Time Series Analysis 1

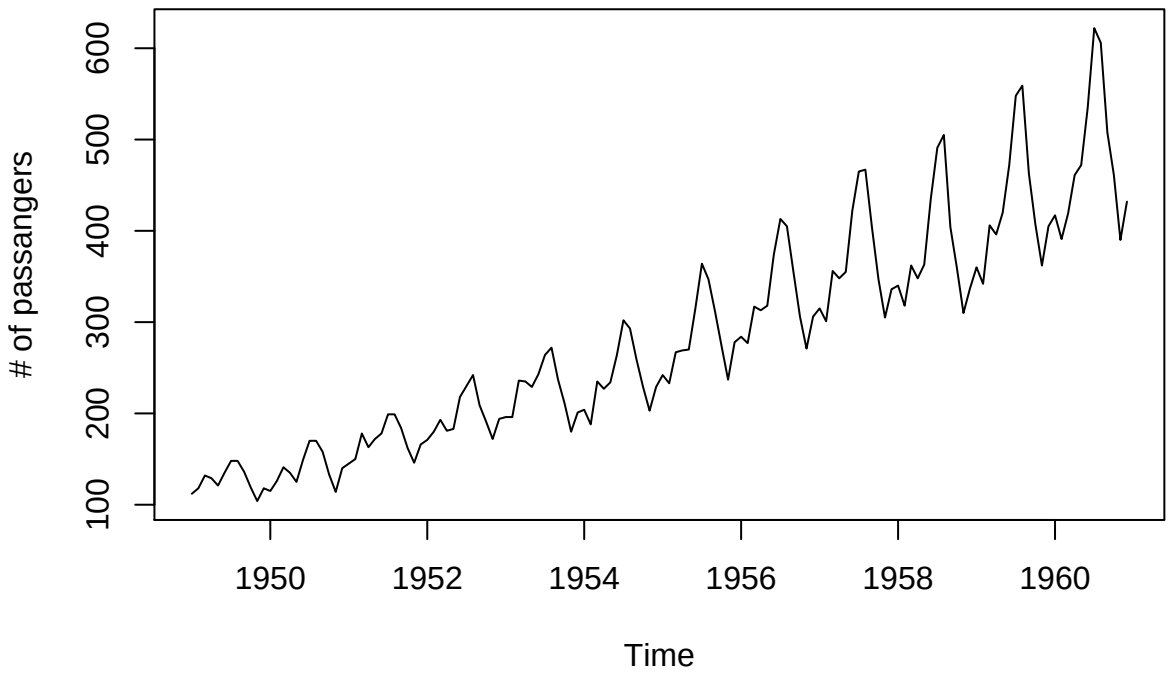
Trend and Seasonality Estimation Example 1

Time Series Analysis
Zhe Zheng

1.Import data and draw a plot of time series

```
rm(list=ls())
library(TSA)
library(accelerometry)
library(mgcv)
options(digits=3)
```

```
data = read.csv("AirPassengers.csv",header=T) #from 1949 Jan
temp = as.vector(data[,2])
temp = ts(temp,start=1949,frequency=12)
ts.plot(temp,ylab="# of passangers")
```



