Lab2. newhomes_sales

1. Load libraries

```
library(tseries)
library(fBasics)
```

```
## Loading required package: timeDate
## Loading required package: timeSeries
##
## Rmetrics Package fBasics
## Analysing Markets and calculating Basic Statistics
## Copyright (C) 2005-2014 Rmetrics Association Zurich
## Educational Software for Financial Engineering and Computational Science
## Rmetrics is free software and comes with ABSOLUTELY NO WARRANTY.
## https://www.rmetrics.org --- Mail to: info@rmetrics.org
```

2. Import Data

```
# set the working directory
setwd("~/Desktop/CSC425/week2/newhomesales")
```

```
# Load data with no variable names into the data frame "myd"
myd=read.table("newhomes_sales.csv",header=T, sep=',', skip=1)
# Prints the first row of the data
myd[1,]
```

```
##     date sales
## 1 Jan-63   591
```

```
# Prints the first col that contains dates
head(myd[,1], 20)
```

```
##  [1] Jan-63 Feb-63 Mar-63 Apr-63 May-63 Jun-63 Jul-63 Aug-63 Sep-63 Oct-63
## [11] Nov-63 Dec-63 Jan-64 Feb-64 Mar-64 Apr-64 May-64 Jun-64 Jul-64 Aug-64
## 599 Levels: 1-Apr 1-Aug 1-Dec 1-Feb 1-Jan 1-Jul 1-Jun 1-Mar 1-May ... Sep-99
```

```
tail(myd[,1], 20)
```

```
##  [1] 11-Apr 11-May 11-Jun 11-Jul 11-Aug 11-Sep 11-Oct 11-Nov 11-Dec 12-Jan
## [11] 12-Feb 12-Mar 12-Apr 12-May 12-Jun 12-Jul 12-Aug 12-Sep 12-Oct 12-Nov
## 599 Levels: 1-Apr 1-Aug 1-Dec 1-Feb 1-Jan 1-Jul 1-Jun 1-Mar 1-May ... Sep-99
```

```
# Prints the second col that contains home sales
head(myd[,2], 20)
```

```
##  [1] 591 464 461 605 586 526 665 570 590 567 579 514 549 609 562 559 523
## [18] 580 575 582
```

```
tail(myd[,2], 20)
```

```
##  [1] 312 308 304 297 292 306 314 327 339 339 366 352 358 369 360 366 367
## [18] 374 361 377
```

3. Create Times Series Variables "homesales"

The 'ts' function is used to create time-series objects for monthly or quarterly data.

For monthly data: ts(data, start = c(year ,month), frequency = 12) where year and month should be replaced by the starting year and month number.

