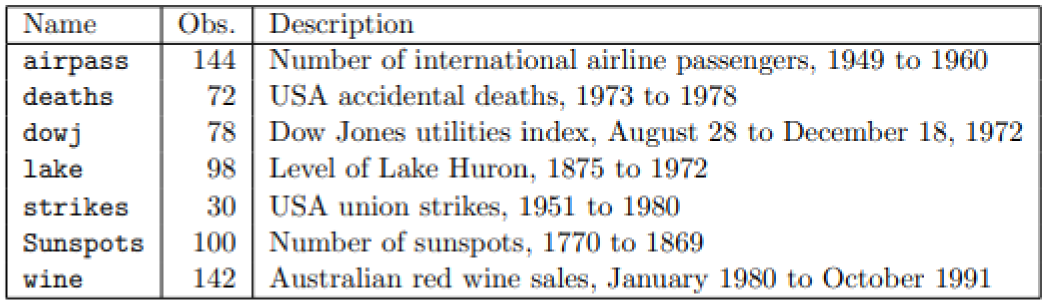
CH1

2021711914 김동영

```
setwd("G:/    /gdrive/work/  /    ")
```

```
library(ggplot2)
library(dplyr)
library(magrittr)
```

데이터셋 불러오는법은 `library(itsmr)` 쓰시면 됩니다. 패키지에 들어있는 데이터목록은 다음과 같고 `data()`함수로 불러오는건 아니고 그냥 패키지에서 바로 부르는 형태입니다.



Name	Obs.	Description
airpass	144	Number of international airline passengers, 1949 to 1960
deaths	72	USA accidental deaths, 1973 to 1978
dowj	78	Dow Jones utilities index, August 28 to December 18, 1972
lake	98	Level of Lake Huron, 1875 to 1972
strikes	30	USA union strikes, 1951 to 1980
Sunspots	100	Number of sunspots, 1770 to 1869
wine	142	Australian red wine sales, January 1980 to October 1991

Figure 1-1

```
dat = itsmr::wine
# 1980        ts data
dat = ts(dat,start=c(1980,1), frequency=12) / 1000
dat # ts
```

```
##        Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
## 1980 0.464 0.675 0.703 0.887 1.139 1.077 1.318 1.260 1.120 0.963 0.996 0.960
## 1981 0.530 0.883 0.894 1.045 1.199 1.287 1.565 1.577 1.076 0.918 1.008 1.063
## 1982 0.544 0.635 0.804 0.980 1.018 1.064 1.404 1.286 1.104 0.999 0.996 1.015
## 1983 0.615 0.722 0.832 0.977 1.270 1.437 1.520 1.708 1.151 0.934 1.159 1.209
## 1984 0.699 0.830 0.996 1.124 1.458 1.270 1.753 2.258 1.208 1.241 1.265 1.828
## 1985 0.809 0.997 1.164 1.205 1.538 1.513 1.378 2.083 1.357 1.536 1.526 1.376
## 1986 0.779 1.005 1.193 1.522 1.539 1.546 2.116 2.326 1.596 1.356 1.553 1.613
## 1987 0.814 1.150 1.225 1.691 1.759 1.754 2.100 2.062 2.012 1.897 1.964 2.186
## 1988 0.966 1.549 1.538 1.612 2.078 2.137 2.907 2.249 1.883 1.739 1.828 1.868
## 1989 1.138 1.430 1.809 1.763 2.200 2.067 2.503 2.141 2.103 1.972 2.181 2.344
## 1990 0.970 1.199 1.718 1.683 2.025 2.051 2.439 2.353 2.230 1.852 2.147 2.286
## 1991 1.007 1.665 1.642 1.525 1.838 1.892 2.920 2.572 2.617 2.047
```

```
plot(dat, xlab='', ylab='(thousands)')
points(dat, pch=0)
```

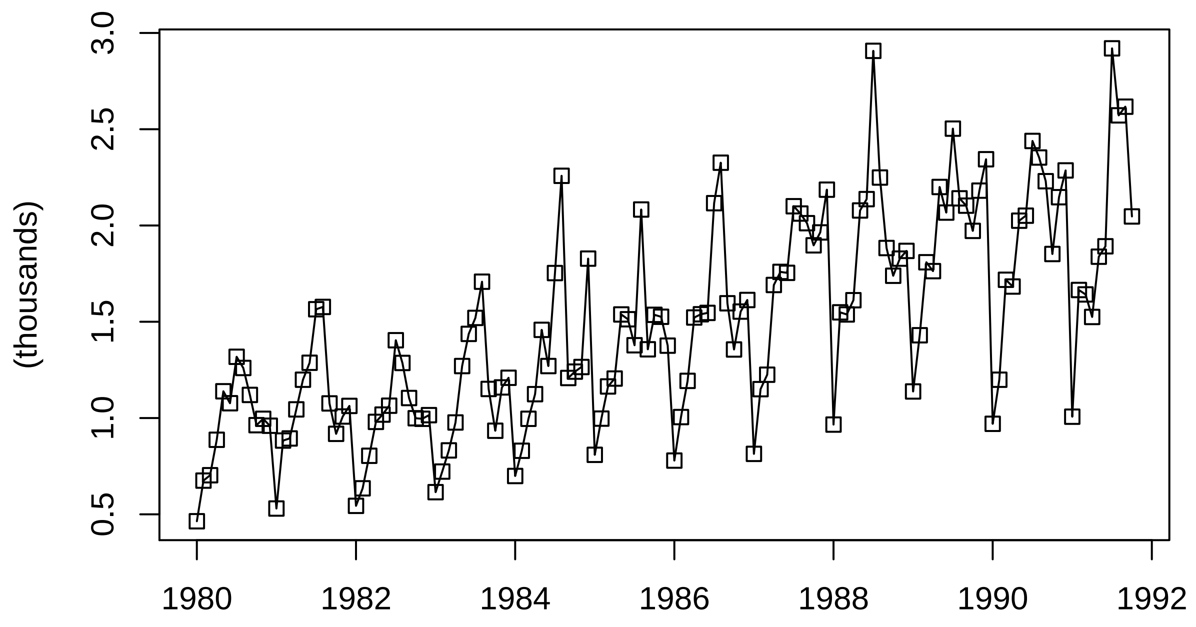


Figure 1-2

```
set.seed(1)
dat = sample(c(-1,1), (1995-1933+1), replace=T)
dat[sample(1:length(dat),3)] = NA
dat = ts(dat,start=1933, end=1995)
plot(dat, type='l', xlab='',ylab='', ylim=c(-2,2))
points(dat, pch=0);abline(h=0)
```

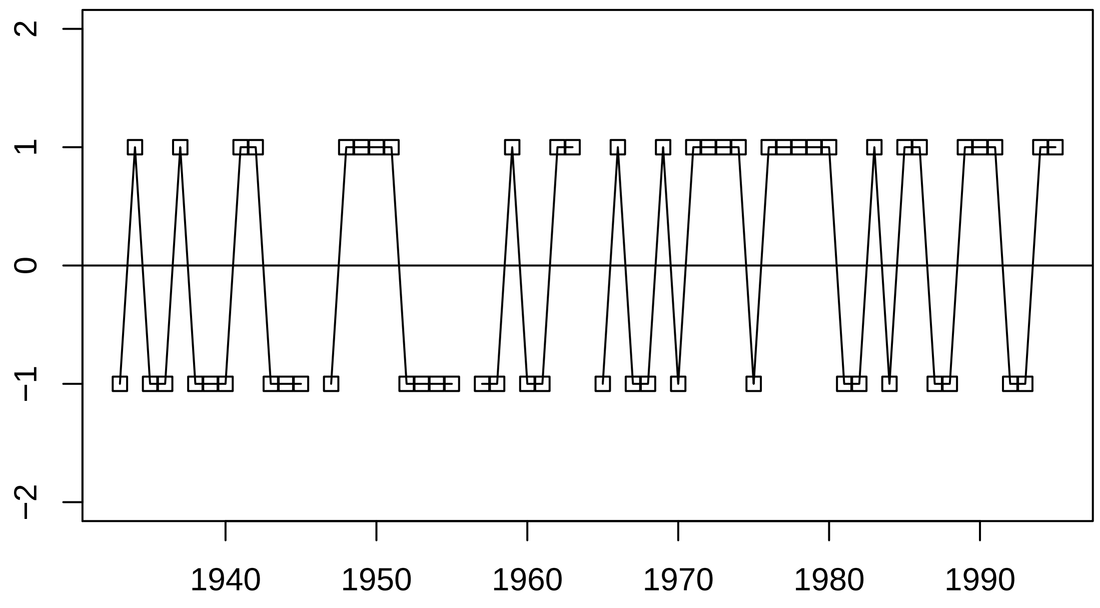


Figure 1-3

```
dat = itsmr::deaths
dat = ts(dat, start=c(1973,1), frequency=12) / 1000
plot(dat, xlab='', ylab='(thousands)')
points(dat, pch=0)
```

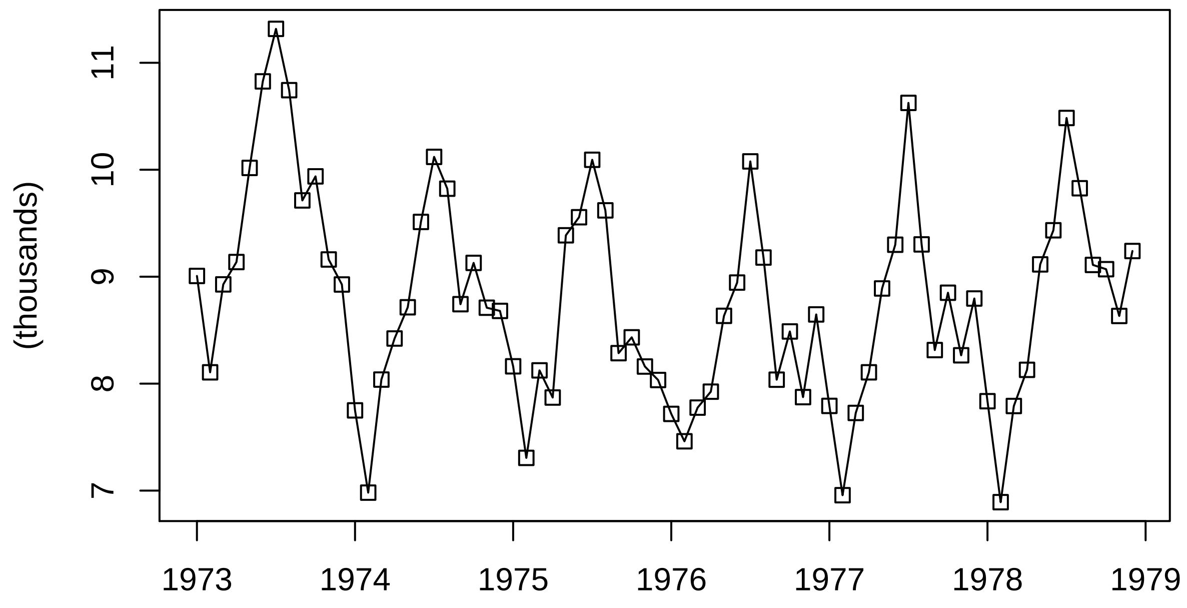


Figure 1-5

```
n = (1990-1790) / 10 + 1 # 21
dat = ts((1:n)^2, start=1790, end=1990, frequency=0.1)
dat = dat + rnorm(n, sd=5)
plot(dat, xlab='',ylab='(Millions)')
points(dat, pch=0)
```