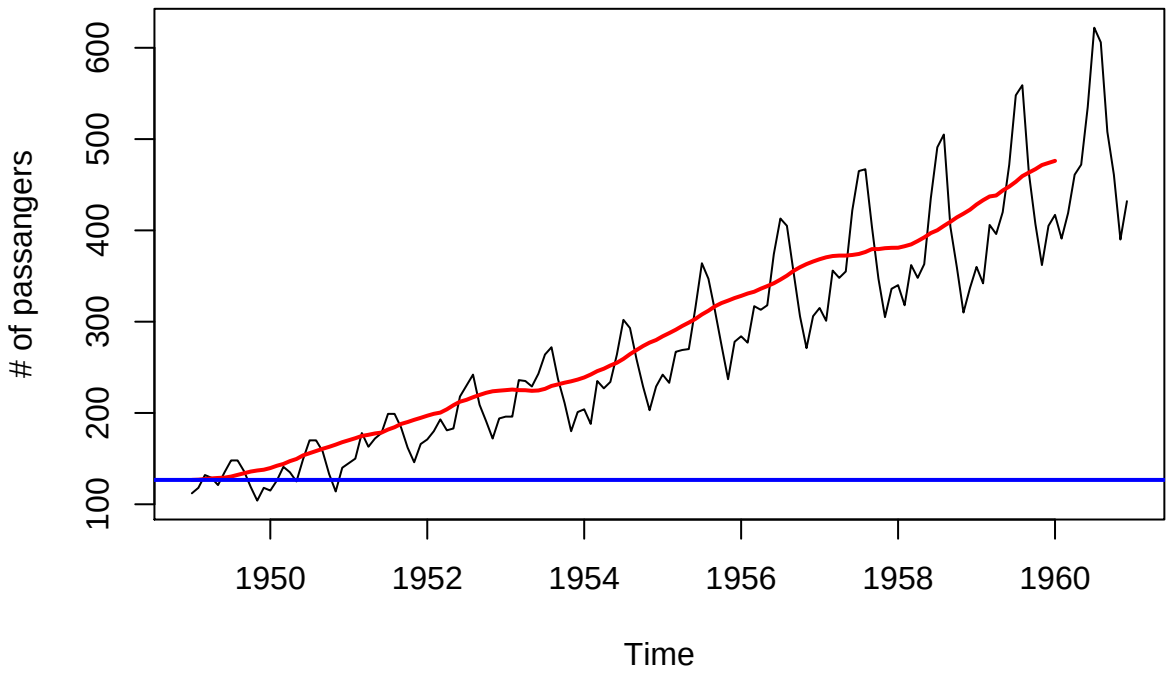
2.TREND ESTIMATION

Is there a trend in the monthly totals of passengers?

```
time.pts = c(1:length(temp))
time.pts = c(time.pts - min(time.pts))/max(time.pts)

## Fit a moving average
mav.fit = movingaves(x = temp, window = 12) #forward avg and round it to integer. Can also use function

temp.fit.mav = ts(mav.fit,start=1949,frequency=12)
ts.plot(temp,ylab="# of passangers",ylim=range( c(temp, temp) ))
lines(temp.fit.mav,lwd=2,col="red")
abline(temp.fit.mav[1],0,lwd=2,col="blue")
```

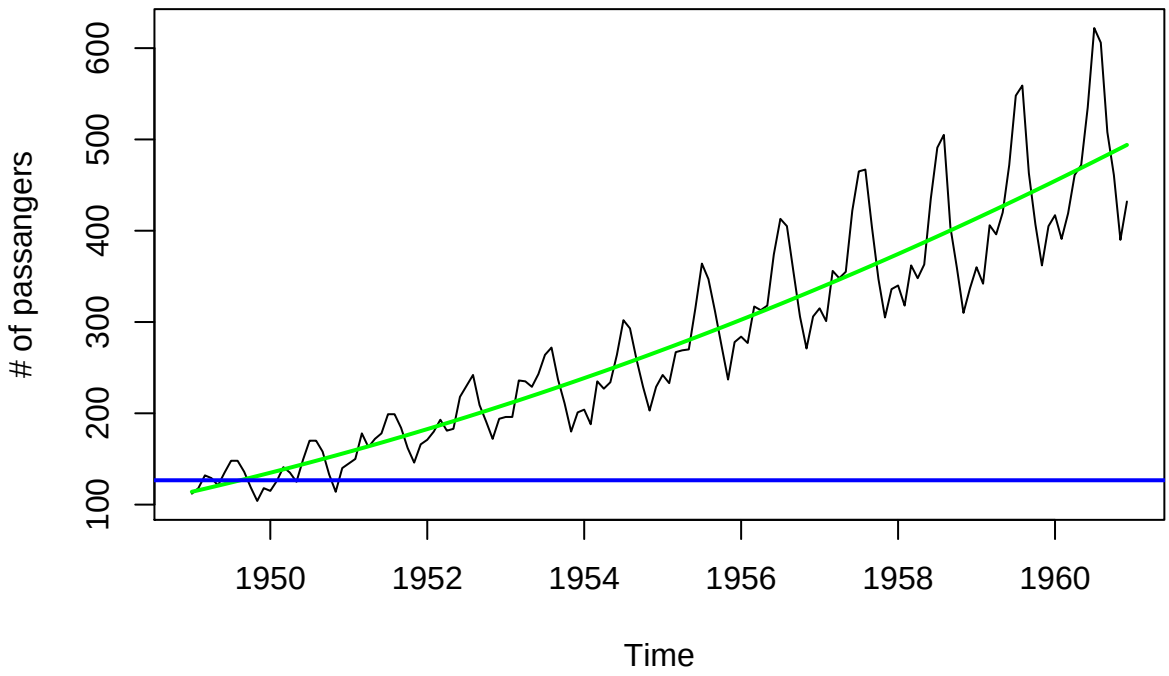


Fit a parametric quadraric polynomial, e.g. $y = a * t + b * t^2$

```
x1 = time.pts
x2 = time.pts^2
lm.fit = lm(temp~x1+x2)
summary(lm.fit)
```

