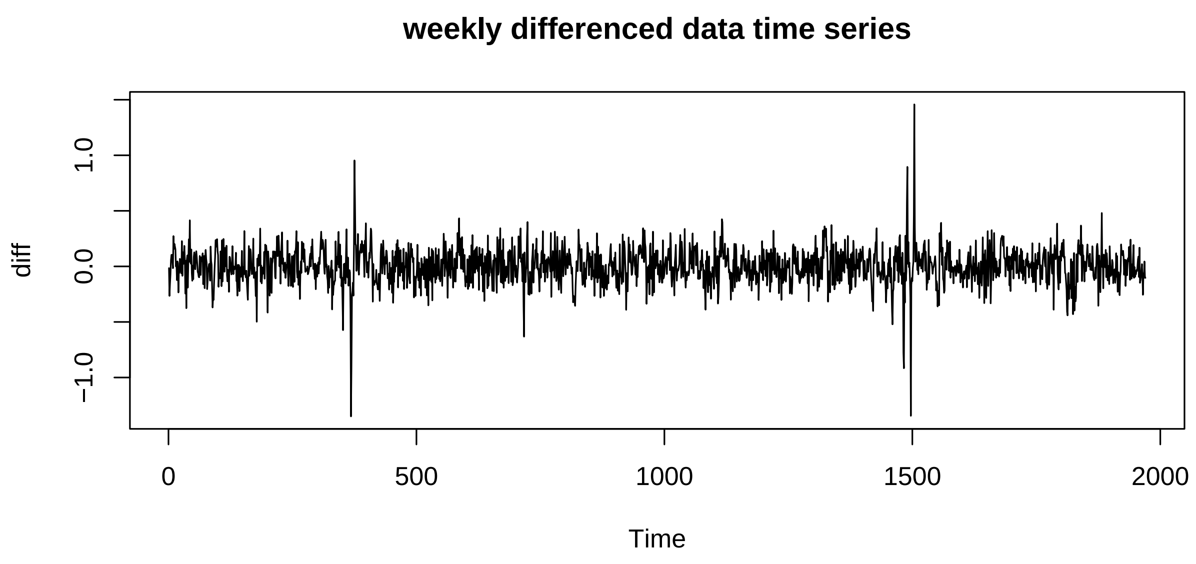
```
ts.plot(Consumption, main="TS Plot")
```



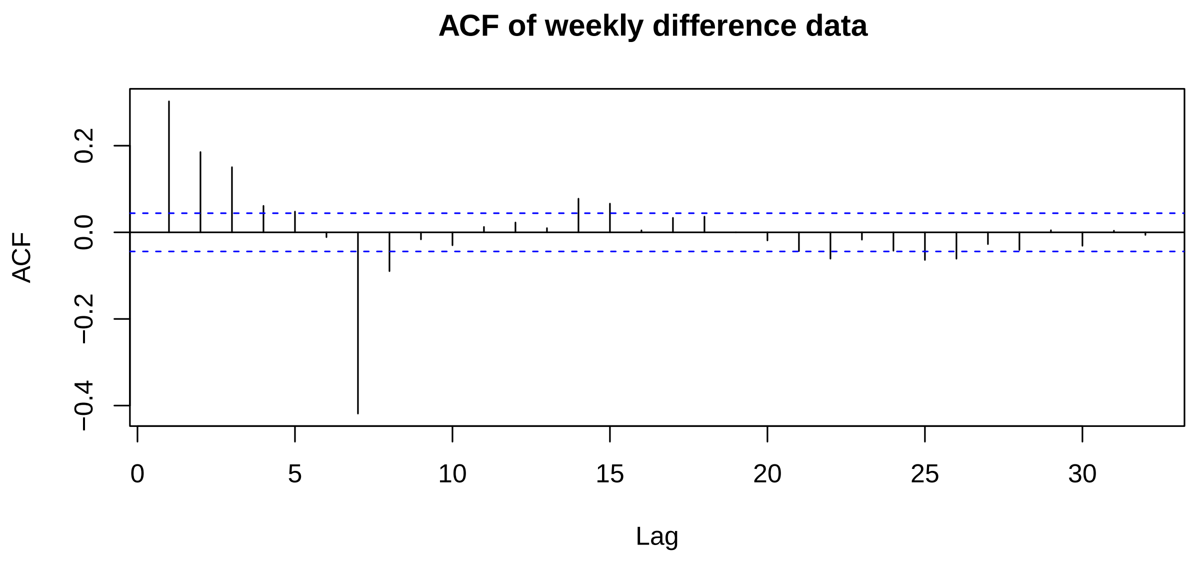
```
acf(Consumption,main="ACF",lag.max=52*4)
```



```
diff.consumption=diff(Consumption,lag=7);
diff.consumption = diff.consumption[!is.na(diff.consumption)]
diff=ts(diff.consumption)
plot(diff,type="l",main="weekly differenced data time series")
```



```
acf(diff, main="ACF of weekly difference data ")
```