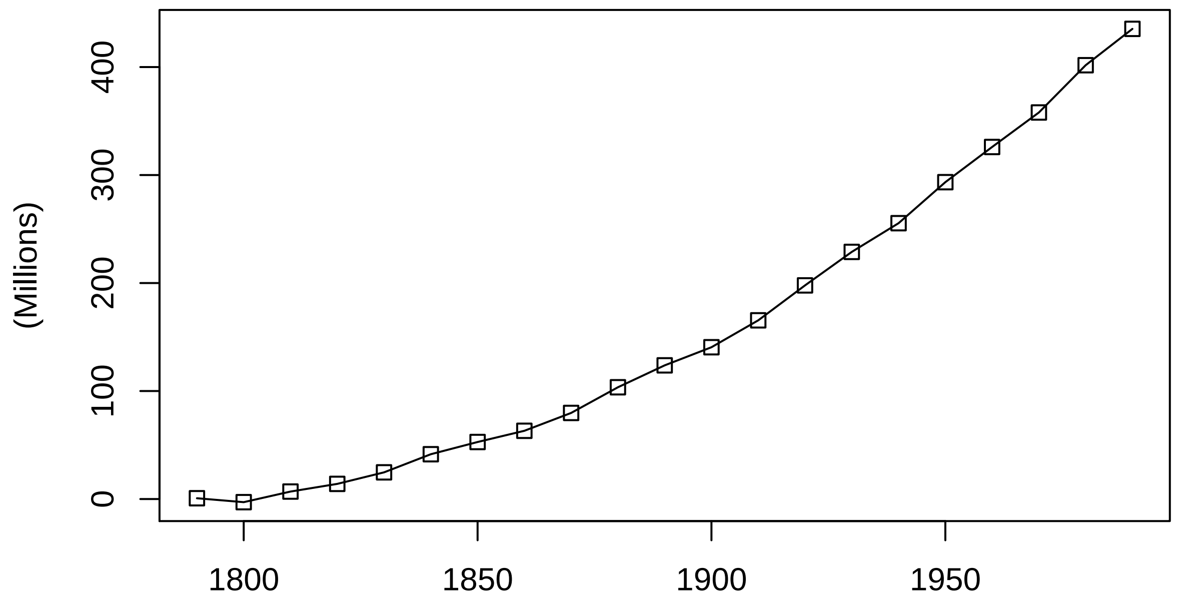
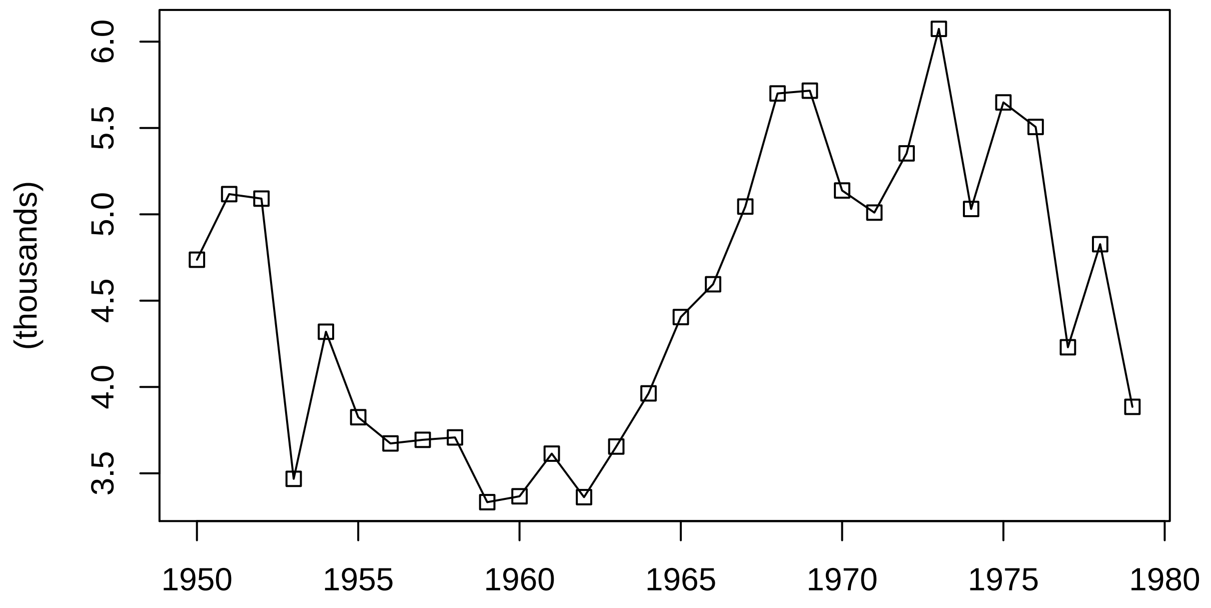


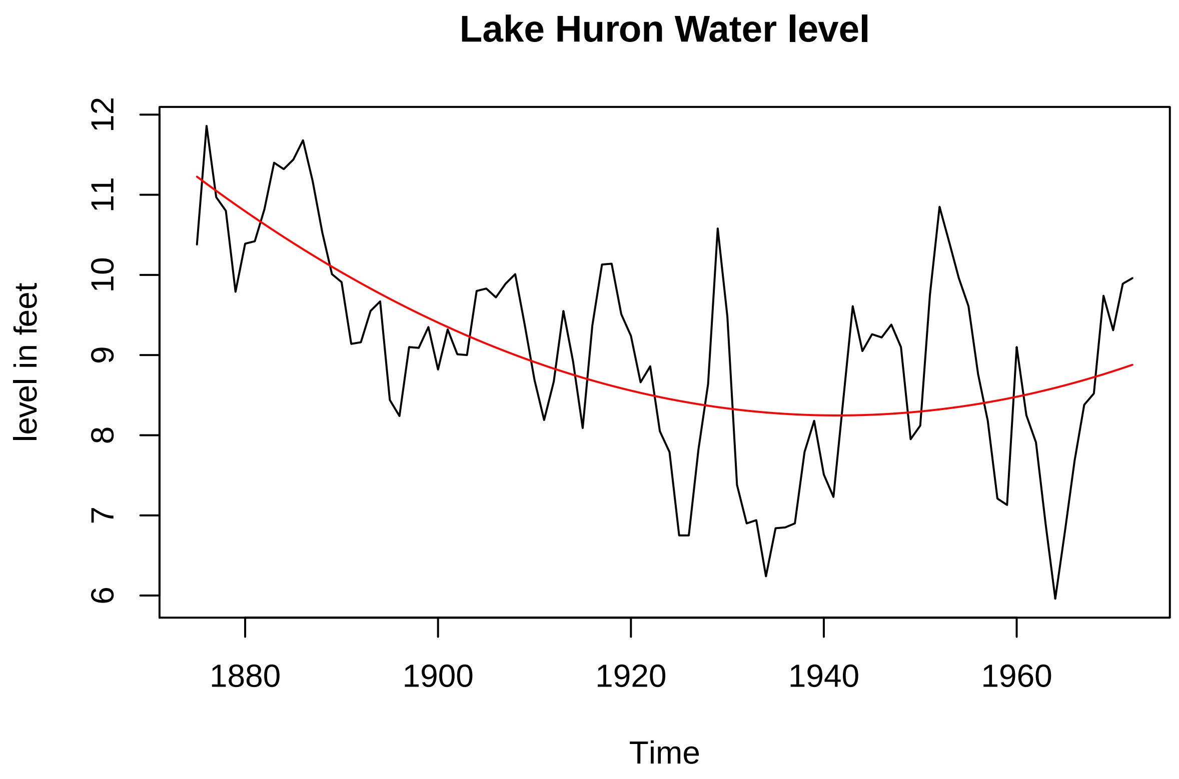
Figure 1-6

```
dat = itsmr::strikes
dat = ts(dat, start=1950) / 1000
plot(dat, xlab='', ylab='(thousands)')
points(dat, pch=0)
```



(p21) Polynomial regression - OLS

```r
dat = itsmr::lake
dat = ts(dat, start=1875)
x = 1875:1972
x2 = x^2
lm_fit = lm(dat ~ 1 + x + x2)
plot(dat, ylab='level in feet')
title('Lake Huron Water level')
lines(x, lm_fit$fitted.values, col='red')
```

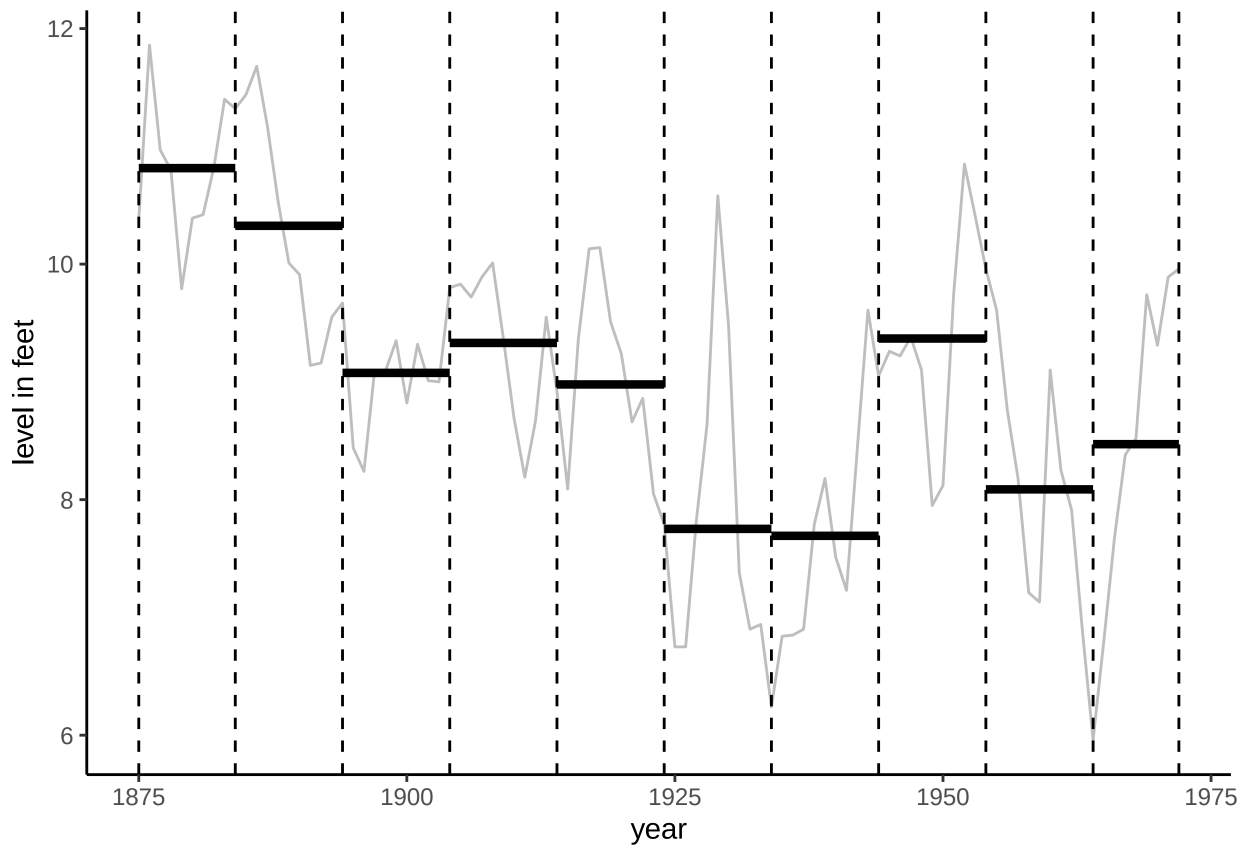


(p23) Estimating trend only - Smoothing

```
dat = itsmr::lake
dat = data.frame(n=1:length(dat), y=dat, t=x)
head(dat)
```

```
##   n     y    t
## 1 1 10.38 1875
## 2 2 11.86 1876
## 3 3 10.97 1877
## 4 4 10.80 1878
## 5 5  9.79 1879
## 6 6 10.39 1880
```

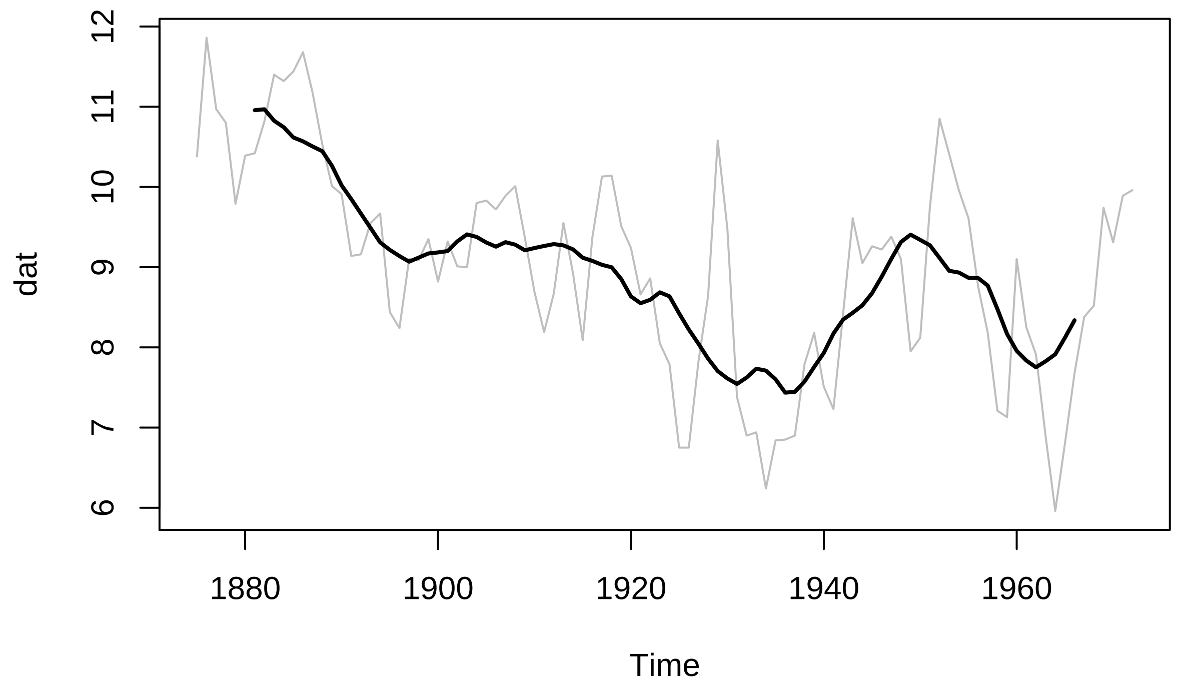
```
dat %<>%
  mutate(group = (n-1) %/% 10)


last_dat = dat %>% group_by(group) %>%
  slice(n()) %>%
  ungroup()

dat = last_dat %>%
  mutate(group = group + 1) %>%
  slice(1:9) %>% bind_rows(dat) %>%
  arrange(group,t) %>%
  group_by(group) %>%
  mutate(local_mean = mean(y))

cut_point = c(1875,last_dat$t)


ggplot(data=dat)+
  geom_line(aes(x=t,y=y),color='grey')+
  geom_line(aes(x=t,y=local_mean,group=group),size=1.5)+
  geom_vline(xintercept=cut_point, linetype=2)+
  labs(x='year', y='level in feet')+
  theme_classic()
```



(p24) Smoothing1 - Moving Average filter

```
local_average = function(x,q){
  #           for
  Wt = c()
  for (i in (q+1):(100-q)){
    w = mean(x[(i-q):(i+q)])
    Wt = c(Wt,w)
  }
  Wt = c(rep(NA,q),Wt,rep(NA,q))
  return(Wt)
}
```

