Introduction to R

Alok K. Shukla

1/23/2017

Asking for help in R:

```
?plot
help(sum)
?help
```

Example of a script

```
x = 100;
y = x+1
print(y)
```

```
## [1] 101
```

Scalar Operations

Basic Math Operations

```
x = 100; y <- 69
69 -> y;
sumxy = x+y;
prodxy = x*y;
diffxy = x-y;
quoxy = x/y
```

Other math operations

```
xsq = x^2; lx = log(x); xroot = sqrt(xsq)
```

Complex Numbers, finding modulus and argument:

```
z = x + 1i*y;
magz = abs(z); argz = Arg(z)
lz = log10(z)
is.complex(z)
```

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

Check for types

```
xstr = "String"
is.numeric(xstr)
```

```
## [1] FALSE
```

Printing

```
x =3.4563
print(x,digits=2)
```

```
## [1] 3.5
```

Vector Operations

```
(xvec = c(4,2,6,8,9))  # concatenation, not a 1D matrix
```

```
## [1] 4 2 6 8 9
```

```
(yvec = c(1,2,3,4,5))
```

```
## [1] 1 2 3 4 5
```

```
(yvec2 = (1:5))
```

```
## [1] 1 2 3 4 5
```

```
(yvec3 = seq(from=1,by=1,length.out = 5))
```

```
## [1] 1 2 3 4 5
```

```
(zvec = c(xvec,yvec))
```

```
##  [1] 4 2 6 8 9 1 2 3 4 5
```

```
length(zvec)
```

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

```
dim(zvec)
```

```
## NULL
```

Accessing elements

```
(x4 = xvec[4])
```

```
## [1] 8
```

Subsetting

```
(x25 = xvec[c(2,5)])
```

```
## [1] 2 9
```

```
(x2to5 = xvec[2:5])
```

```
## [1] 2 6 8 9
```

```
(xgt5 = xvec[xvec>5])
```

```
## [1] 6 8 9
```

Mathematical Operations

```
xyadd = xvec + yvec; xyprod = xvec*yvec; c=xvec^2 ; xlog = log(xvec)
xyadd
```

```
## [1]  5  4  9 12 14
```

```
xyprod
```

```
## [1]  4  4 18 32 45
```

```
c
```

```
## [1] 16  4 36 64 81
```

```
xlog
```

```
## [1] 1.3862944 0.6931472 1.7917595 2.0794415 2.1972246
```

Statistical Summary

```
summary(xvec)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     2.0     4.0     6.0     5.8     8.0     9.0
```

```
(mux = mean(xvec))
```

```
## [1] 5.8
```

```
(sigmax = sqrt(var(xvec)))
```

```
## [1] 2.863564
```

```
(minx = min(xvec))
```

```
## [1] 2
```

Properties of vector

```
(n2 = length(xvec2 <- rnorm(4))) # on the fly
```

```
## [1] 4
```

```
names(xvec2) = c('First','Second','Third','Fourth')
print(xvec2)
```

```
##       First      Second       Third      Fourth
## -0.02335036  1.05322251  3.05359780  2.96660040
```

Matrix operations

Creating a matrix

```
(A = matrix(1:20, nrow = 4, ncol = 5)) # ncol optional
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    5    9   13   17
## [2,]    2    6   10   14   18
## [3,]    3    7   11   15   19
## [4,]    4    8   12   16   20
```

```
(B = matrix(1:20, nrow = 4, ncol =5, byrow = 'T'))
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    2    3    4    5
## [2,]    6    7    8    9   10
## [3,]   11   12   13   14   15
## [4,]   16   17   18   19   20
```