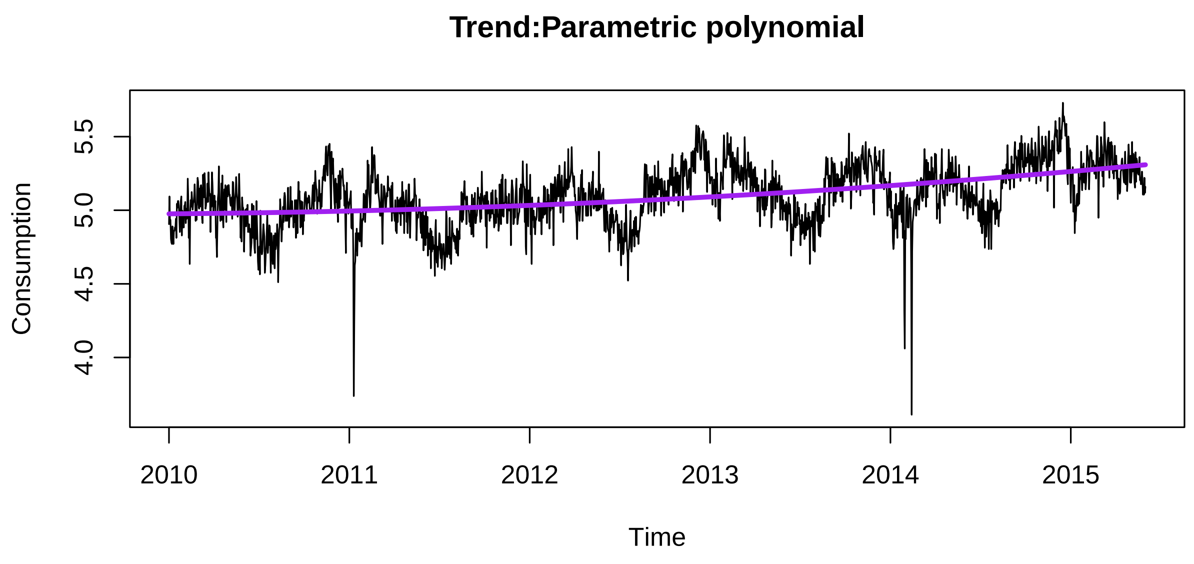
Response: From the TS plot we can see periods of in increasing and decreasing trends. We can also see the presence of non constant variance. From the ACF plot, we can see many lags have significant autocorrelation and we can also clearly observe the seasonality. On the differenced time series, the trend appears to be removed (Constant mean) but there is still presence of non constant variance. From the ACF plot of differenced data, we can see the effect of seasonality has been removed to a great extent. The ACF plot still has significant autocorrelation in many lags but dies down quickly. The differenced time series is much closer to stationarity than the original series.

Trend estimation models on the time series data: Parametric Quadratic Polynomial, and Splines.