For quarterly data: ts(data, start = c(year, quarter), frequency = 3) included in tseries. See R document in week 2 for more information.

```
salests = ts(myd$sales, start= c(1963,1), frequency=12)
salests
```

```
##       Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
## 1963  591  464  461  605  586  526  665  570  590  567  579  514
## 1964  549  609  562  559  523  580  575  582  590  583  548  540
## 1965  533  559  556  555  544  614  554  615  587  555  616  609
## 1966  599  541  557  545  499  434  435  377  358  387  382  369
## 1967  416  408  439  479  503  499  515  504  531  566  500  502
## 1968  494  543  490  501  441  441  493  507  501  502  469  511
## 1969  480  524  474  450  447  461  436  422  396  401  441  452
## 1970  461  373  389  445  466  485  481  515  564  545  570  582
## 1971  618  618  681  662  618  646  706  659  625  647  710  689
## 1972  698  711  640  684  677  687  681  773  767  843  735  772
## 1973  781  737  725  661  660  650  601  566  561  565  547  519
## 1974  523  539  572  544  590  534  534  492  511  448  450  417
## 1975  416  422  477  543  579  557  569  566  556  609  680  669
## 1976  603  644  591  611  570  591  664  648  696  708  735  767
## 1977  825  839  872  799  807  805  755  808  842  819  829  835
## 1978  795  791  814  864  857  834  789  756  812  872  798  805
## 1979  754  723  793  748  727  700  715  729  727  670  597  559
## 1980  592  541  474  370  469  552  636  659  596  561  562  532
## 1981  511  510  514  470  467  415  431  378  338  356  382  457
## 1982  368  365  374  339  384  370  375  407  481  480  554  521
## 1983  582  562  596  638  664  651  606  572  608  632  644  773
## 1984  691  696  641  639  615  630  619  567  662  687  597  597
## 1985  645  682  671  620  678  722  766  726  678  655  733  721
## 1986  733  728  880  857  789  728  698  621  763  669  707  784
## 1987  709  732  713  735  651  637  658  657  666  650  625  595
## 1988  585  663  669  699  684  717  679  688  703  718  628  658
## 1989  719  622  567  608  656  642  731  697  639  645  684  630
## 1990  620  591  574  542  534  545  542  528  496  465  493  464
## 1991  401  482  507  508  517  516  511  526  487  524  575  558
## 1992  676  639  554  546  554  596  627  636  650  621  614  650
## 1993  596  604  602  701  626  653  655  645  726  704  769  812
## 1994  619  686  747  692  691  621  628  656  677  715  646  629
## 1995  626  559  616  621  674  725  765  701  678  696  664  709
## 1996  714  769  721  736  746  721  770  826  770  720  771  805
```

```
## 1997  830  801  831  744  760  793  805  815  840  800  864  793
## 1998  872  866  836  866  887  923  876  846  864  893  995  949
## 1999  875  848  863  918  888  923  900  893  826  872  863  873
## 2000  873  856  900  841  857  793  887  848  912  933  880  983
## 2001  936  963  939  909  885  882  880  866  853  871  924  979
## 2002  880  948  923  936  978  957  956 1014 1044 1006 1024 1048
## 2003  999  936  999 1012 1078 1193 1168 1206 1131 1144 1093 1129
## 2004 1165 1159 1276 1186 1241 1180 1088 1175 1214 1305 1179 1242
## 2005 1203 1319 1328 1260 1286 1274 1389 1255 1244 1336 1214 1239
## 2006 1174 1061 1116 1123 1086 1074  965 1035 1016  941 1003  998
## 2007  891  828  833  887  842  793  778  699  686  727  641  619
## 2008  627  593  535  536  504  487  477  435  433  393  389  377
## 2009  336  372  339  337  376  393  411  418  386  396  375  352
## 2010  345  336  381  422  280  305  283  282  317  291  287  326
## 2011  308  273  301  312  308  304  297  292  306  314  327  339
## 2012  339  366  352  358  369  360  366  367  374  361  377
```

```
# start date
start(salests)
```

```
## [1] 1963    1
```

```
# end date
end(salests)
```

```
## [1] 2012   11
```

```
#computes change in monthly home sales
chg = (salests-lag(salests,-1))/lag(salests,-1)
# to add 'chg' to dataframe create a new column and then add values of 'chg'
#myd[chg]<-NA
myd$chg <- c(NA, chg)
```

4. Compute Summary Statistics

```
basicStats(myd$sales)
```

```
##             X..myd.sales
## nobs        5.990000e+02
## NAs         0.000000e+00
## Minimum     2.730000e+02
## Maximum     1.389000e+03
## 1. Quartile 5.165000e+02
## 3. Quartile 7.930000e+02
## Mean        6.652504e+02
## Median      6.390000e+02
## Sum         3.984850e+05
## SE Mean     8.961442e+00
## LCL Mean    6.476507e+02
## UCL Mean    6.828501e+02
## Variance    4.810416e+04
## Stdev       2.193266e+02
## Skewness    7.431410e-01
## Kurtosis    4.797460e-01
```