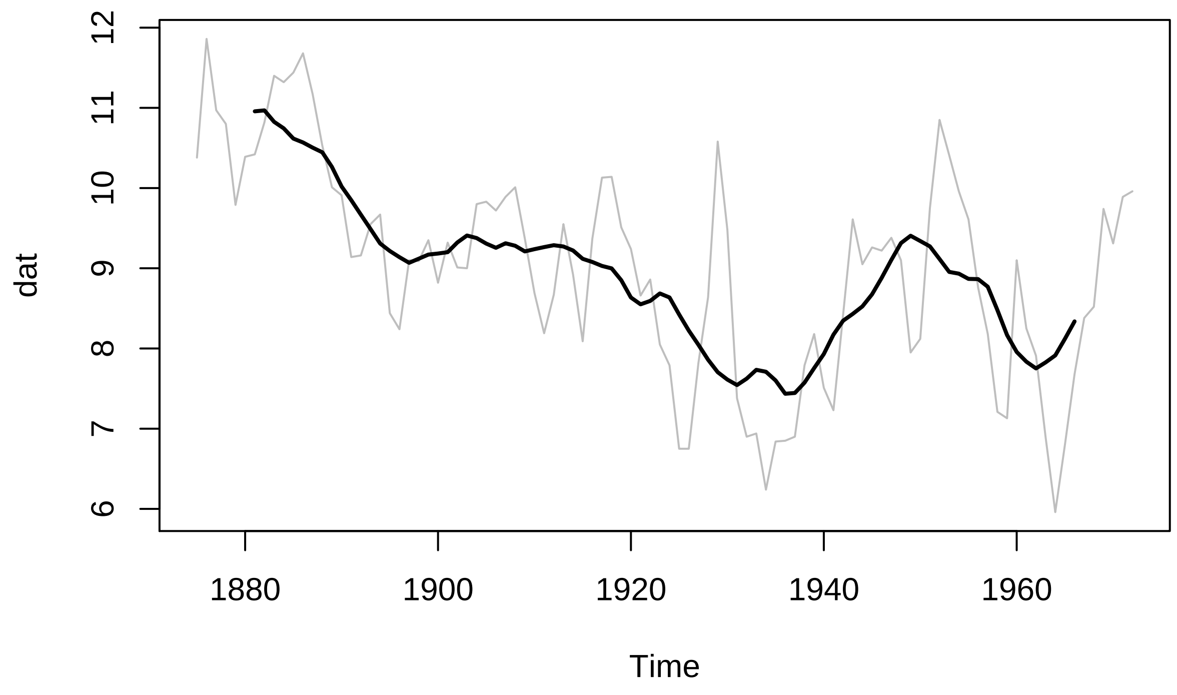
```
dat = itsmr::lake
dat = ts(dat, start=1875)
dat_smooth = local_average(dat,6) %>% ts(start=1875)

plot(dat, col='grey')
lines(dat_smooth, lwd=2)
```



```
#
plot(dat, col='grey')
dat_smooth = forecast::ma(dat, 13)
```

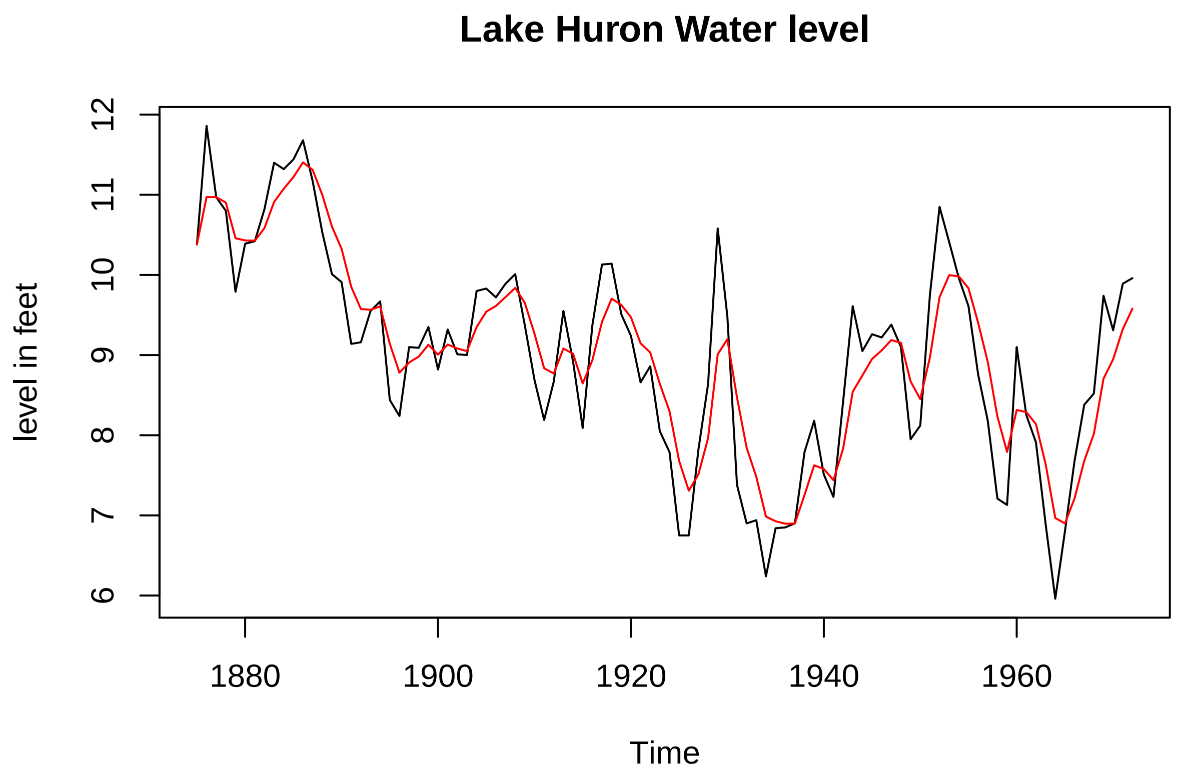
```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
lines(dat_smooth, lwd=2)
```



(p29) Exponential smoothing

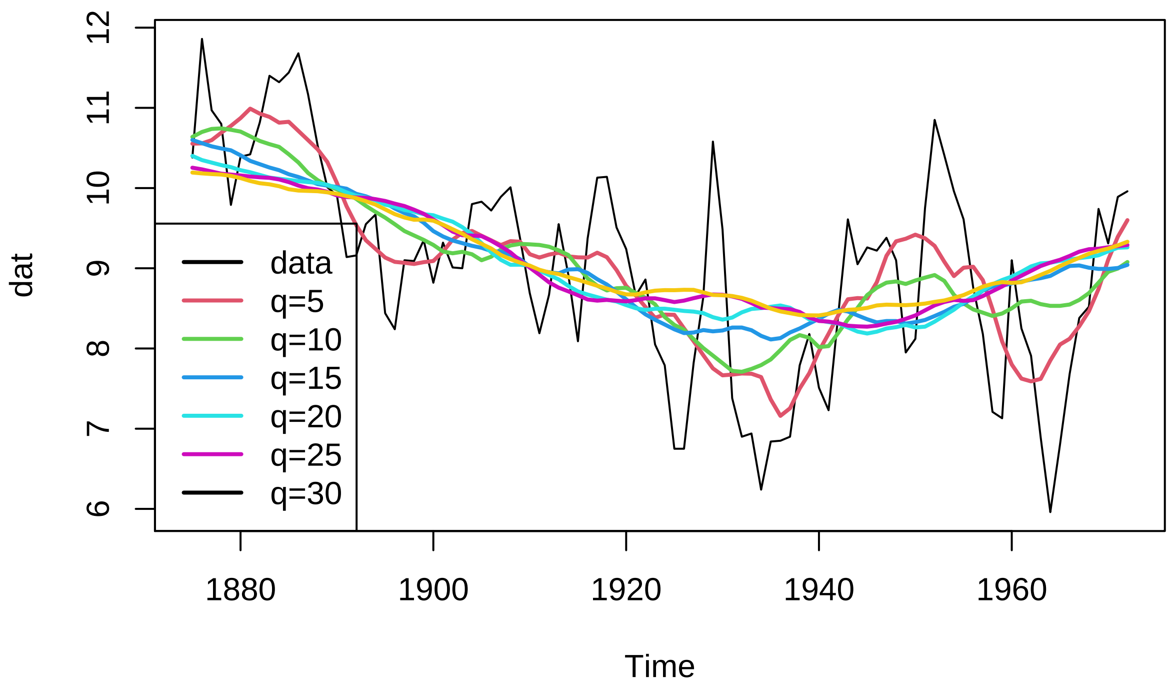
```
ex4 = itsmr::smooth.exp(dat, 0.4) %>% ts(start=1875)
plot(dat, ylab='level in feet')
lines(ex4, col='red');title('Lake Huron Water level')
```



(p30) Weakness of Smoothing - bandwidth selection

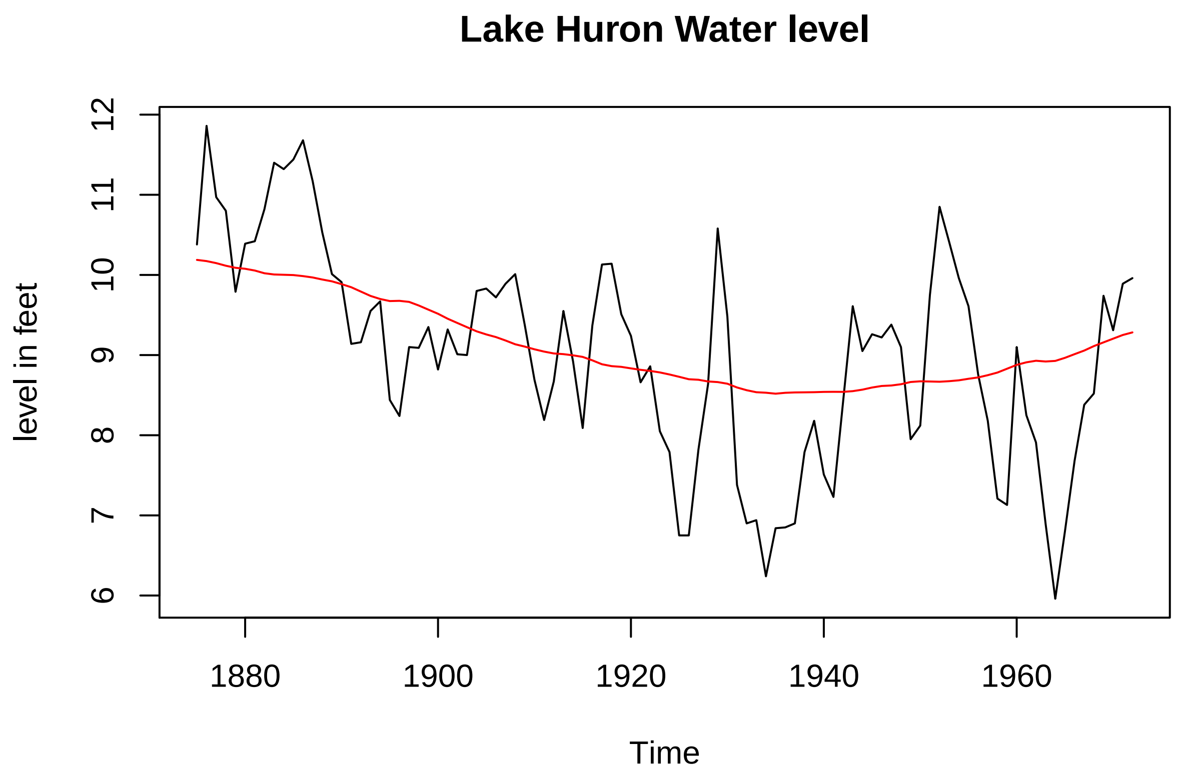
원래는 앞뒤로 NA로 주는게 맞는데 꽉 채운 데이터 쓸려고 smooth.ma 사용

```
plot(dat)
q_vec = seq(5,30,5)

for (q in q_vec){
  dat_smooth= itsmr::smooth.ma(dat, q) %>% ts(start=1875)
  lines(dat_smooth, col=q / 5 + 1, lwd=2)
}

legend('bottomleft',
       c('data',paste0('q=',q_vec)),
       col=1:6,lwd=2)
```



(p32) Smoothing - MA Bandwidth selection

```
plot(dat, ylab='level in feet')
dat_smooth = itsmr::smooth.ma(dat, q=33) %>% ts(start=1875)
lines(dat_smooth,col='red')
title('Lake Huron Water level')
```