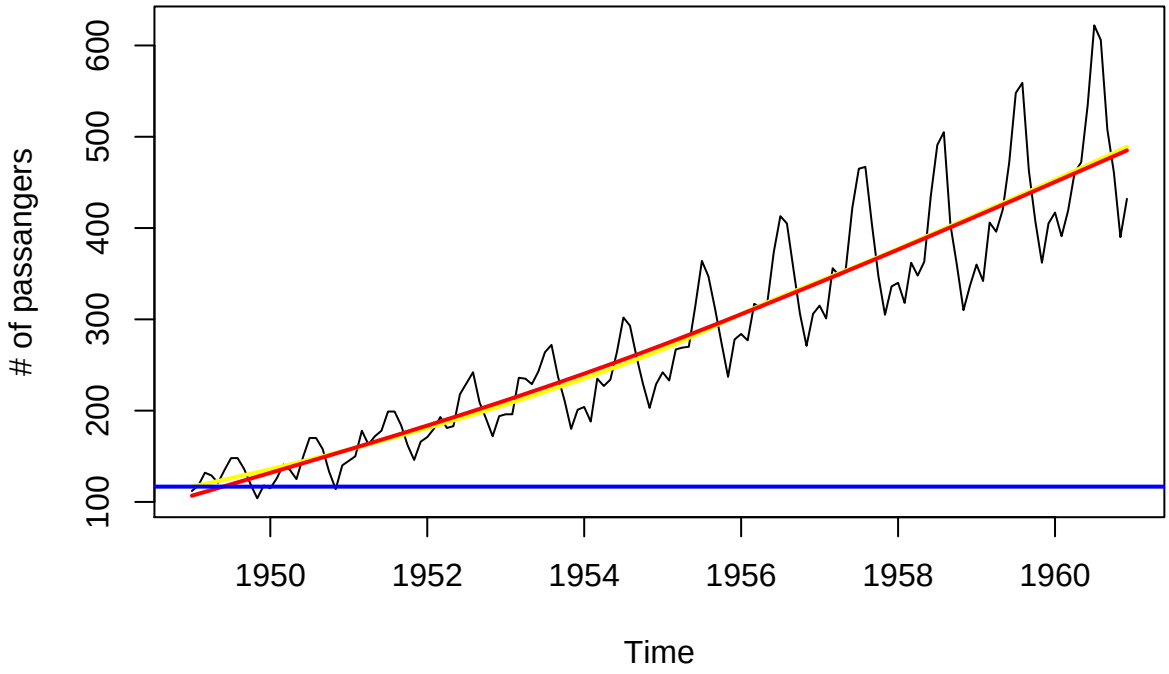
```
##
## Call:
## lm(formula = temp ~ x1 + x2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -100.35  -27.34   -7.44   21.60  146.12
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    114.0       11.1   10.30  < 2e-16 ***
## x1             238.3       51.5    4.63  8.4e-06 ***
## x2             145.3       50.2    2.89   0.0044 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 44.9 on 141 degrees of freedom
## Multiple R-squared:  0.862,  Adjusted R-squared:  0.86
## F-statistic:  440 on 2 and 141 DF,  p-value: <2e-16
```

```
temp.fit.lm = ts(fitted(lm.fit),start=1949,frequency=12)
ts.plot(temp,ylab="# of passangers")
lines(temp.fit.lm,lwd=2,col="green")
abline(temp.fit.mav[1],0,lwd=2,col="blue")
```



Fit a trend using non-parametric regression

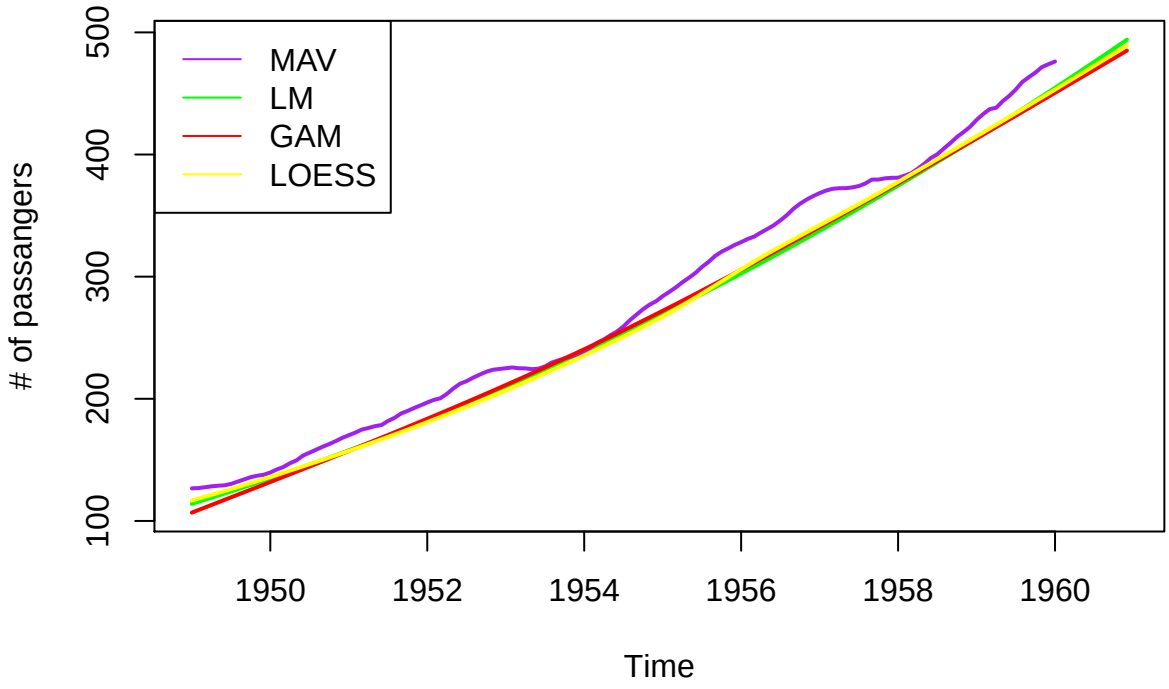
```
## Local Polynomial Trend Estimation
loc.fit = loess(temp~time.pts)
temp.fit.loc = ts(fitted(loc.fit),start=1949,frequency=12)
## Splines Trend Estimation
#library(mgcv)
gam.fit = gam(temp~s(time.pts))
temp.fit.gam = ts(fitted(gam.fit),start=1949,frequency=12)
## Is there a trend?
ts.plot(temp,ylab="# of passangers")
lines(temp.fit.loc,lwd=2,col="yellow")
lines(temp.fit.gam,lwd=2,col="red")
abline(temp.fit.loc[1],0,lwd=2,col="blue")
```