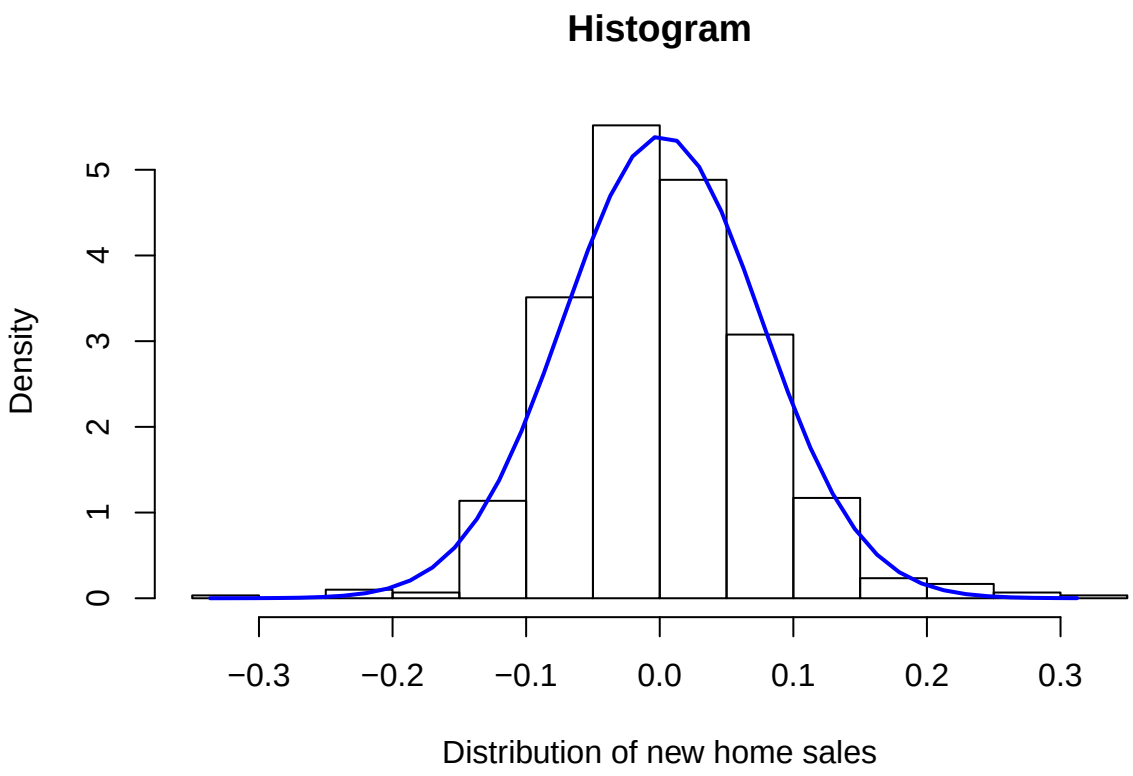
```r
# same results if we use time series object rts
basicStats(salests)
```

```
##                  salests
## nobs        5.990000e+02
## NAs         0.000000e+00
## Minimum     2.730000e+02
## Maximum     1.389000e+03
## 1. Quartile 5.165000e+02
## 3. Quartile 7.930000e+02
## Mean        6.652504e+02
## Median      6.390000e+02
## Sum         3.984850e+05
## SE Mean     8.961442e+00
## LCL Mean    6.476507e+02
## UCL Mean    6.828501e+02
## Variance    4.810416e+04
## Stdev       2.193266e+02
## Skewness    7.431410e-01
## Kurtosis    4.797460e-01
```