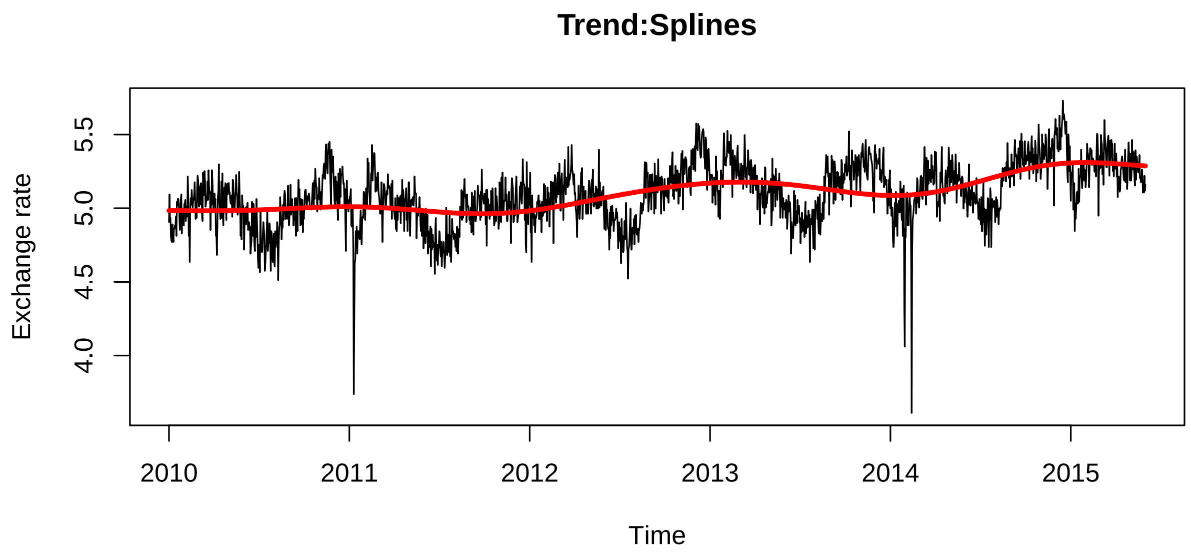
```
time.pts<-c(1:length(Consumption))
time.pts<-c(time.pts - min(time.pts))/max(time.pts)
# Parametric polynomial
x1<-time.pts
x2<-time.pts^2
lm.fit<-lm(Consumption~x1+x2)
summary(lm.fit)
```

```
##
## Call:
## lm(formula = Consumption ~ x1 + x2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.5667 -0.1123  0.0131  0.1174  0.4905
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.97572    0.01213 410.222  < 2e-16 ***
## x1           0.05089    0.05605   0.908    0.364
## x2           0.28244    0.05430   5.202 2.18e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.18 on 1974 degrees of freedom
## Multiple R-squared:  0.2307, Adjusted R-squared:  0.2299
## F-statistic: 295.9 on 2 and 1974 DF,  p-value: < 2.2e-16
```

```
fit.lm= ts(fitted(lm.fit),start=c(2010,1,1),frequency=365)
ts.plot(Consumption,main="Trend:Parametric polynomial")
lines(fit.lm,lwd=3,col="purple")
```



```
# Splines
gam.fit<-gam(Consumption~s(time.pts))
fit.gam= ts(fitted(gam.fit),start=c(2010,1,1),frequency=365)

ts.plot(Consumption,main="Trend:Splines",ylab="Exchange rate")
lines(fit.gam,lwd=3,col="red")
```



```
#Precision
preds1 <- as.vector(fit.lm)
preds2 <- as.vector(fit.gam)
obs <- as.vector(Consumption)

PM1 <- sum((preds1-obs)^2)/sum((obs-mean(obs))^2)
PM2 <- sum((preds2-obs)^2)/sum((obs-mean(obs))^2)
```

Response: From the plot of overlayed fitted values, we can see that splines model is able to capture the overall trend better than the parametric polynomial regression.The accuracy measure for parametric polynomial is 0.769 and for the spline model it is 0.718.