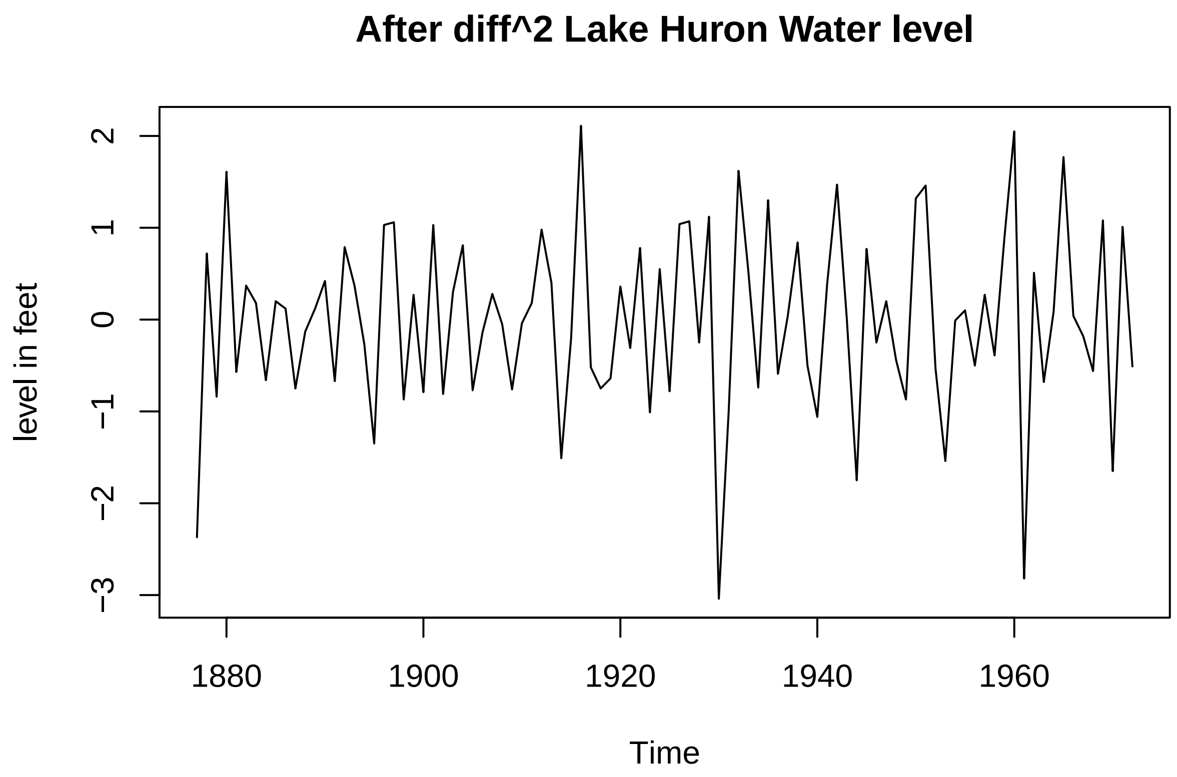


(p34) Estimating trend only - Differncing

```
dat
```

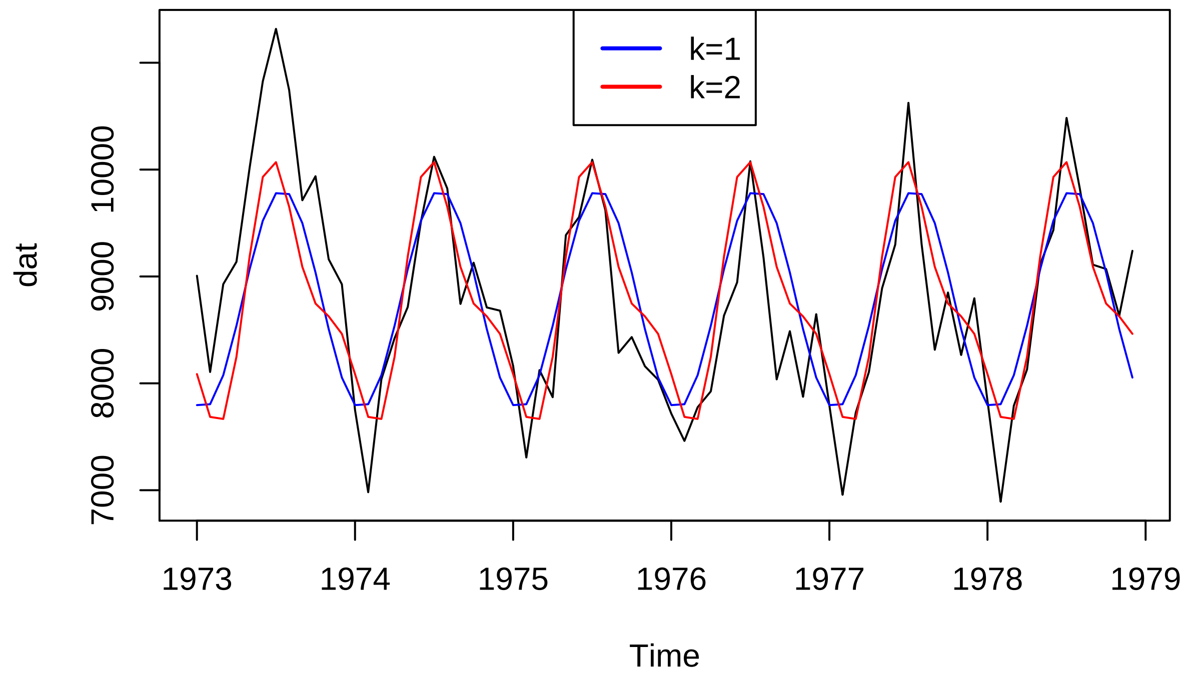
```
## Time Series:
## Start = 1875
## End = 1972
## Frequency = 1
##  [1] 10.38 11.86 10.97 10.80  9.79 10.39 10.42 10.82 11.40 11.32 11.44 11.68
## [13] 11.17 10.53 10.01  9.91  9.14  9.16  9.55  9.67  8.44  8.24  9.10  9.09
## [25]  9.35  8.82  9.32  9.01  9.00  9.80  9.83  9.72  9.89 10.01  9.37  8.69
## [37]  8.19  8.67  9.55  8.92  8.09  9.37 10.13 10.14  9.51  9.24  8.66  8.86
## [49]  8.05  7.79  6.75  6.75  7.82  8.64 10.58  9.48  7.38  6.90  6.94  6.24
## [61]  6.84  6.85  6.90  7.79  8.18  7.51  7.23  8.42  9.61  9.05  9.26  9.22
## [73]  9.38  9.10  7.95  8.12  9.75 10.85 10.41  9.96  9.61  8.76  8.18  7.21
## [85]  7.13  9.10  8.25  7.91  6.89  5.96  6.80  7.68  8.38  8.52  9.74  9.31
## [97]  9.89  9.96
```

```
y = diff(diff(dat))
plot(y, ylab='level in feet');title('After diff^2 Lake Huron Water level')
```



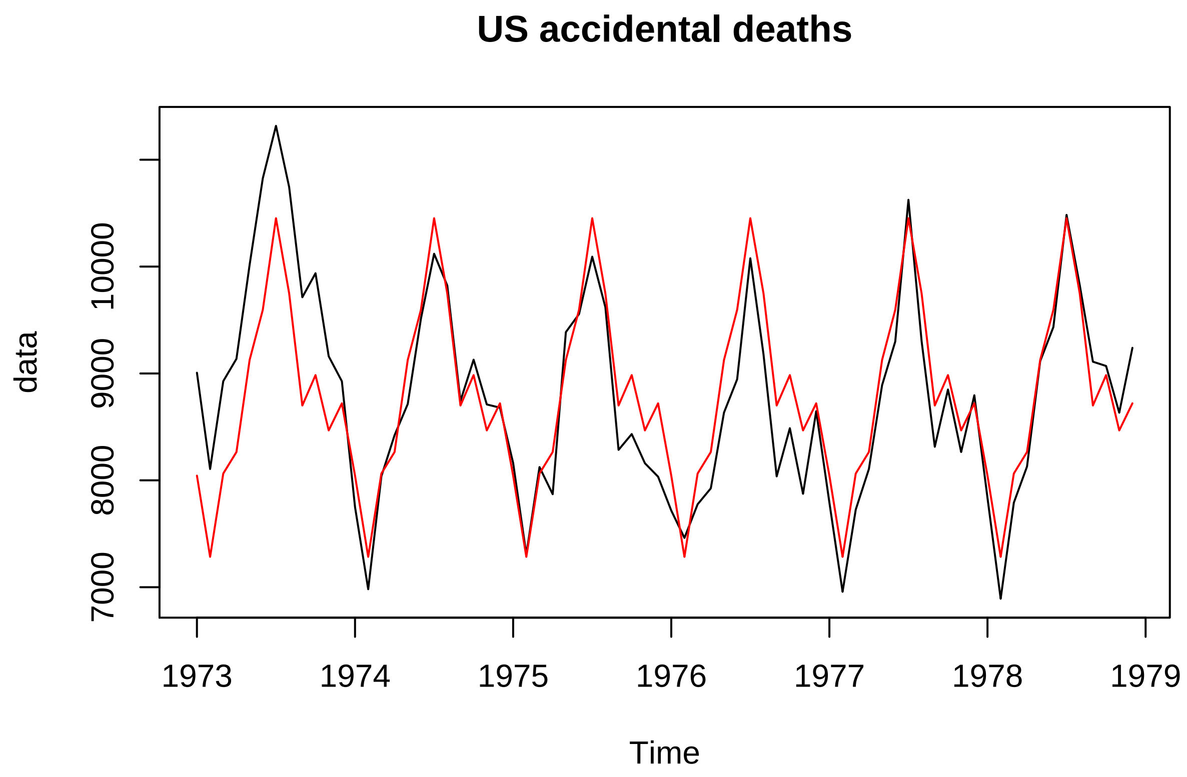
(p39) Harmonic regression

```
dat = itsmr::deaths %>% ts(start=c(1973,1),frequency=12)
n = length(dat)
t=1:n; f1 = 6; f2 = 12;
costerm1 = cos(f1*2*pi/n*t); sinterm1 = sin(f1*2*pi/n*t);
costerm2 = cos(f2*2*pi/n*t); sinterm2 = sin(f2*2*pi/n*t);
plot(dat)

lm_fit1 = lm(dat ~ 1 + costerm1 + sinterm1)
lm_fit2 = lm(dat ~ 1 + costerm1 + sinterm1 + costerm2 + sinterm2)

dat_season1 = lm_fit1$fitted.values %>% ts(start=c(1973,1),frequency=12)
dat_season2 = lm_fit2$fitted.values %>% ts(start=c(1973,1),frequency=12)

lines(dat_season1, col='blue')
lines(dat_season2, col='red')
legend('top',
       paste0('k=',c(1,2)),
       col=c('blue','red'),lwd=2)
```



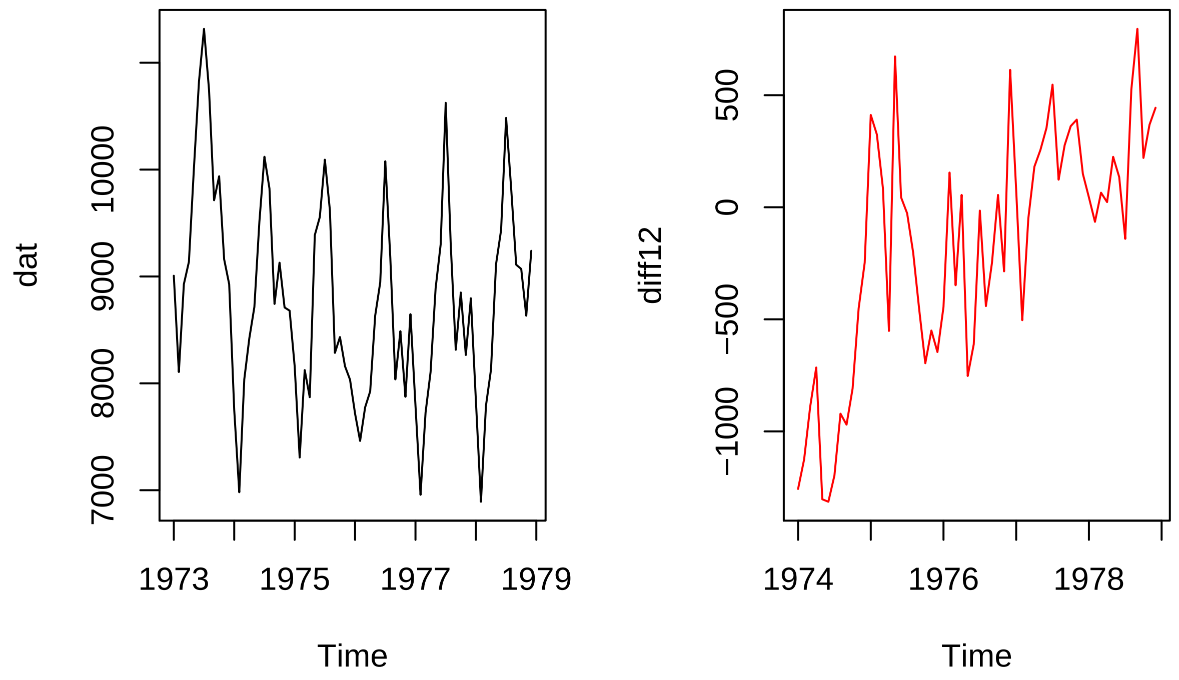
p(41) Seasonal smoothing

```
dat = itsmr::deaths %>% ts(start=c(1973,1),frequency=12)
dat_season = matrix(dat,ncol=12,byrow=T) %>%
  apply(2,mean) %>% rep(6) %>%
  ts(start=c(1973,1),frequency=12)

plot(dat, ylab='data')
lines(dat_season, col='red')
title('US accidental deaths')
```