5. Create Histogram

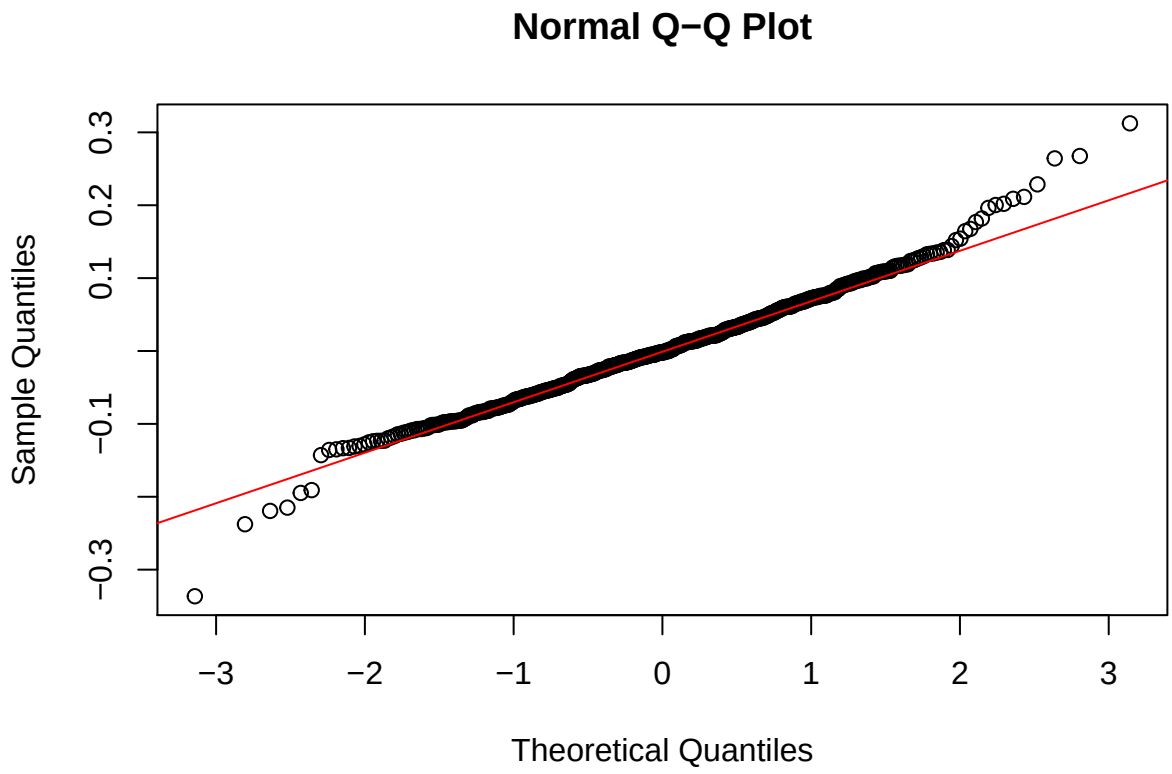
```r
# creates 2 by 2 display for 4 plots
par(mfcol = c(1,1))
hist(chg, xlab = "Distribution of new home sales", prob = TRUE, main = "Histogram")

# add approximating normal density curve
xfit <- seq(min(chg), max(chg), length = 40)
yfit <- dnorm(xfit, mean = mean(chg), sd = sd(chg))
lines(xfit, yfit, col = "blue", lwd = 2)
```



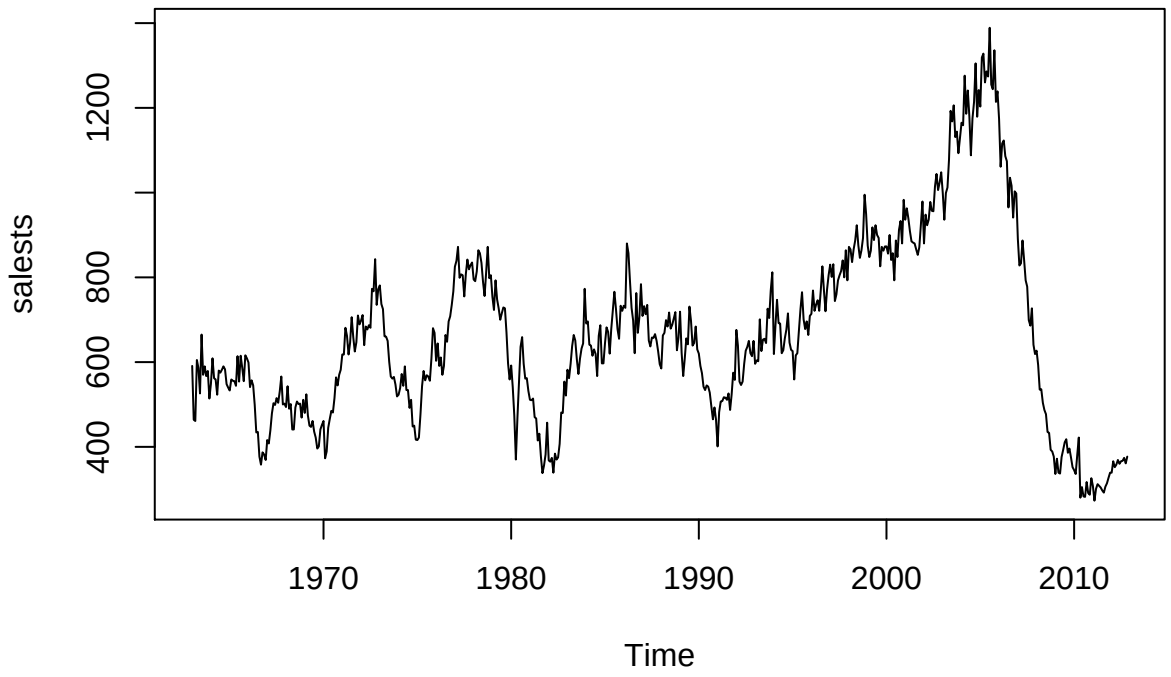
6. Create Normal Probability Plot

```
qqnorm(chg)
qqline(chg, col = 2)
```