Seasonality estimation model: Using ANOVA approach fr(1) Using Month seasonality only; (2) Using both Months and Week seasonality.

```
model1=dynlm(Consumption~month)
summary(model1)
```

```
##
## Time series regression with "ts" data:
## Start = 2010(1), End = 2015(152)
##
## Call:
## dynlm(formula = Consumption ~ month)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.53559 -0.10808 -0.00224  0.11133  0.51191
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.10405    0.01242 411.091  < 2e-16 ***
## monthAug    -0.08959    0.01825  -4.908 9.95e-07 ***
## monthDec     0.14405    0.01825   7.892 4.90e-15 ***
## monthFeb     0.04246    0.01784   2.380 0.017429 *
## monthJan    -0.10657    0.01742  -6.119 1.14e-09 ***
## monthJul    -0.26858    0.01825 -14.714  < 2e-16 ***
## monthJun    -0.18326    0.01842  -9.951  < 2e-16 ***
## monthMar     0.09289    0.01742   5.333 1.08e-07 ***
## monthMay     0.01018    0.01742   0.584 0.558951
## monthNov     0.14592    0.01842   7.923 3.83e-15 ***
## monthOct     0.06501    0.01825   3.562 0.000377 ***
## monthSep     0.03170    0.01842   1.721 0.085355 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1666 on 1965 degrees of freedom
## Multiple R-squared:  0.3438, Adjusted R-squared:  0.3401
## F-statistic: 93.59 on 11 and 1965 DF,  p-value: < 2.2e-16
```

```
fit.model1=model1$fitted


model2=dynlm(Consumption~month+week)
summary(model2)
```

```
##
## Time series regression with "ts" data:
## Start = 2010(1), End = 2015(152)
##
## Call:
## dynlm(formula = Consumption ~ month + week)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.50319 -0.10499 -0.00312  0.11110  0.43990
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    5.092126   0.015125 336.671  < 2e-16 ***
## monthAug      -0.089578   0.017898  -5.005 6.09e-07 ***
## monthDec       0.143522   0.017898   8.019 1.82e-15 ***
## monthFeb       0.042583   0.017495   2.434 0.015025 *
## monthJan      -0.106432   0.017078  -6.232 5.62e-10 ***
## monthJul      -0.268634   0.017898 -15.009  < 2e-16 ***
## monthJun      -0.183184   0.018058 -10.144  < 2e-16 ***
## monthMar       0.092380   0.017078   5.409 7.11e-08 ***
## monthMay       0.010316   0.017078   0.604 0.545908
## monthNov       0.145963   0.018058   8.083 1.09e-15 ***
## monthOct       0.065157   0.017898   3.640 0.000279 ***
## monthSep       0.031303   0.018058   1.733 0.083175 .
## weekMonday     0.083796   0.013744   6.097 1.30e-09 ***
## weekSaturday   0.008763   0.013732   0.638 0.523426
## weekSunday     0.027137   0.013732   1.976 0.048272 *
## weekThursday  -0.025480   0.013744  -1.854 0.063896 .
## weekTuesday    0.010313   0.013744   0.750 0.453135
## weekWednesday -0.020604   0.013744  -1.499 0.134002
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1633 on 1959 degrees of freedom
## Multiple R-squared:  0.371,  Adjusted R-squared:  0.3655
## F-statistic: 67.96 on 17 and 1959 DF,  p-value: < 2.2e-16
```

```
fit.model2=model2$fitted

anova(model1,model2)
```

```
## Analysis of Variance Table
##
## Model 1: Consumption ~ month
## Model 2: Consumption ~ month + week
##   Res.Df    RSS Df Sum of Sq     F    Pr(>F)
## 1   1965 54.524
## 2   1959 52.265  6    2.2588 14.11 8.696e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Response: Small p value shows that we can reject null hypothesis (Simpler model is better). SO the model with monthly and weekly seasonality is better.

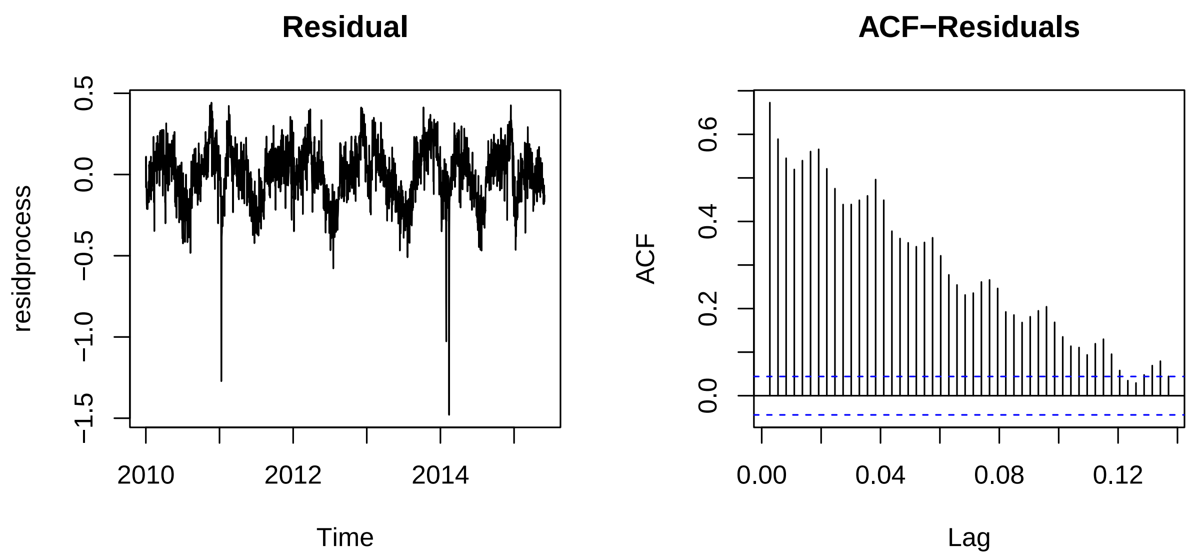
Estimating both trend and seasonality: Parametric Polynomial Regression and Nonparametric modeling (spline). For simplicity,I am using Month+Week for seasonality for both trend fittings.