(p43) Seasonal differencing

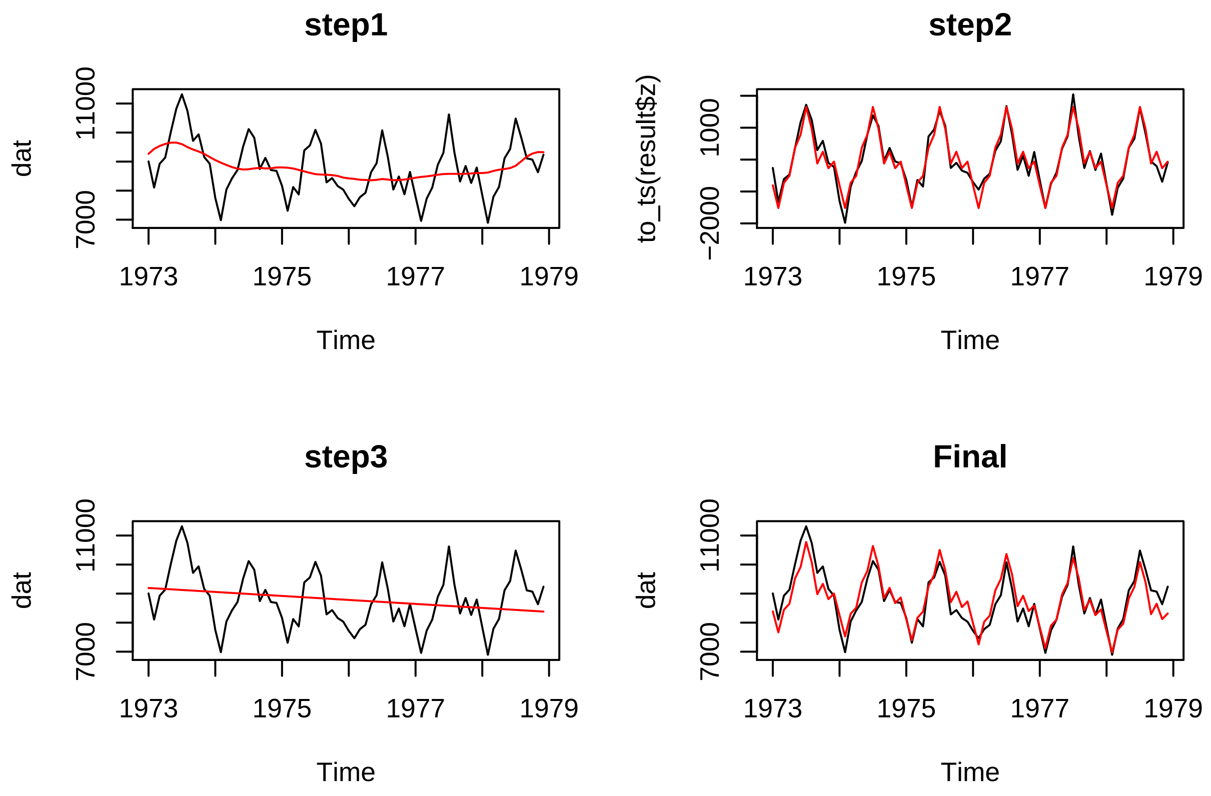
```
dat = itsmr::deaths %>% ts(start=c(1973,1),frequency=12)
diff12 = diff(dat, lag=12) %>%
  ts(start=c(1974,1), frequency=12)
par(mfrow=c(1,2))
plot(dat);plot(diff12, col='red')
```



(p50) Accidental deaths - classical decomposition

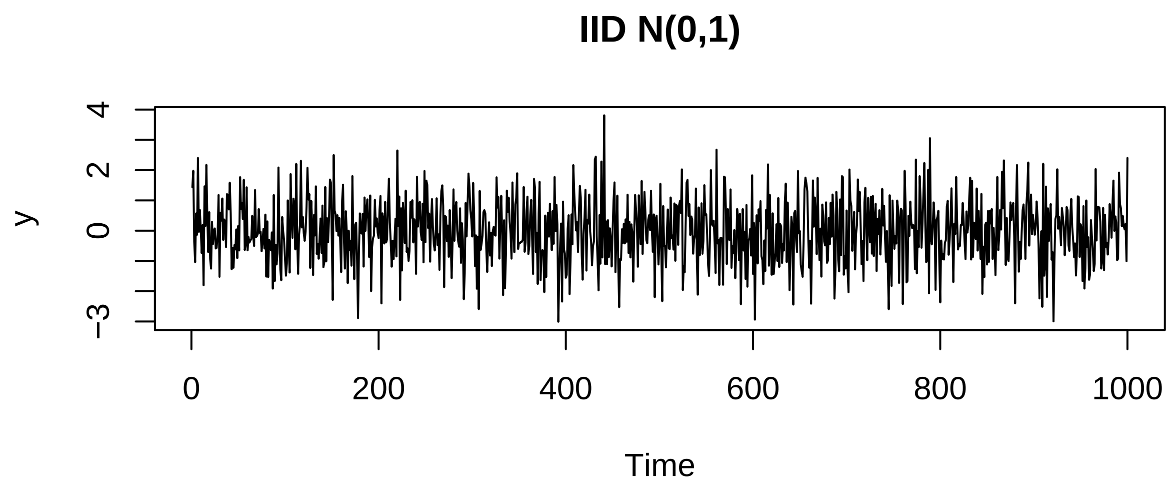
```
classical = function(data, d, order){
    n=length(data);
    # step 1
    q=ifelse(d%%2, (d-1)/2, d/2)
    x=c(rep(data[1], q), data, rep(data[n], q));
    if(d %%2 == 0){
    ff= c(.5, rep(1, 2*q-1), .5)/d;}
    if(d %%2 == 1){
    ff= rep(1, 2*q+1)/d;}
    xx = stats::filter(x, ff, method = c("convolution"))
    mhat = na.omit(xx);
    mhat = as.numeric(mhat);
    # step 2
    z = data - as.numeric(mhat);
    st = itsmr::season(z, d);
    # step 3 (regression)
    mnew = itsmr::trend(data-st, order);
    # step 4 (residuals)
    fit = mnew + st;
    resi = data - fit;
return(list(fit=fit, st=st, m=mnew, resi=resi, m1=mhat,
            z=z))
}
```

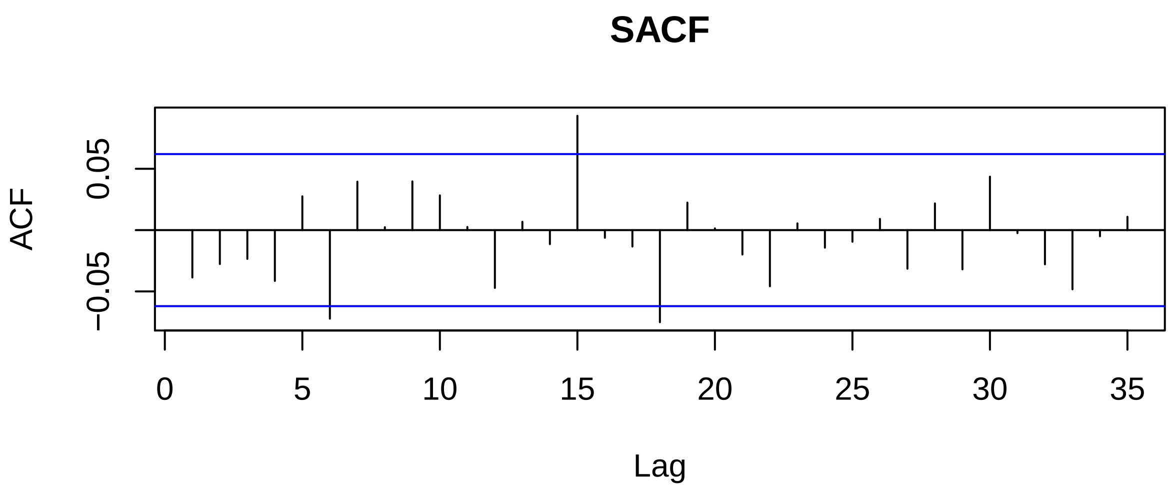
```r
to_ts = function(x) ts(x, start=c(1973,1),frequency=12)
dat = itsmr::deaths %>% ts(start=c(1973,1),frequency=12)
result= classical(dat, d=12, order=1)

par(mfrow=c(2,2))
plot(dat)
lines(to_ts(result$m1), col='red')
title('step1')

plot(to_ts(result$z))
lines(to_ts(result$st), col='red')
title('step2')

plot(dat)
lines(to_ts(result$m), col='red')
title('step3')

plot(dat)
lines(to_ts(result$fit), col='red')
title('Final')
```



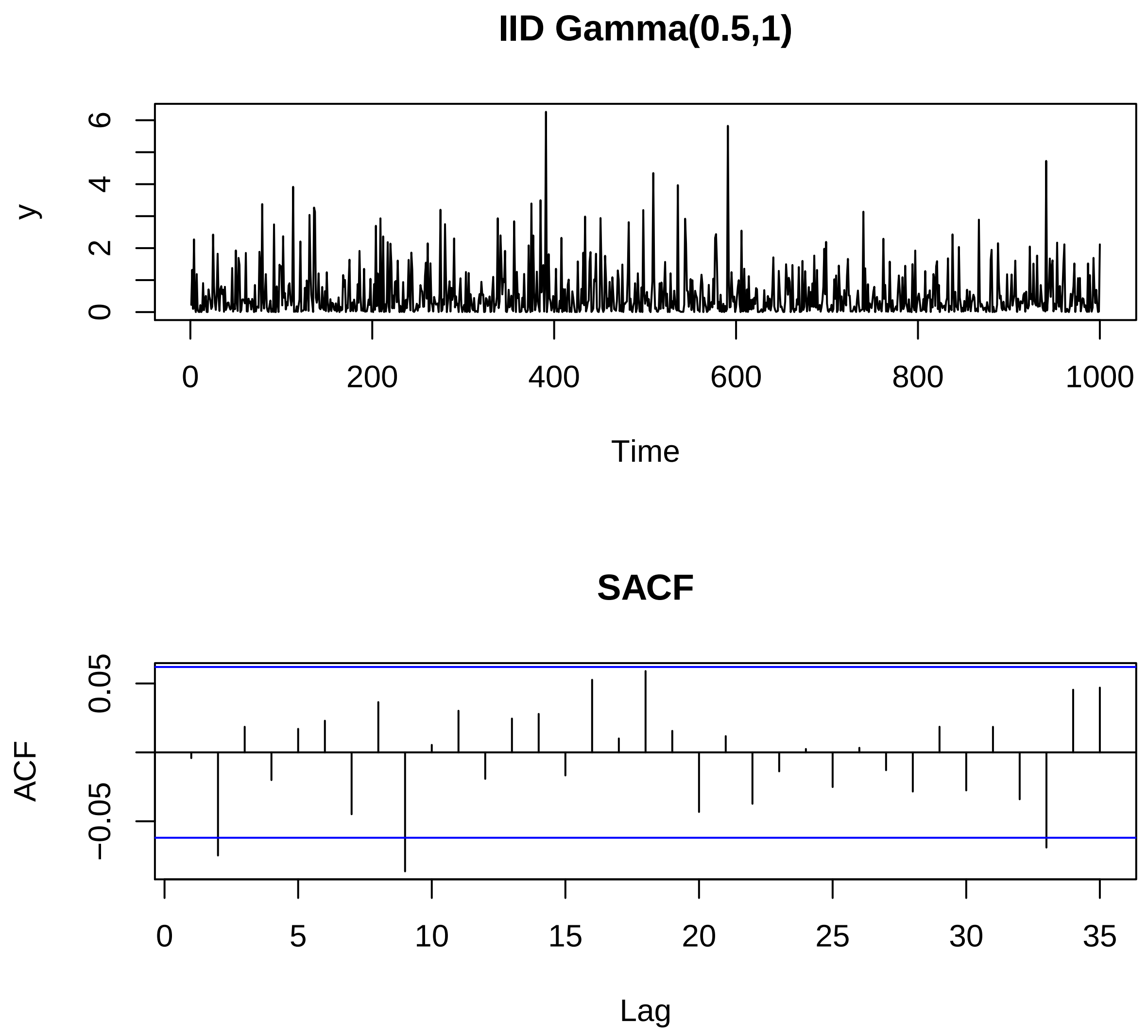
```r
acf2 <- function(data, lag){
  if(missing(lag)){ lag = 35;}

  thr=qnorm(1-.05/2, mean=0, sd=1/sqrt(length(data)));
  a1 = acf(data, lag, plot = FALSE);
  x = seq(1, lag, by=1);
  y = a1$acf[-1];
  plot(x, y, type="h", xlab = "Lag", ylab="ACF");
  abline(h=0, col = "black");
  abline(h = -thr, col="blue");
  abline(h = thr, col="blue");
  title("SACF");
}
data_and_acf = function(y, lag=35, plot_title='title', line=F){
  par(mfrow=c(2,1))
  plot(y, type='l',xlab='Time')
  title(plot_title)
  if (line == T){
    lines(1:1000, 1:1000 * 0.01 - 2, col='red')
  }
  acf2(y, lag=lag)
}
```

```r
y = rnorm(1000)
data_and_acf(y, plot_title='IID N(0,1)')
```