Compare all estimated trends, [1]moving average(MAV) [2]parametric quadraric polynomial(LM) [3] Splines Trend Estimation(GAM) [4]Local Polynomial Trend Estimation(LOESS)

```
all.val = c(temp.fit.mav,temp.fit.lm,temp.fit.gam,temp.fit.loc)
ylim= c(min(all.val),max(all.val))
ts.plot(temp.fit.lm,lwd=2,col="green",ylim=ylim,ylab="# of passangers")
```

```
lines(temp.fit.mav,lwd=2,col="purple")
lines(temp.fit.gam,lwd=2,col="red")
lines(temp.fit.loc,lwd=2,col="yellow")
legend(x= "topleft", y=0.92,legend=c("MAV","LM","GAM","LOESS"),lty = 1,col=c("purple","green","red","ye
```



3.SEASONALITY ESTIMATION

```
## Estimate seasonality using ANOVA approach
month = season(temp)    #depends on frequency of time series
## model with intercept
model1 = lm(temp~month)
summary(model1)
```

```
##
## Call:
## lm(formula = temp ~ month)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -203.3  -93.5  -17.0   87.2  270.7
##
## Coefficients:
```

```
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      241.75      34.08    7.09  7.2e-11 ***
## monthFebruary     -6.75      48.20   -0.14    0.889
## monthMarch        28.42      48.20    0.59    0.556
## monthApril        25.33      48.20    0.53    0.600
## monthMay          30.08      48.20    0.62    0.534
## monthJune         69.92      48.20    1.45    0.149
## monthJuly        109.58      48.20    2.27    0.025 *
## monthAugust      109.33      48.20    2.27    0.025 *
## monthSeptember    60.67      48.20    1.26    0.210
## monthOctober      24.83      48.20    0.52    0.607
## monthNovember     -8.92      48.20   -0.19    0.854
## monthDecember     20.08      48.20    0.42    0.678
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 118 on 132 degrees of freedom
## Multiple R-squared:  0.106,  Adjusted R-squared:  0.0316
## F-statistic: 1.42 on 11 and 132 DF,  p-value: 0.169
```

```r
## All seasonal mean effects (model without intercept)
model2 = lm(temp~month-1)
summary(model2)
```

```
##
## Call:
## lm(formula = temp ~ month - 1)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -203.3  -93.5  -17.0   87.2  270.7
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## monthJanuary      241.8       34.1    7.09  7.2e-11 ***
## monthFebruary     235.0       34.1    6.90  2.0e-10 ***
## monthMarch        270.2       34.1    7.93  8.3e-13 ***
## monthApril        267.1       34.1    7.84  1.4e-12 ***
## monthMay          271.8       34.1    7.98  6.3e-13 ***
## monthJune         311.7       34.1    9.15  9.4e-16 ***
## monthJuly         351.3       34.1   10.31  < 2e-16 ***
## monthAugust       351.1       34.1   10.30  < 2e-16 ***
## monthSeptember    302.4       34.1    8.87  4.4e-15 ***
## monthOctober      266.6       34.1    7.82  1.5e-12 ***
## monthNovember     232.8       34.1    6.83  2.8e-10 ***
## monthDecember     261.8       34.1    7.68  3.1e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 118 on 132 degrees of freedom
## Multiple R-squared:  0.862,  Adjusted R-squared:  0.85
## F-statistic:   69 on 12 and 132 DF,  p-value: <2e-16
```

Estimate seasonality using cos-sin model

```
har=harmonic(temp,1)
model3=lm(temp~har)
summary(model3)
```