```
#Equally spaced time points

lm.fit=dynlm(Consumption~x1+x2+month+week)
summary(lm.fit)
gam.fit=gam(Consumption~s(time.pts)+month+week)
fit.gam=ts(fitted(gam.fit),start=c(2010,1,1),frequency=365)
fit.lm=ts(fitted(lm.fit),start=c(2010,1,1),frequency=365)
ts.plot(Consumption,main="Trend-Seasonality using Parametric polynomial regression",ylab="Temp")
lines(fit.lm,lwd=3,col="yellow")
ts.plot(Consumption,main="Trend-Seasonality using Non parametric model",ylab="Temp")
lines(fit.gam,lwd=3,col="red")

preds1 <- as.vector(fit.lm)
preds2 <- as.vector(fit.gam)
obs <- as.vector(Consumption)

PM1 <- sum((preds1-obs)^2)/sum((obs-mean(obs))^2)
PM2 <- sum((preds2-obs)^2)/sum((obs-mean(obs))^2)
```

Response: The spline model has lower PM measure (0.38737) compared to the parametric model (0.438). In general both models are able to reasonably capture the trend and seasonality in the data.

Model residual Analysis

```
residprocess=Consumption-fitted(gam.fit)
residprocess=ts(residprocess,start=c(2010,1,1),frequency=365)
par(mfcol=c(1,2))
ts.plot(residprocess,main="Residual")
acf(residprocess,main="ACF-Residuals",lag.max =50)
```



Response: From the residuals we can see that trend has been removed. The time series plot still shows non constant variance. From the ACF plot we can see that the effect of seasonality is greatly removed. But it still has significant autocorrealtion in the intial lags and dies down quickly. The residual process is non stationary but it is much closer to stationarity than the original model.

ARIMA FORECASTING Using the iterative model selection approach (with corrected AIC score as model performance metric) I found that ARIMA model with the order of (2,0,1), and seasonal order (1,0,1), period=7 captures this data perfectly.

```
train <- Consumption[1:(length(Consumption)-14)]

mod = arima(train, order = c(2,0,1),seasonal = list(order =
c(1,0,1),period=7),method = "ML")
AIC(mod)
```