Concatenation, row and column-wise

```
rbind(A,B)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    5    9   13   17
```

```
## [2,]    2    6   10   14   18
## [3,]    3    7   11   15   19
## [4,]    4    8   12   16   20
## [5,]    1    2    3    4    5
## [6,]    6    7    8    9   10
## [7,]   11   12   13   14   15
## [8,]   16   17   18   19   20
```

```
cbind(A,B)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
## [1,]    1    5    9   13   17    1    2    3    4     5
## [2,]    2    6   10   14   18    6    7    8    9    10
## [3,]    3    7   11   15   19   11   12   13   14    15
## [4,]    4    8   12   16   20   16   17   18   19    20
```

Accessing elements, display is always row-wise

```
(p = A[2,3])
```

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

```
(q = B[1:3,2:4])
```

```
##      [,1] [,2] [,3]
## [1,]    2    3    4
## [2,]    7    8    9
## [3,]   12   13   14
```

```
(r = A[3,])
```

```
## [1]  3  7 11 15 19
```

```
(s = B[,4])
```

```
## [1]  4  9 14 19
```

```
# Vec
xvec[-3]
```

```
## [1] 4 2 8 9
```

```
A[,-3]
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    5   13   17
## [2,]    2    6   14   18
## [3,]    3    7   15   19
## [4,]    4    8   16   20
```

n-D Array

```
(B2 = array(c(1:3),c(4,5))) # Data, dim
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    2    3    1    2
## [2,]    2    3    1    2    3
## [3,]    3    1    2    3    1
## [4,]    1    2    3    1    2
```

```
as.vector(B2) # vectorization, stacks columns
```

```
##  [1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2
```

```
dim(B2)
```

```
## [1] 4 5
```

```
(C2 = array(seq(1,3,length=12),c(2,3,2)))
```

```
## , , 1
##
##          [,1]     [,2]     [,3]
## [1,] 1.000000 1.363636 1.727273
## [2,] 1.181818 1.545455 1.909091
##
## , , 2
##
##          [,1]     [,2]     [,3]
## [1,] 2.090909 2.454545 2.818182
## [2,] 2.272727 2.636364 3.000000
```

```
dim(C2)
```

```
## [1] 2 3 2
```

```
C2[,,1]
```

```
##          [,1]     [,2]     [,3]
## [1,] 1.000000 1.363636 1.727273
## [2,] 1.181818 1.545455 1.909091
```

Element-wise operations

```
(C = A+B)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    2    7   12   17   22
## [2,]    8   13   18   23   28
## [3,]   14   19   24   29   34
## [4,]   20   25   30   35   40
```

```
(D = A*B)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1   10   27   52   85
## [2,]   12   42   80  126  180
## [3,]   33   84  143  210  285
## [4,]   64  136  216  304  400
```

```
(E = A^2)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1   25   81  169  289
## [2,]    4   36  100  196  324
## [3,]    9   49  121  225  361
## [4,]   16   64  144  256  400
```

```
(F = log(B)/A)
```

```
##           [,1]      [,2]      [,3]      [,4]       [,5]
## [1,] 0.0000000 0.1386294 0.1220680 0.1066380 0.09467282
## [2,] 0.8958797 0.3243184 0.2079442 0.1569446 0.12792139
```

```
## [3,] 0.7992984 0.3549867 0.2331772 0.1759372 0.14252896
## [4,] 0.6931472 0.3541517 0.2408643 0.1840274 0.14978661
```

Matrix operations

```
# A %*% A
t(A) %*% A
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]   30   70  110  150  190
## [2,]   70  174  278  382  486
## [3,]  110  278  446  614  782
## [4,]  150  382  614  846 1078
## [5,]  190  486  782 1078 1374
```