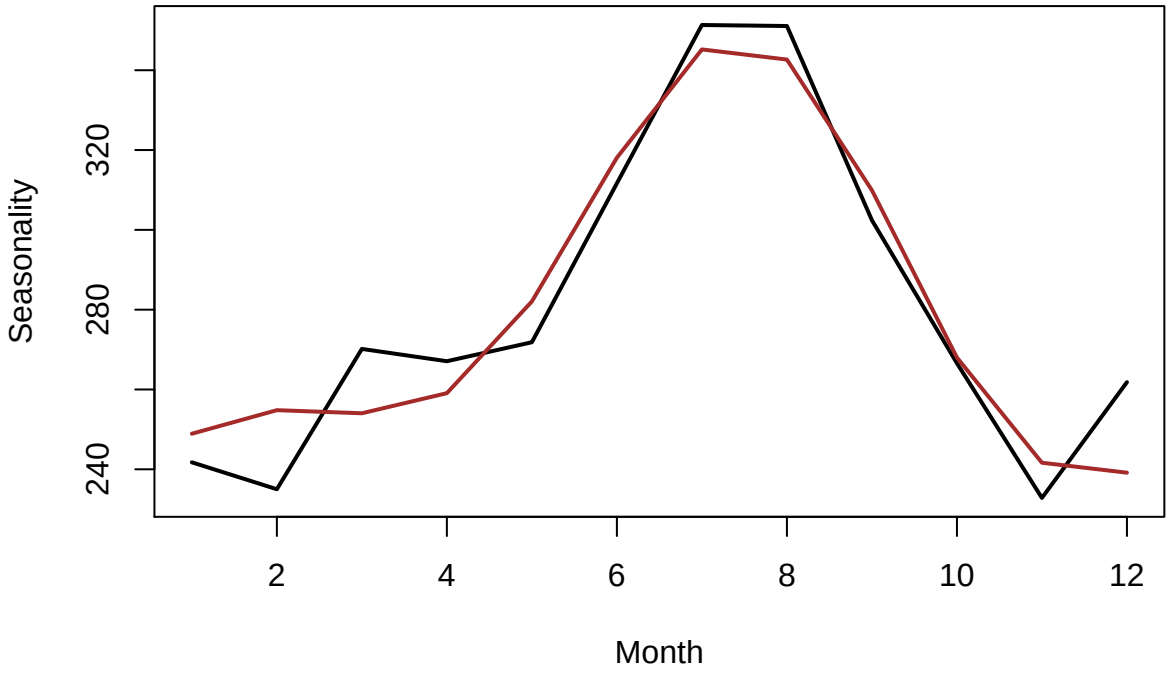
```
##
## Call:
## lm(formula = temp ~ har)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -184.8  -95.7  -16.9   95.3  293.6
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      280.30       9.65   29.05  < 2e-16 ***
## harcos(2*pi*t)   -48.15      13.64   -3.53  0.00056 ***
## harsin(2*pi*t)    -4.46      13.64   -0.33  0.74405
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 116 on 141 degrees of freedom
## Multiple R-squared:  0.0818, Adjusted R-squared:  0.0688
## F-statistic: 6.28 on 2 and 141 DF,  p-value: 0.00244
```

```
har2=harmonic(temp,2)
model4=lm(temp~har2)
summary(model4)
```

```
##
## Call:
## lm(formula = temp ~ har2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -197.2  -99.1  -15.1   91.0  276.8
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       280.30       9.64   29.08  < 2e-16 ***
## har2cos(2*pi*t)   -48.15      13.63   -3.53  0.00056 ***
## har2cos(4*pi*t)    16.76      13.63    1.23  0.22086
## har2sin(2*pi*t)    -4.46      13.63   -0.33  0.74385
## har2sin(4*pi*t)    11.62      13.63    0.85  0.39548
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 116 on 139 degrees of freedom
## Multiple R-squared:  0.0964, Adjusted R-squared:  0.0704
## F-statistic: 3.71 on 4 and 139 DF,  p-value: 0.00673
```

Compare Seasonality Estimates

```
## Seasonal Means Model
st1 = coef(model2)
## Cos-Sin Model
st2 = fitted(model4)[1:12]
plot(1:12,st1,lwd=2,type="l",xlab="Month",ylab="Seasonality")
lines(1:12,st2,lwd=2, col="brown")
```



4. TREND AND SEASONALITY ESTIMATION AT ONCE

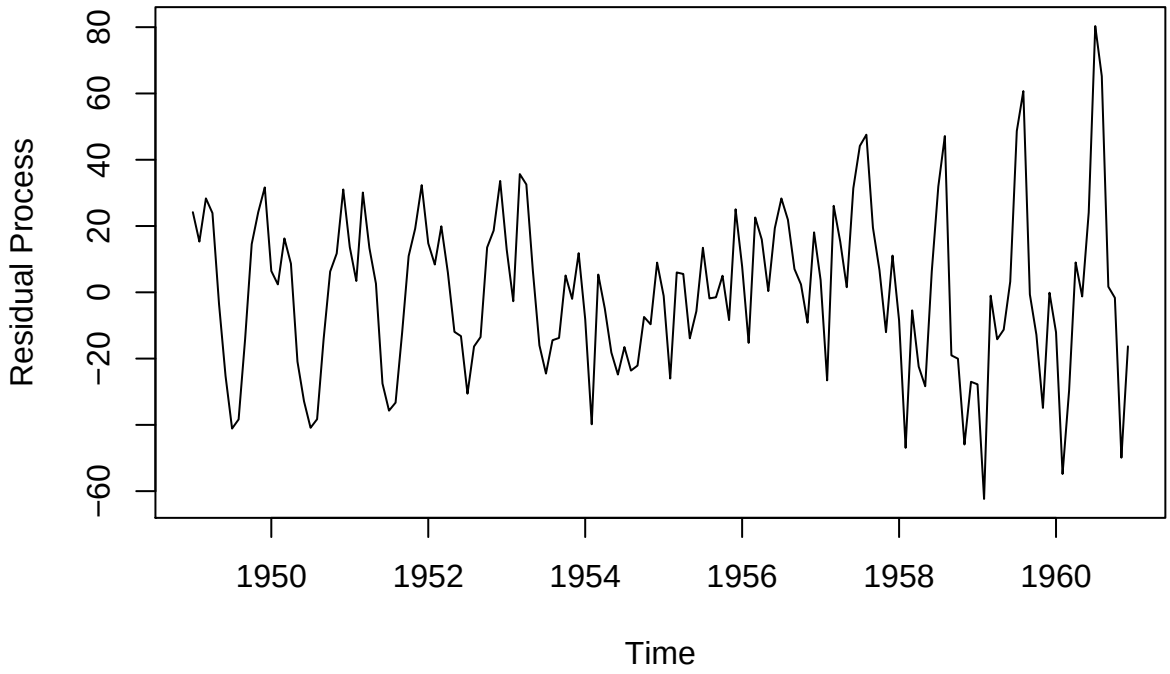
Using linear regression to fit a parametric model for both trend and seasonality

```
x1 = time.pts
x2 = time.pts^2
har2=harmonic(temp,2)
lm.fit = lm(temp~x1+x2+har2)
summary(lm.fit)
```

```
##
## Call:
```