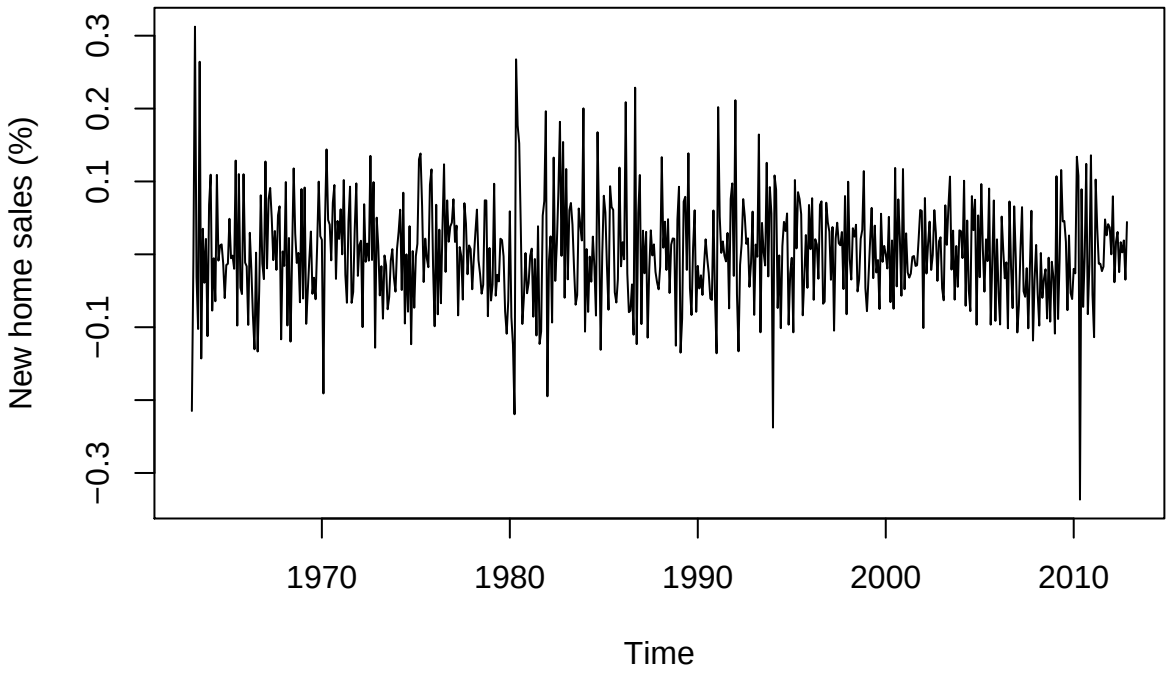


7. Create Time Plot

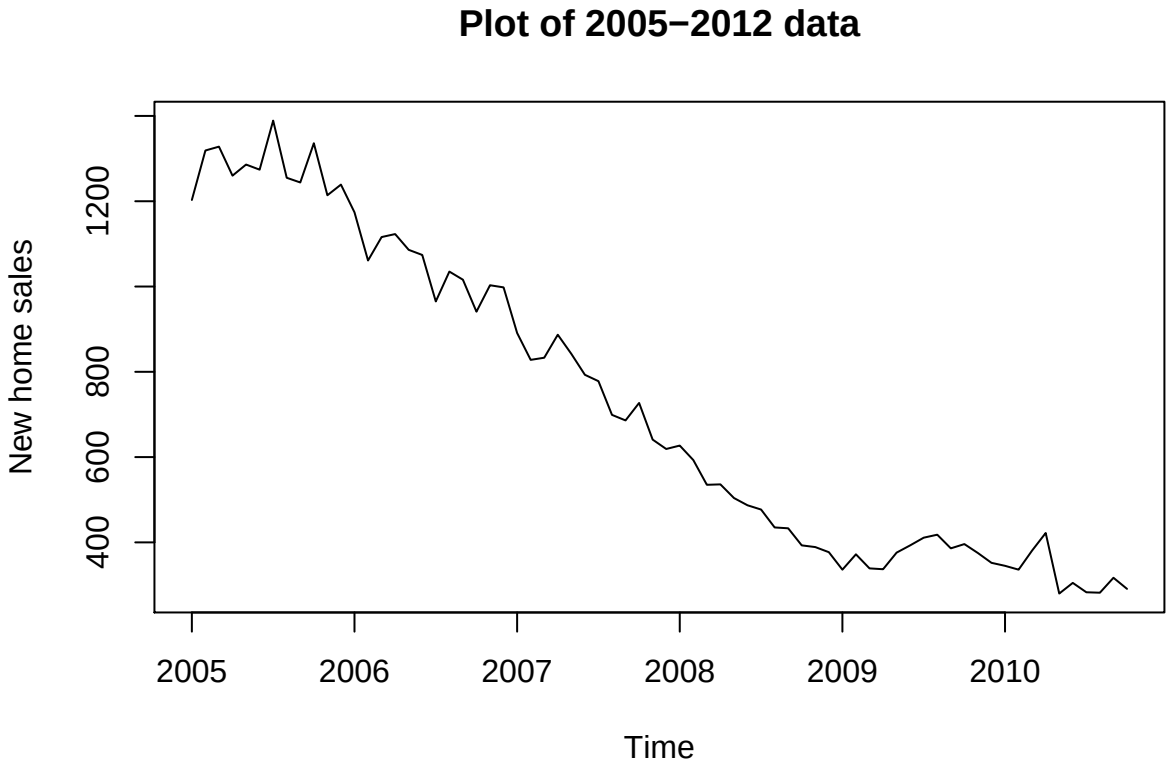
```
# use time series object to draw time plot indexed with time
plot(salests)
```




```
# Time plot of changes in new home sales
plot(chg, ylab = 'New home sales (%)')
```




```
#creats subsets of data in smaller period of time
sales_05 = window(salests, start = c(2005,1), end = c(2010,10))
# plot the subset of data
plot(sales_05, type = 'l', ylab = "New home sales", main = "Plot of 2005-2012 data")
```



8. Normality Tests

```
# Perfom Jarque-Bera normality test
normalTest(chg,method=c("jb"))
```

```
##
## Title:
##  Jarque - Bera Normalality Test
##
## Test Results:
##   STATISTIC:
##     X-squared: 58.0173
##   P VALUE:
##     Asymptotic p Value: 2.521e-13
##
## Description:
##  Thu Sep 21 15:37:35 2017 by user:
```

```
# Fat-tail test
k_stat = kurtosis(chg)/sqrt(24/length(chg))
print("Kurtosis test statistic")
```

```
## [1] "Kurtosis test statistic"
```

```
k_stat
```

```
## [1] 7.305846
## attr(,"method")
## [1] "excess"
```

```
print("P-value = ")
```

```
## [1] "P-value = "
```

```
2 * (1-pnorm(abs(k_stat)))
```

```
## [1] 2.755574e-13
## attr(,"method")
## [1] "excess"
```

```
# Skewness test
skew_test = skewness(chg)/sqrt(6/length(chg))
skew_test
```

```
## [1] 1.878044
## attr(,"method")
## [1] "moment"
```

```
# P-value
2 * ( 1- pnorm(abs(skew_test)))
```