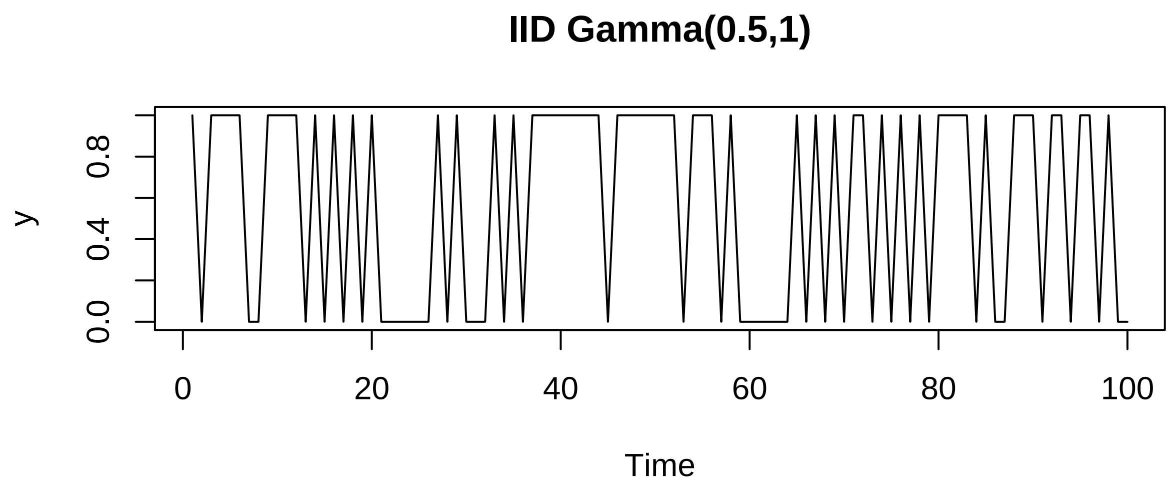


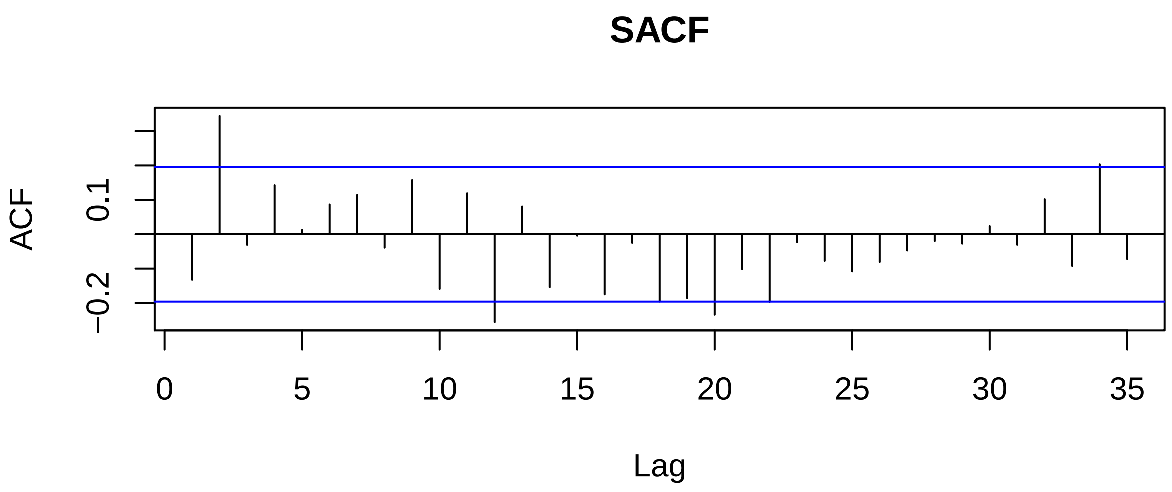


```r
y = rgamma(1000, 0.5, 1)
data_and_acf(y, plot_title = 'IID Gamma(0.5,1)')
```

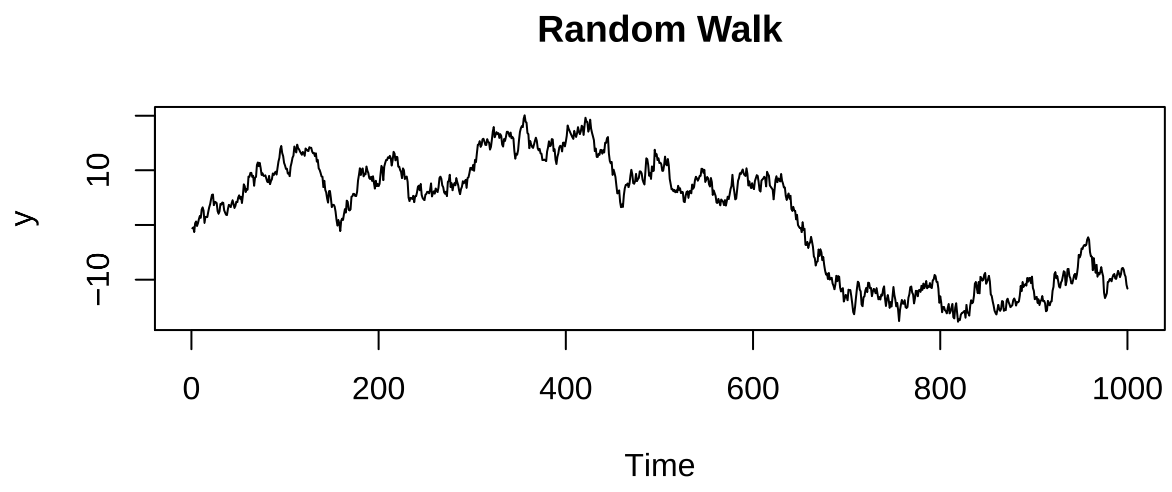


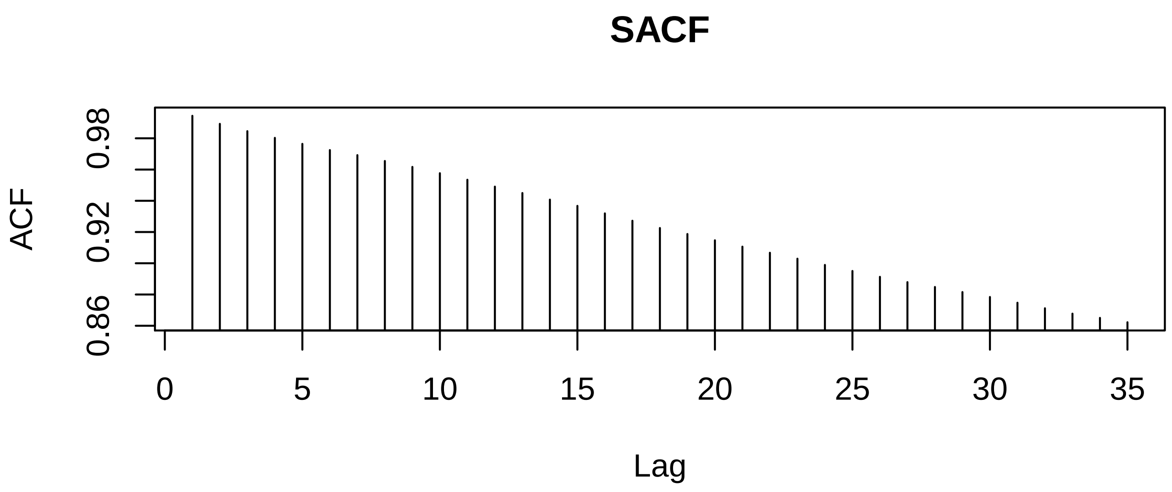
```
y = rbinom(100, 1, 1/2)
data_and_acf(y, plot_title = 'IID Gamma(0.5,1)')
```



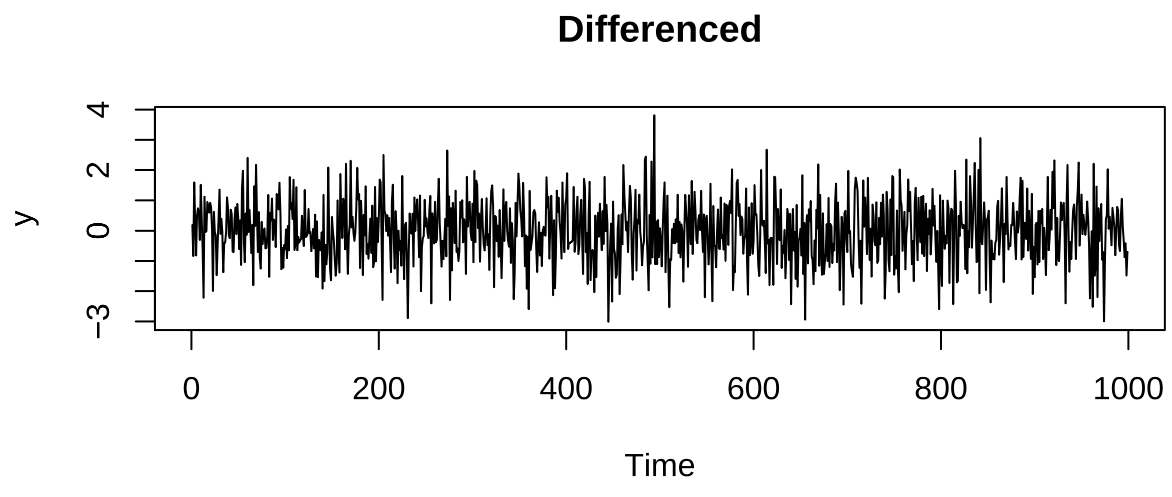


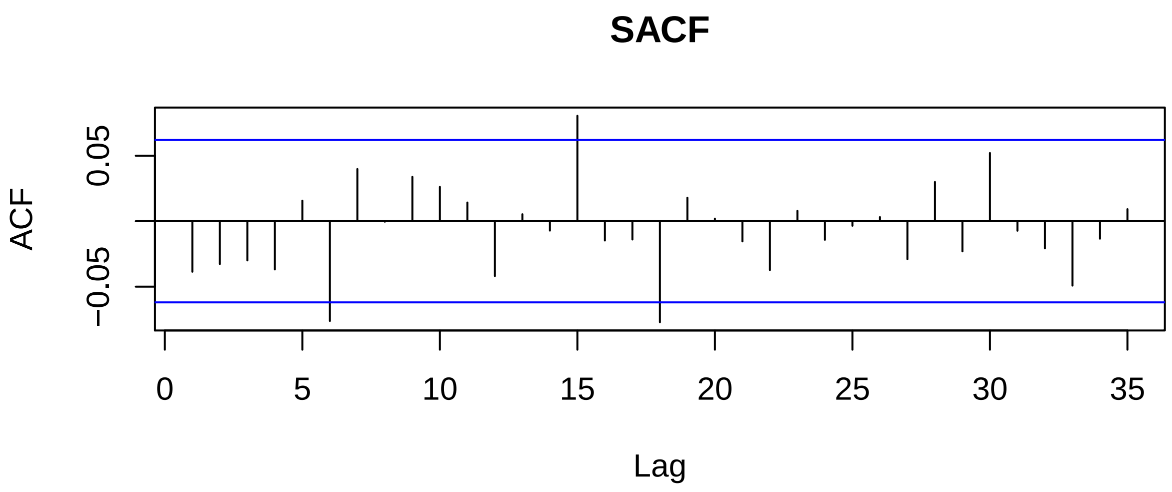
```r
set.seed(1)
y = rnorm(1000)
y = cumsum(y)
data_and_acf(y, plot_title = 'Random Walk')
```



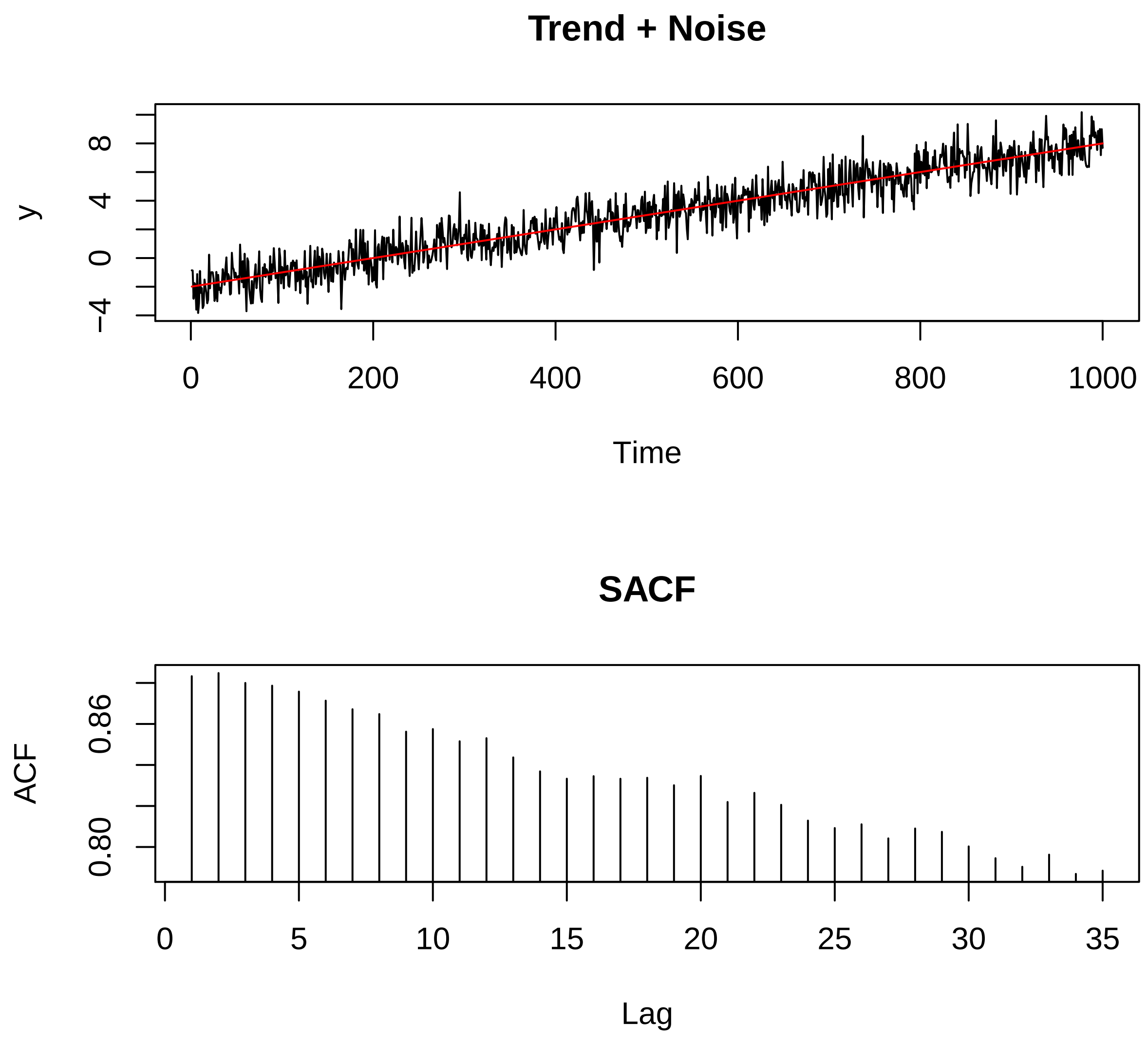


```
y = diff(y)
data_and_acf(y, plot_title = 'Differenced')
```