Inverse

```
A = matrix(rnorm(16),nrow=4);
(Ainv = solve(A)) # Not as by hand
```

```
##            [,1]         [,2]        [,3]       [,4]
## [1,] -0.3127610  0.341175986  0.09205861  0.1452811
## [2,] -0.2096946  0.176847440 -0.53816936 -0.5184807
## [3,]  0.6112753  0.007700959 -0.19462411  0.3162485
## [4,]  0.5678561 -1.037113037 -0.25966879  1.1908793
```

```
(Ainv = qr.solve(A)) # More effecient
```

```
##            [,1]         [,2]        [,3]       [,4]
## [1,] -0.3127610  0.341175986  0.09205861  0.1452811
## [2,] -0.2096946  0.176847440 -0.53816936 -0.5184807
## [3,]  0.6112753  0.007700959 -0.19462411  0.3162485
## [4,]  0.5678561 -1.037113037 -0.25966879  1.1908793
```

```
(Ainv %*% A)
```

```
##              [,1]          [,2]          [,3]         [,4]
## [1,] 1.000000e+00  4.857226e-17 -6.245005e-17 4.163336e-17
## [2,] 1.110223e-16  1.000000e+00  0.000000e+00 0.000000e+00
## [3,] 2.220446e-16  1.838807e-16  1.000000e+00 5.551115e-17
## [4,] 4.440892e-16 -1.665335e-16  0.000000e+00 1.000000e+00
```

Enough, save workspace variables

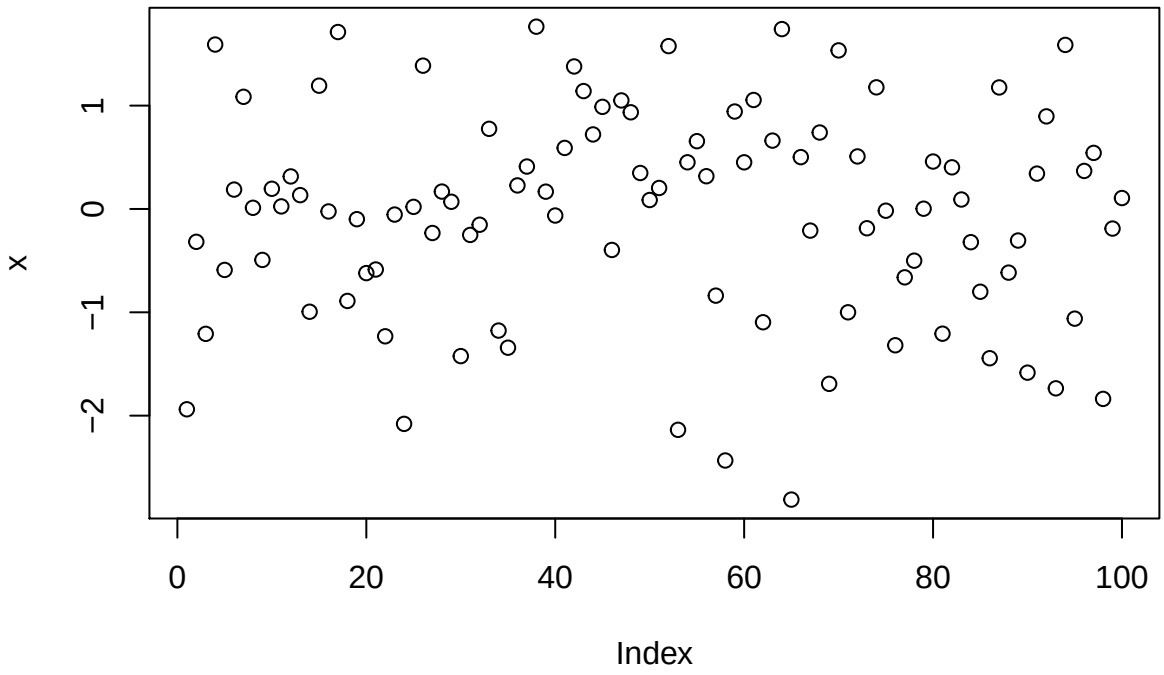
save.image(file='var.RData') # Clear now and load load('var.RData')