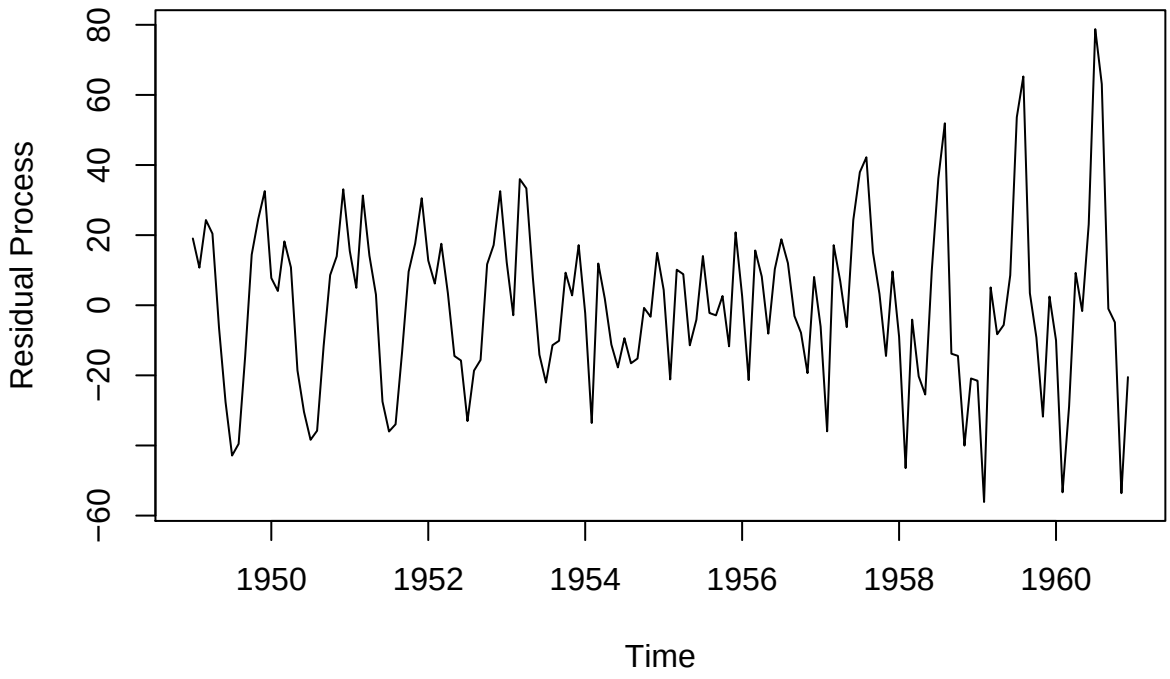
```
## lm(formula = temp ~ x1 + x2 + har2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -62.35 -15.45   0.95  14.91  80.34
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       114.01       6.20   18.39  < 2e-16 ***
## x1                236.56      28.82    8.21  1.5e-13 ***
## x2                148.04      28.09    5.27  5.2e-07 ***
## har2cos(2*pi*t)   -45.59       2.96  -15.40  < 2e-16 ***
## har2cos(4*pi*t)    19.41       2.96    6.55  1.0e-09 ***
## har2sin(2*pi*t)     5.50       2.97    1.86    0.066 .
## har2sin(4*pi*t)    16.25       2.96    5.49  1.9e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25.1 on 137 degrees of freedom
## Multiple R-squared:  0.958,  Adjusted R-squared:  0.956
## F-statistic:  521 on 6 and 137 DF,  p-value: <2e-16
```

```r
dif.fit.lm = ts((temp-fitted(lm.fit)),start=1949,frequency=12)
ts.plot(dif.fit.lm,ylab="Residual Process")
```