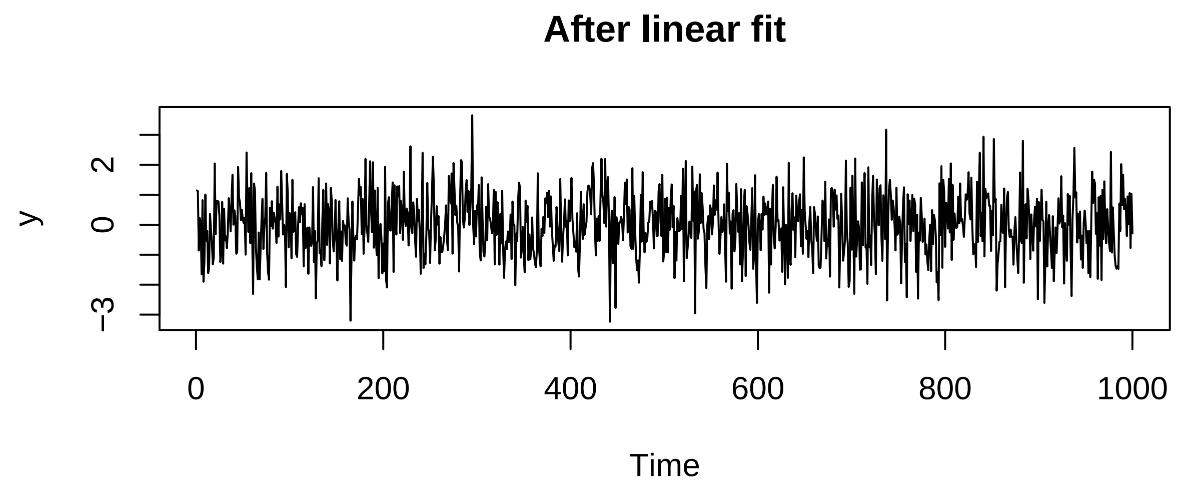


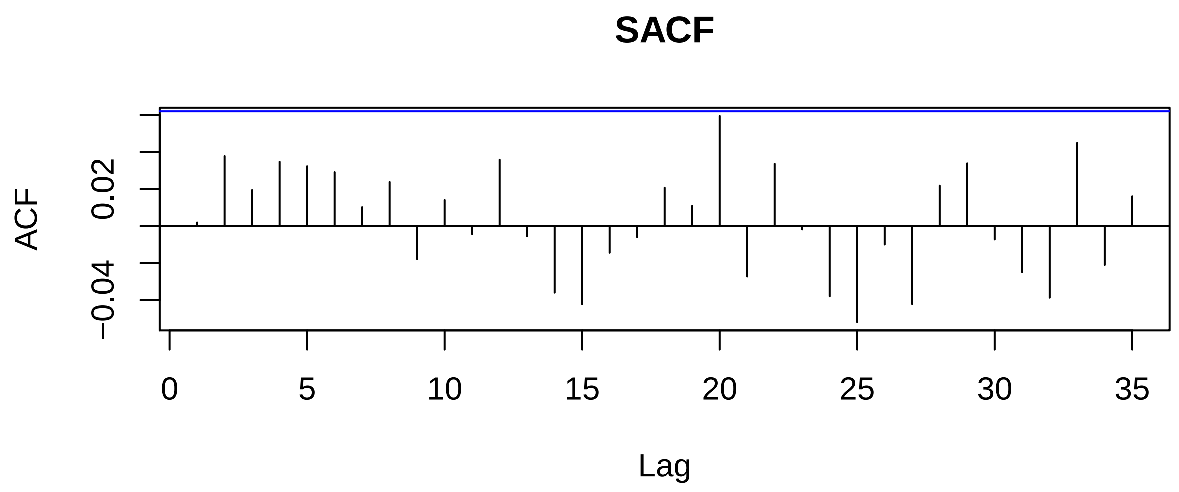


```
y = 1:1000 * 0.01 - 2 + rnorm(1000)
data_and_acf(y, plot_title = 'Trend + Noise',line=T)
```

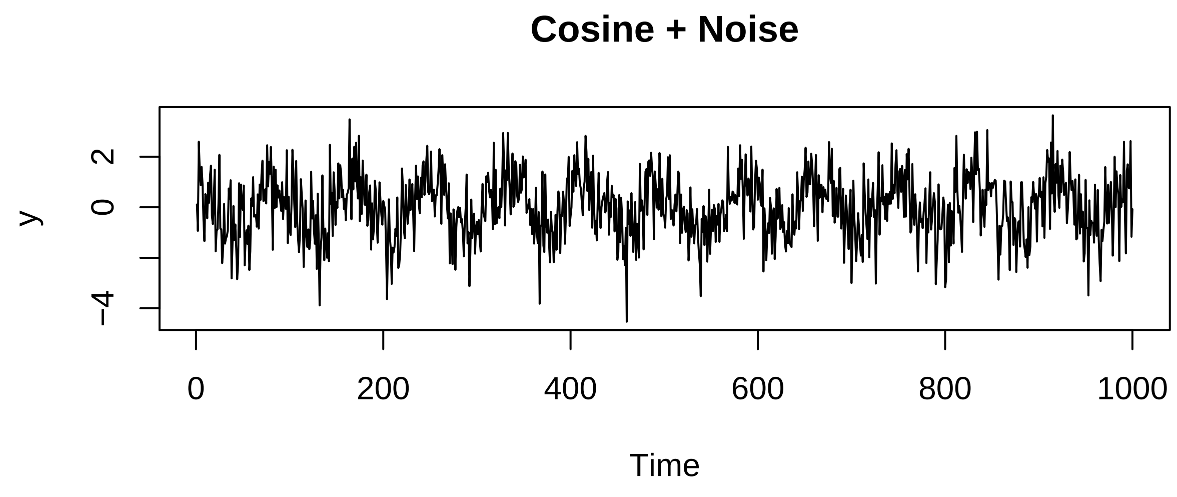


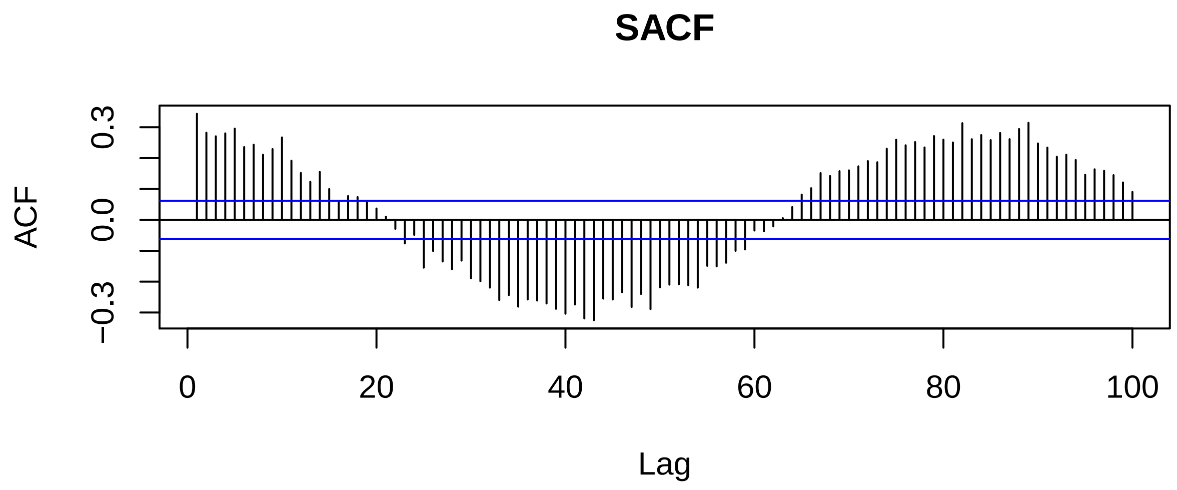
```
t = 1:length(y)
lm_fit = lm(y ~ t)
data_and_acf(y - lm_fit$fitted.values, plot_title='After linear fit')
```



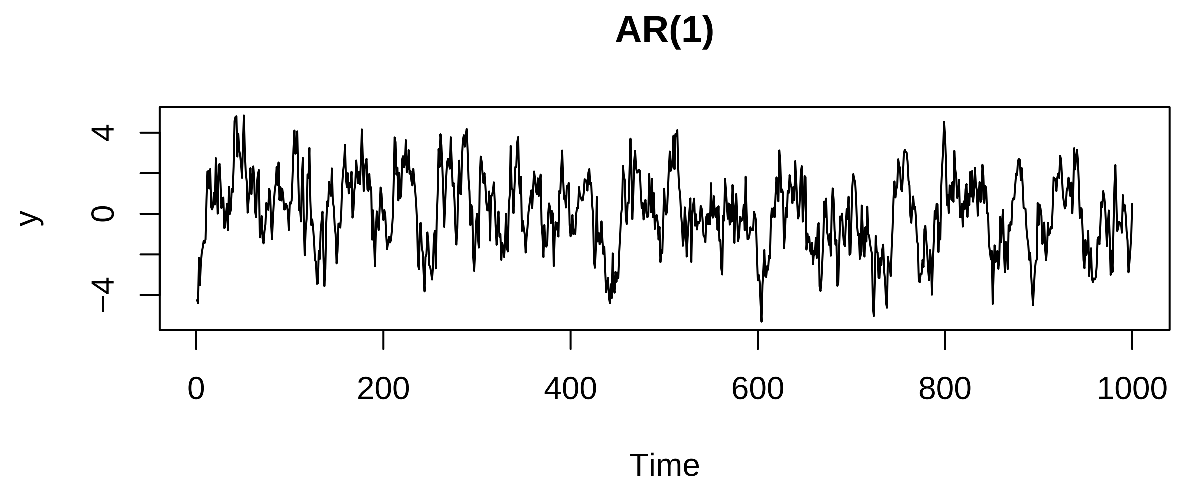


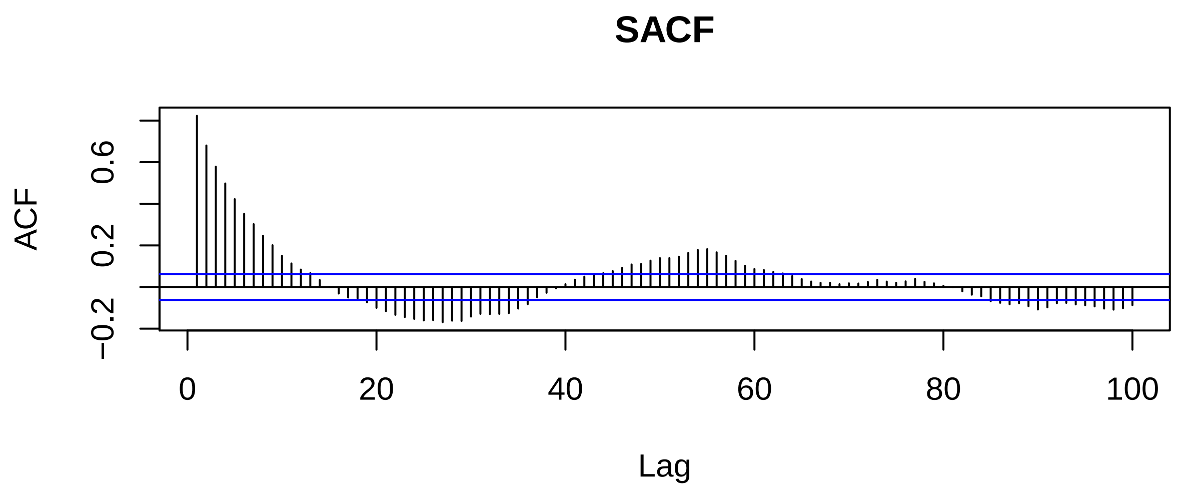
```
n = 1000
t=1:n; f1 = 12;
costerm1 = cos(f1*2*pi/n*t);
y = costerm1 + rnorm(n)
data_and_acf(y, lag=100, plot_title='Cosine + Noise')
```