```
## [1] -2909.786
```

```
mod
```

```
##
## Call:
## arima(x = train, order = c(2, 0, 1), seasonal = list(order = c(1, 0, 1), period = 7),
##     method = "ML")
##
## Coefficients:
##          ar1      ar2      ma1    sar1     sma1  intercept
##       1.1513  -0.1669  -0.7787  0.9992  -0.9865     5.0949
## s.e.  0.0328   0.0309   0.0218  0.0008   0.0058     0.1997
##
## sigma^2 estimated as 0.01312:  log likelihood = 1461.89,  aic = -2911.79
```

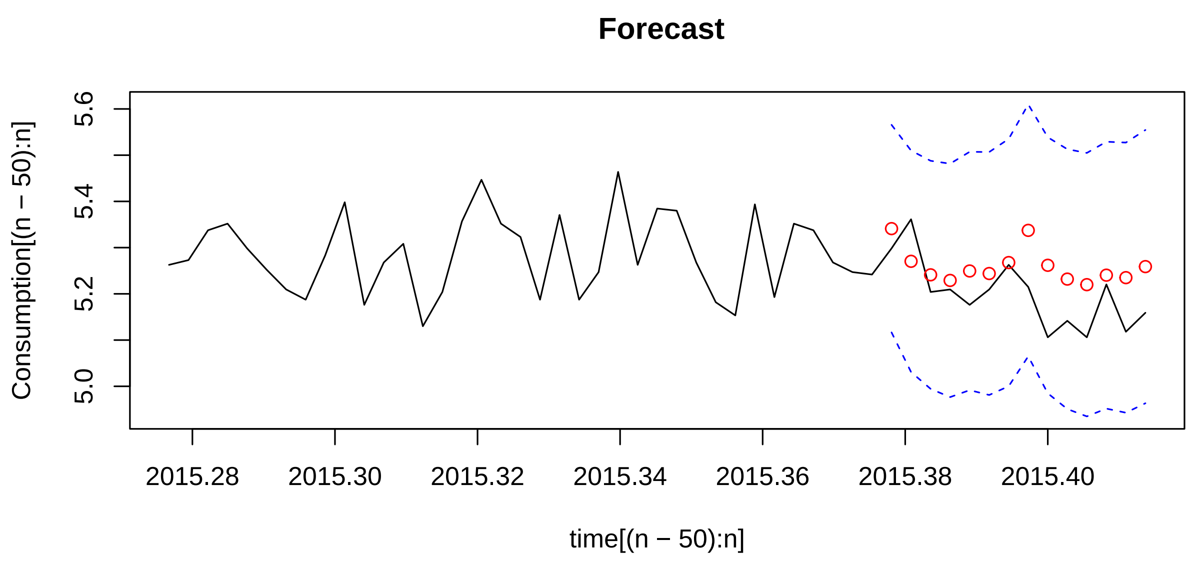
```
## p-value function for the z-test taking as input the test statistic
pvalue.coef <- function(tv){
2*(1-pnorm(abs(tv)))
}
## compute the test statistics
tvalues <-as.numeric(mod$coef)/as.numeric(sqrt(diag(mod$var.coef)))

## Apply the pvalue.coef function
pvalues<- sapply(tvalues, pvalue.coef)

pvalues
```

```
## [1] 0.000000e+00 6.805546e-08 0.000000e+00 0.000000e+00 0.000000e+00
## [6] 0.000000e+00
```

Response: From the p values we can observe that all coefficients are statistically significant. The AIC of the model is -2909.7

Forecasting for the next two weeks with 95% CI also provided in the plot

```
time <- time(Consumption)
n<-length(Consumption)
ntrain<-length(train)
forecast <- as.vector(predict(mod,n.ahead=14))
lower <- forecast$pred-1.96*forecast$se
upper<- forecast$pred+1.96*forecast$se

plot(time[(n-50):n],Consumption[(n-50):n],type="l",ylim=c(min(lower),max(upper)),main=" Forecast")
lines(time[(ntrain+1):n],lower,lty=2,lwd=1,col="blue")
points(time[(ntrain+1):n],forecast$pred,col="red")
lines(time[(ntrain+1):n],upper,lty=2,lwd=1,col="blue")
```



Response: Most of the predicted values are notably higher than the actual values; However, the actual values lie within the 95% confidence intervals. In general the forecasts are good.

MODEL PERFORMANCE EVALUATION Mean squared error (MSE), Mean absolute error (MAE), mean absolute percent error (MAPE), percision measure (PM).

```
prediction = forecast$pred
observed = Consumption[(ntrain+1):n]
#MSE
mean((observed-mean(observed))^2)
```

```
## [1] 0.005035677
```

```
#MAE
mean(abs(prediction-observed))
```

```
## [1] 0.07302701
```

```
#MAPE
mean(abs(prediction-observed)/observed)
```

```
## [1] 0.01411033
```

```
#Precision
sum((prediction-observed)^2)/sum((observed-mean(observed))^2)
```

```
## [1] 1.459355
```

```
#No of points in prediction band
sum(observed<lower)+sum(observed>upper)
```

```
## [1] 0
```

Response: All the metrics are providing satisying results.