Plotting

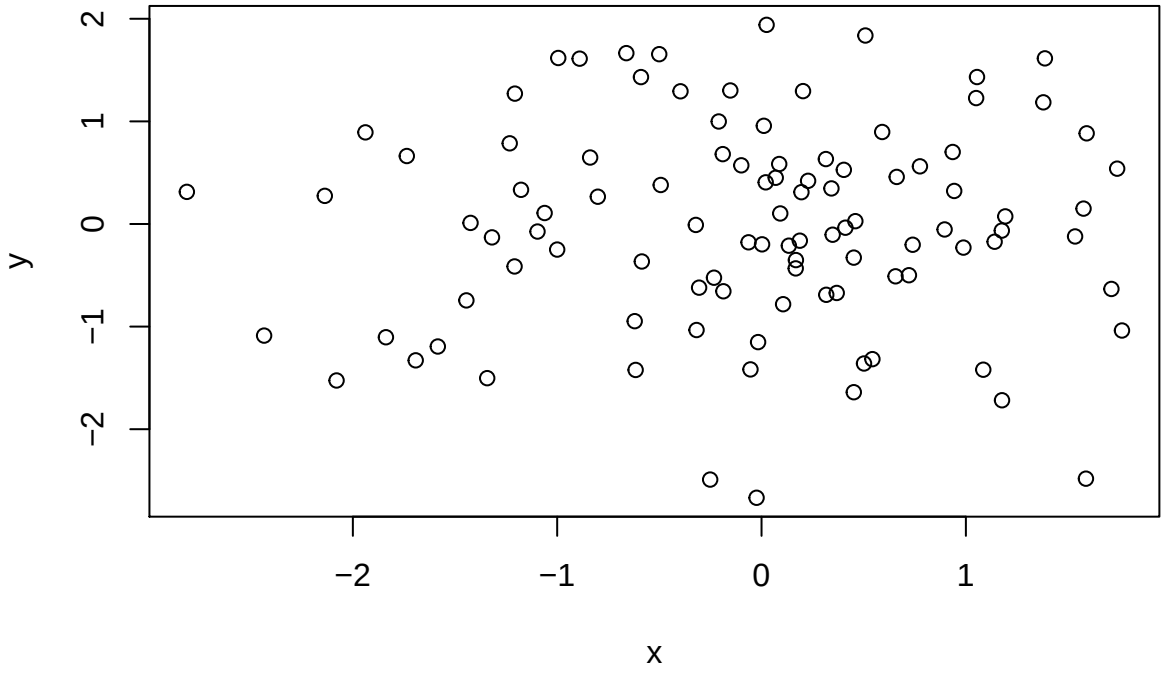
```
x = rnorm(100)
y = rnorm(100)
```

Scatter plots are default

```
plot(x)
```