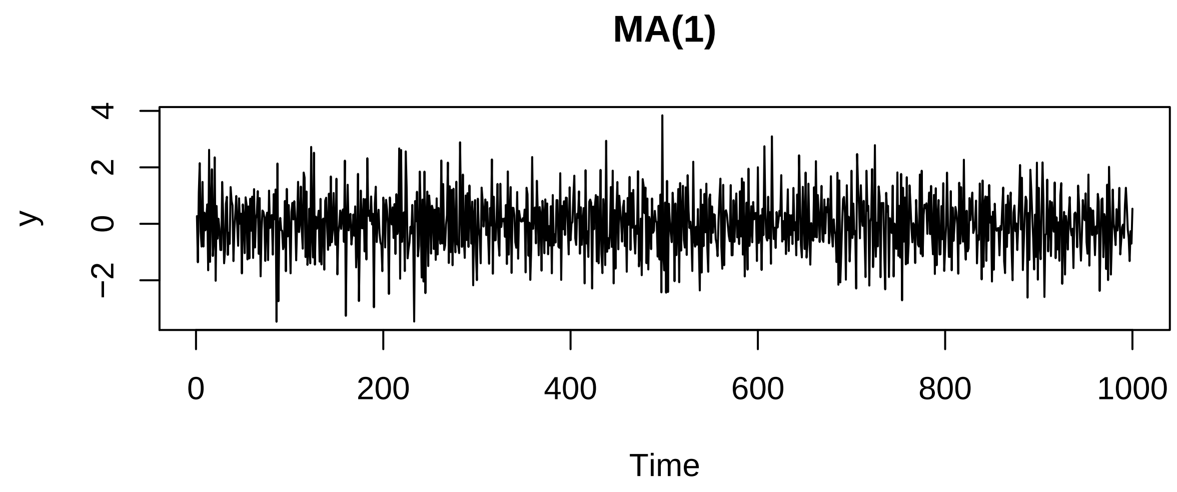


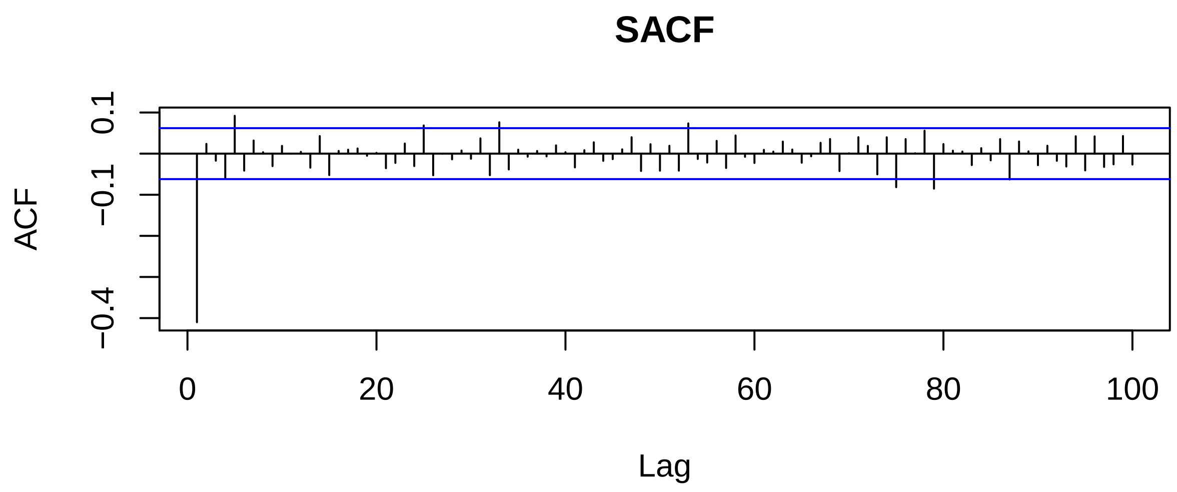
```
y = arima.sim(n=1000, list(ar=c(0.8)))
data_and_acf(y, lag=100, plot_title='AR(1)')
```





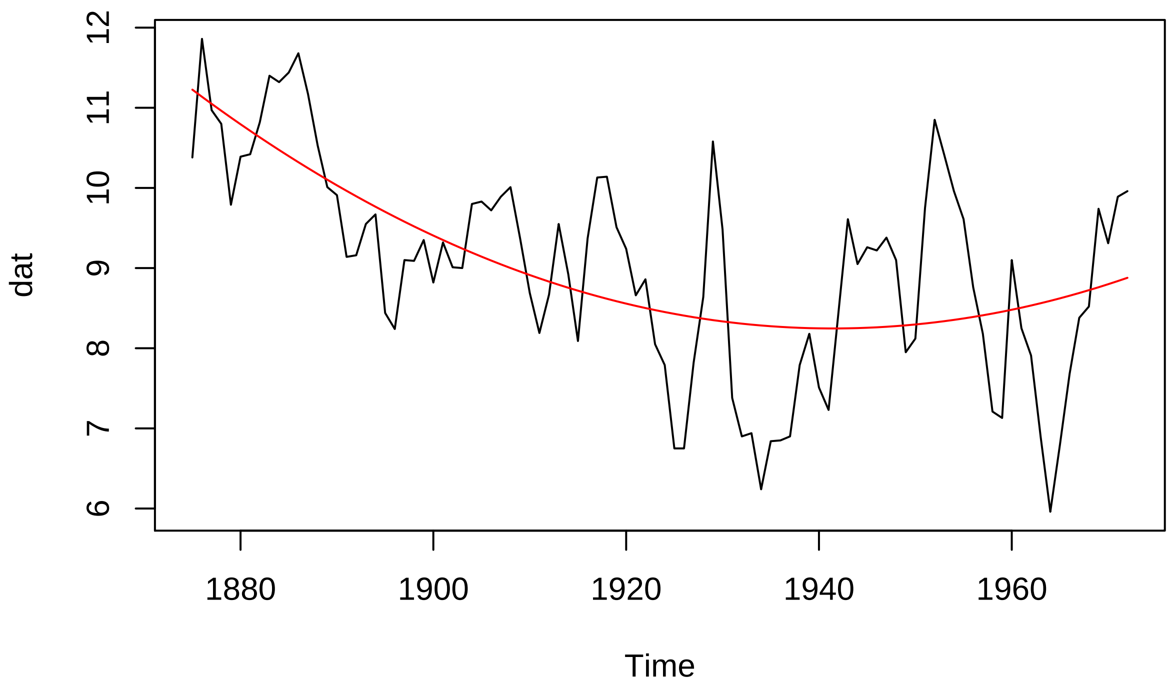
```
y = arima.sim(n=1000, list(ma=c(-0.5)))
data_and_acf(y, lag=100, plot_title='MA(1)')
```



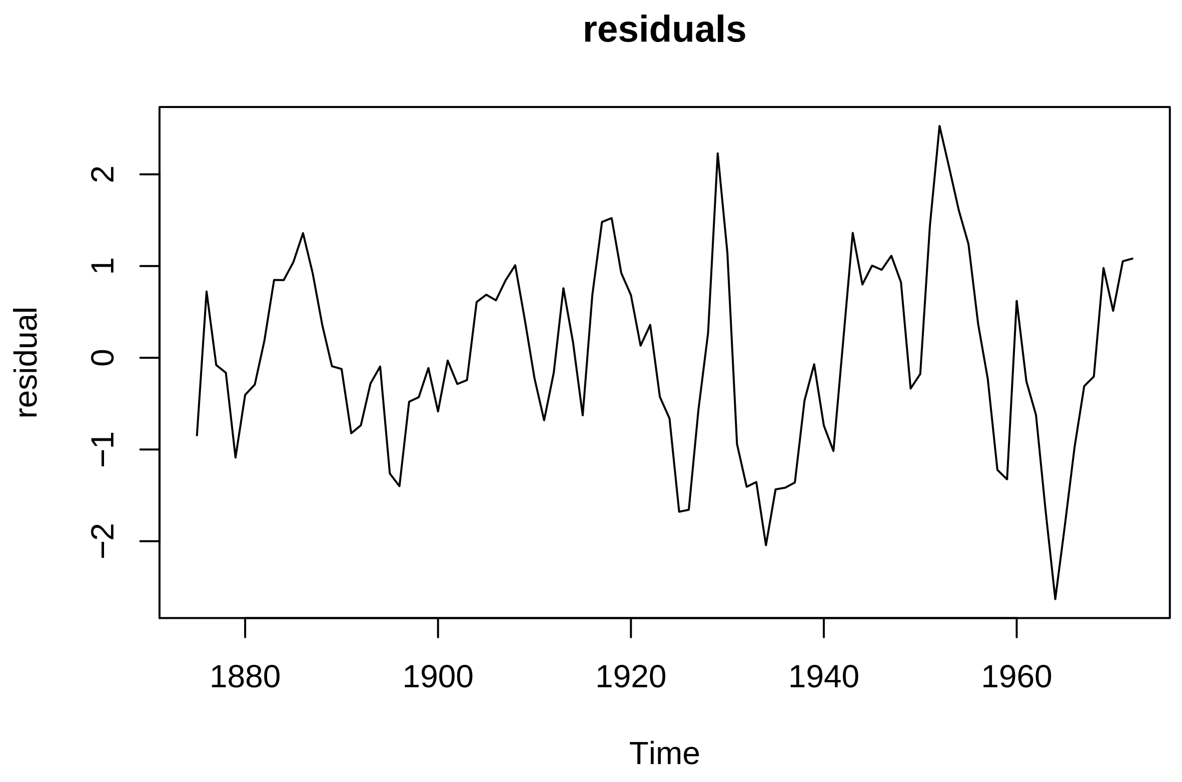


(p75) Test of randomness: Lake Huron

```
dat = itsmr::lake %>% ts(start=1875)
t = 1:length(dat)
fitted = lm(dat ~ 1 + t + I(t^2))$fitted.values%>%
  ts(start=1875)

plot.ts(dat)
lines(fitted, col='red')
```

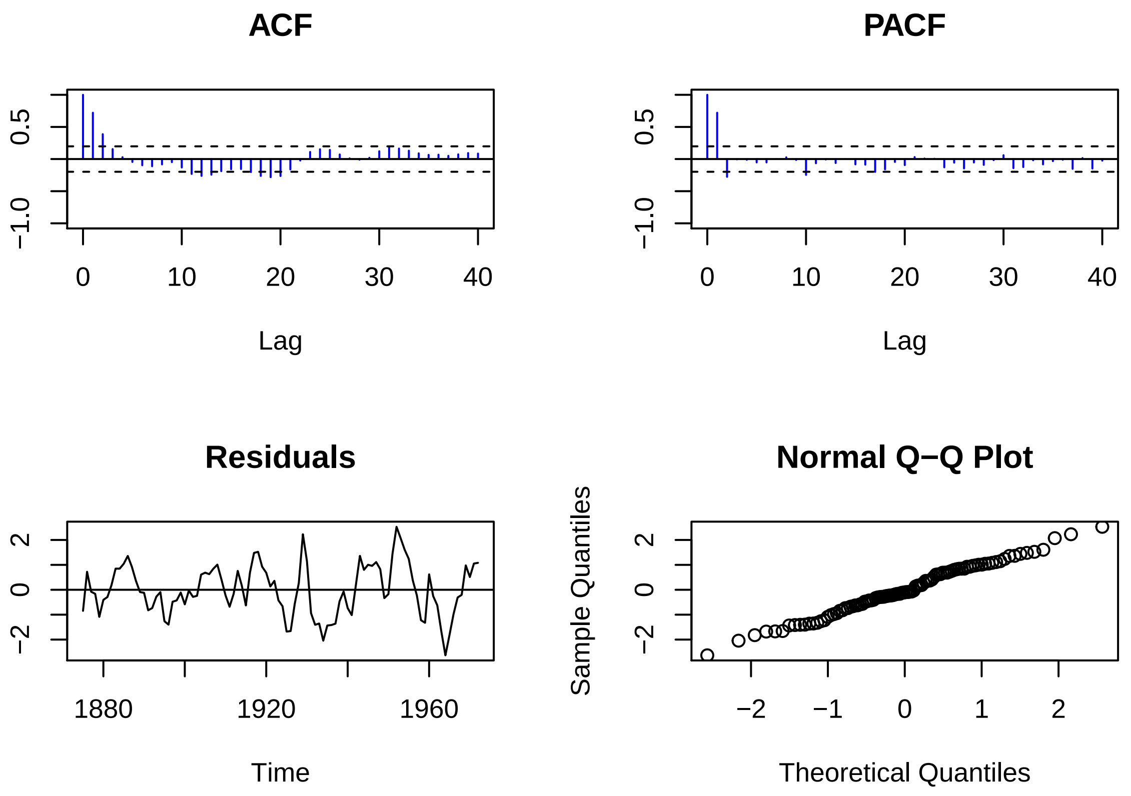


```
residual = dat - fitted
plot(residual);title('residuals')
```



```
itsmr::test(residual)
```

```
## Null hypothesis: Residuals are iid noise.
## Test                        Distribution Statistic   p-value
## Ljung-Box Q                Q ~ chisq(20)    138.67         0 *
## McLeod-Li Q                Q ~ chisq(20)     56.45         0 *
## Turning points T      (T-64)/4.1 ~ N(0,1)        40         0 *
## Diff signs S        (S-48.5)/2.9 ~ N(0,1)        50    0.6015
## Rank P         (P-2376.5)/162.9 ~ N(0,1)      2406    0.8563
```



```
nortest::lillie.test(residual)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  residual
## D = 0.072445, p-value = 0.2335
```

```
tseries::jarque.bera.test(residual)
```

```
##
##  Jarque Bera Test
##
## data:  residual
## X-squared = 0.53757, df = 2, p-value = 0.7643
```