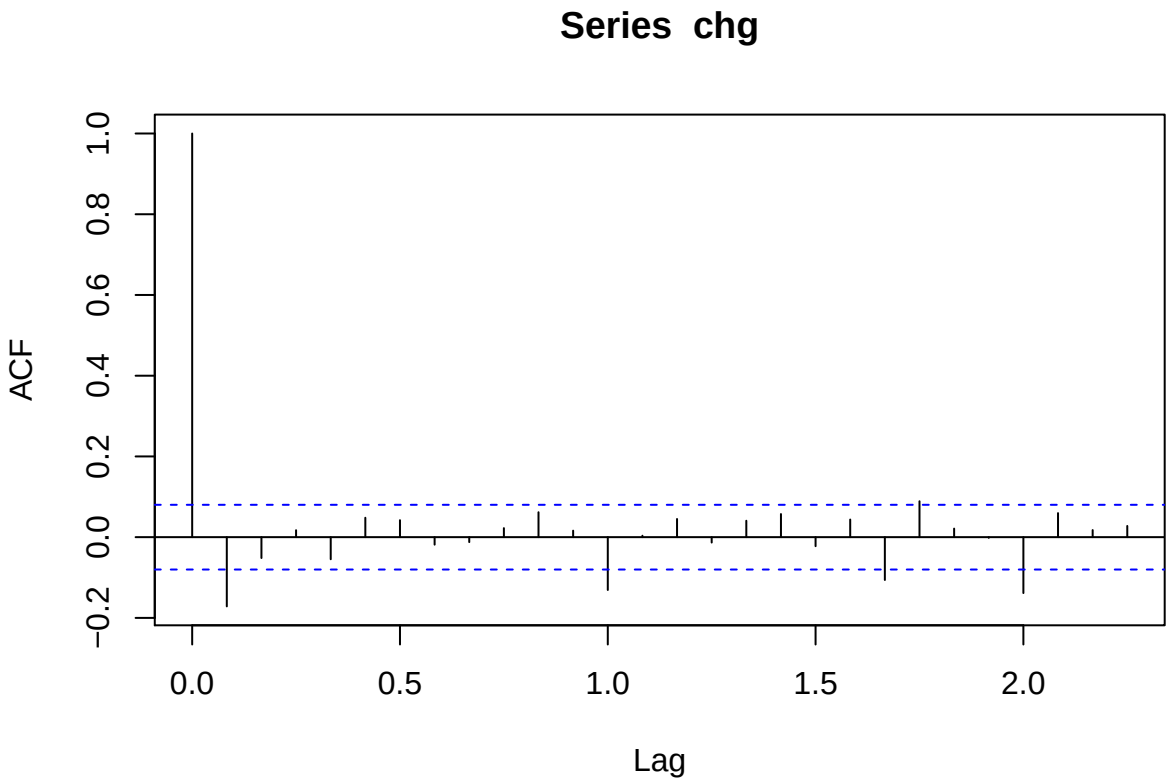
```
## [1] 0.06037515
## attr(,"method")
## [1] "moment"
```