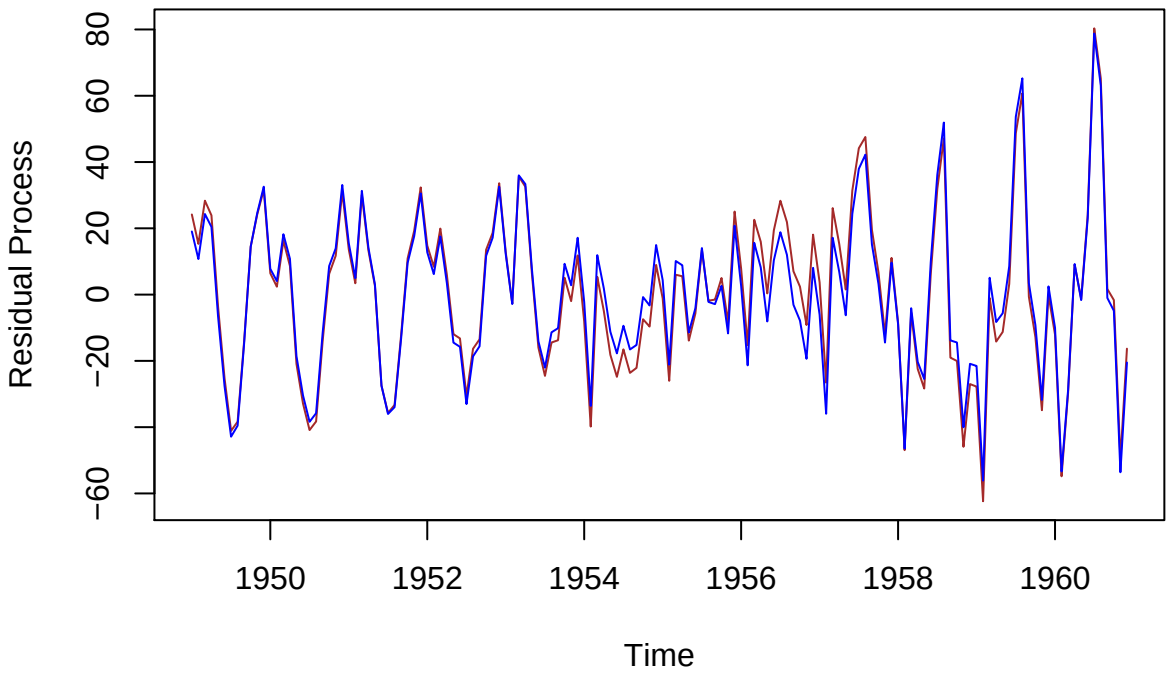
Fit a non-parametric model for trend and linear model for seasonality

```
gam.fit = gam(temp~s(time.pts)+har2)
dif.fit.gam = ts((temp-fitted(gam.fit)),start=1949,frequency=12)
ts.plot(dif.fit.gam,ylab="Residual Process")
```

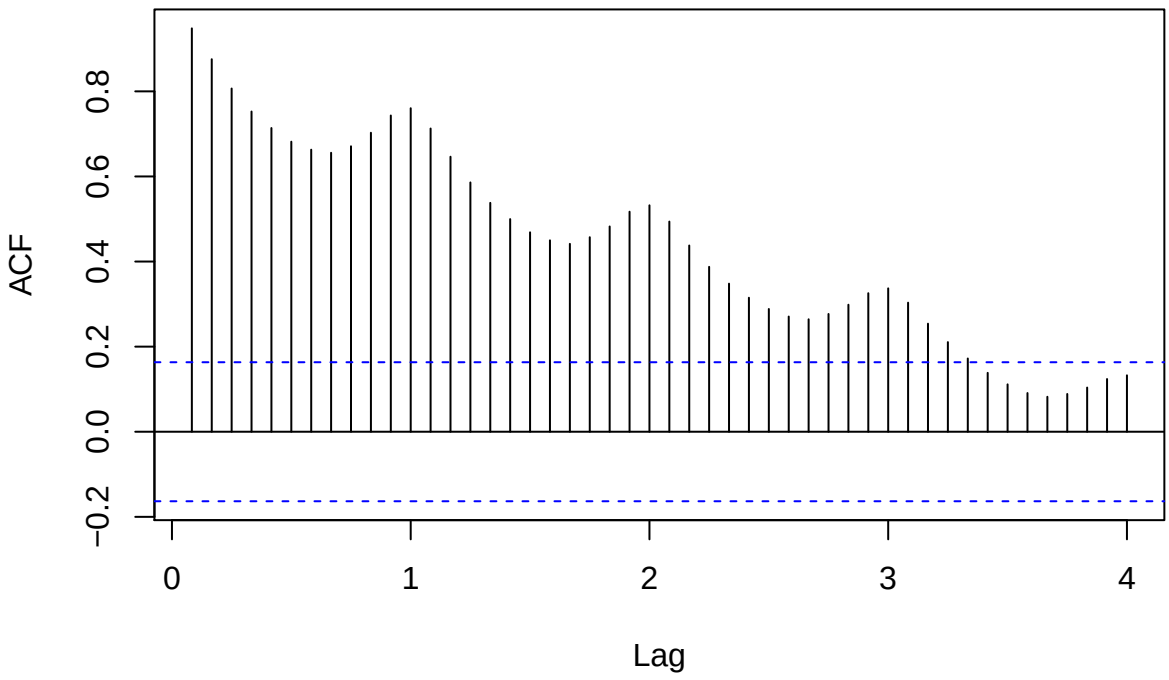


Compare approaches by plotting residuals and looking at acf functions. -> find out the residual process is not yet a stationary process, we can further do some modeling and it would be ARMA(p,q) in next chaper

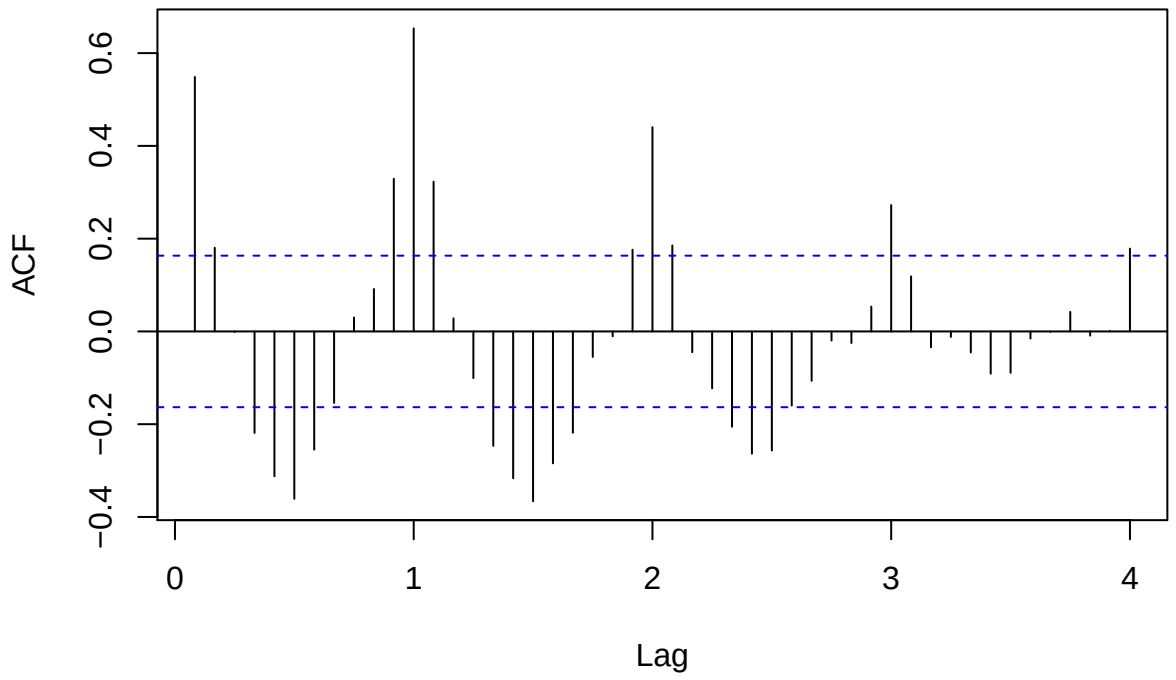
```
ts.plot(dif.fit.lm,ylab="Residual Process",col="brown")
lines(dif.fit.gam,col="blue")
```



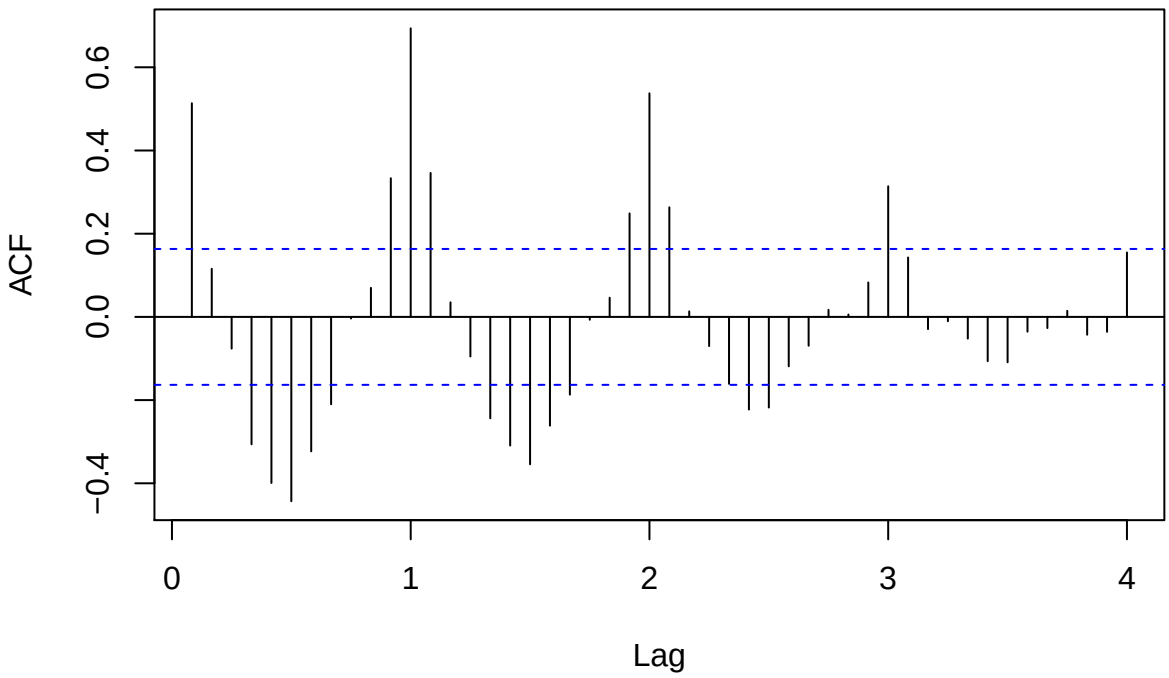

```
acf(temp,lag.max=12*4,main="")
```




```
acf(dif.fit.lm,lag.max=12*4,main="")
```




```
acf(dif.fit.gam,lag.max=12*4,main="")
```


ACF
0.6
0.4
0.2
0.0
−0.4
0
1
2
3
4
Lag