9. Compute ACF and Plot Correlogram

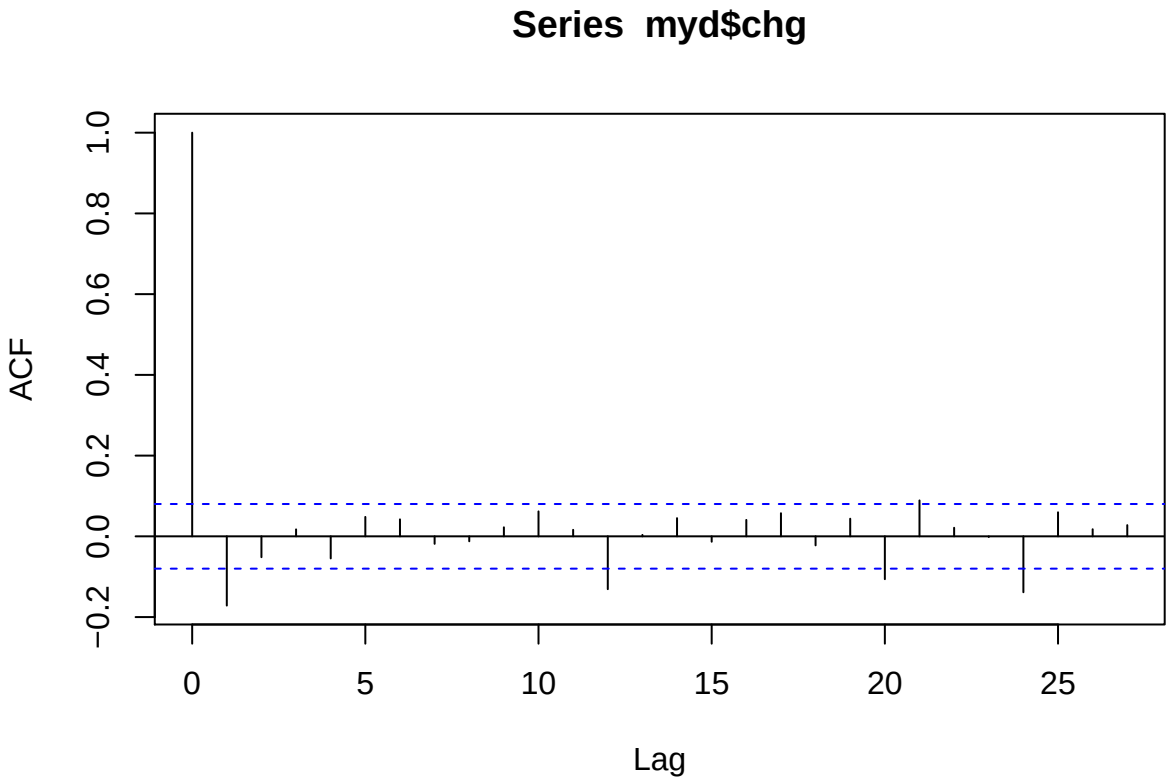
```
# prints acf to concole
acf(chg, plot = F)
```

```
##
## Autocorrelations of series 'chg', by lag
##
## 0.0000 0.0833 0.1667 0.2500 0.3333 0.4167 0.5000 0.5833 0.6667 0.7500
##  1.000 -0.172 -0.052  0.017 -0.055  0.048  0.042 -0.019 -0.012  0.022
## 0.8333 0.9167 1.0000 1.0833 1.1667 1.2500 1.3333 1.4167 1.5000 1.5833
##  0.062  0.016 -0.131  0.004  0.045 -0.013  0.041  0.057 -0.022  0.044
## 1.6667 1.7500 1.8333 1.9167 2.0000 2.0833 2.1667 2.2500
## -0.106  0.089  0.021 -0.002 -0.138  0.060  0.018  0.028
```

```
# plots acf (correlogram)
acf(chg, plot = T)
```




```
# to plot correlogram showing lags in integer values
acf(myd$chg, na.action = na.pass) #na.action=na.pass to allow missing values
```



10.Compute Ljung-Box Test for White Noise (No Autocorrelation)

```
# to lag 6
Box.test(chg, lag = 6, type = 'Ljung')
```

```
##
##  Box-Ljung test
##
## data:  chg
## X-squared = 23.756, df = 6, p-value = 0.0005791
```

```
# to lag 12
Box.test(chg, lag = 12, type = 'Ljung')
```

```
##
##  Box-Ljung test
##
## data:  chg
## X-squared = 37.352, df = 12, p-value = 0.0001958
```