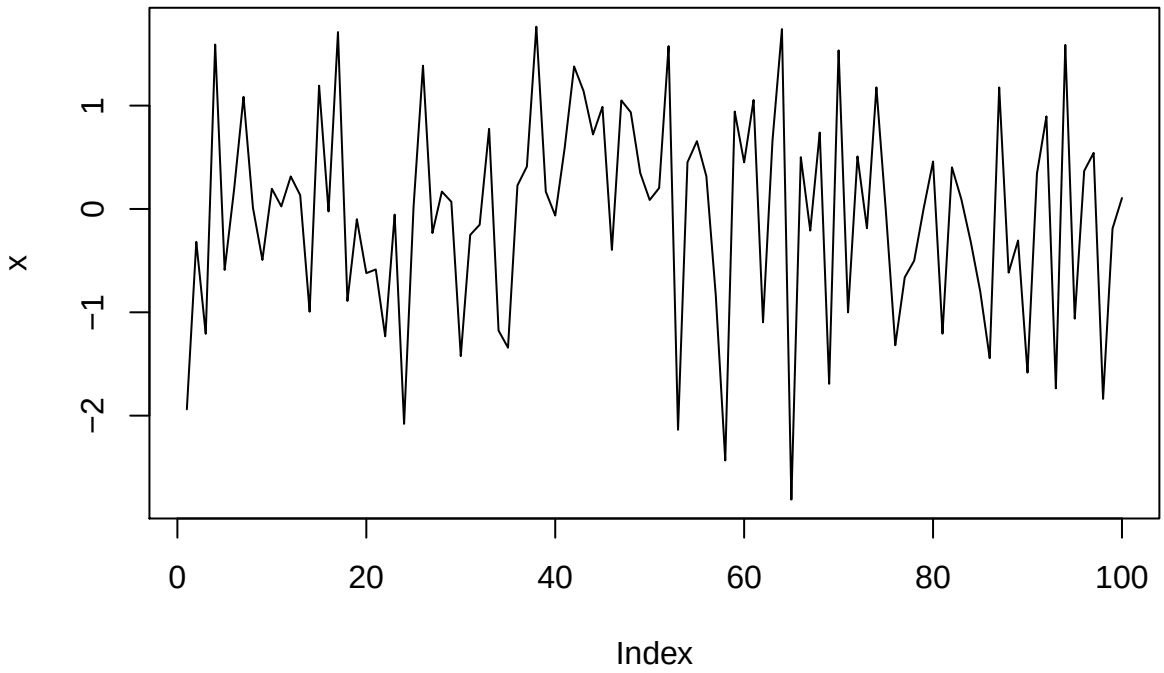



```
plot(x,y)
```



Other kinds

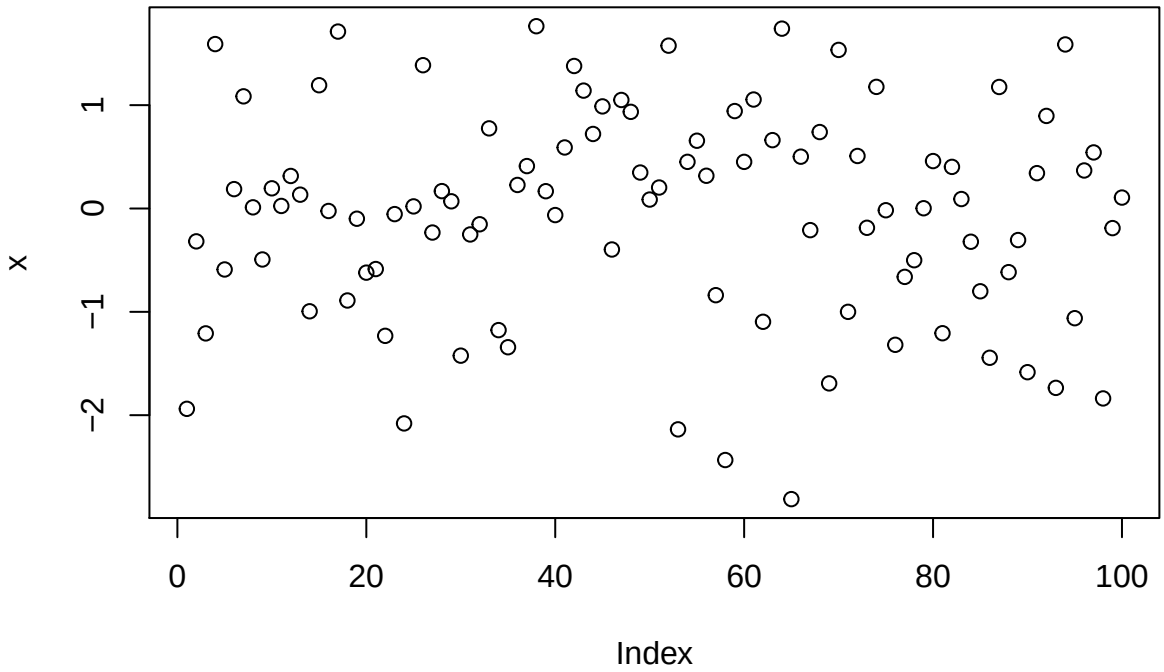
```
plot(x,type='l') # Default when TS
```



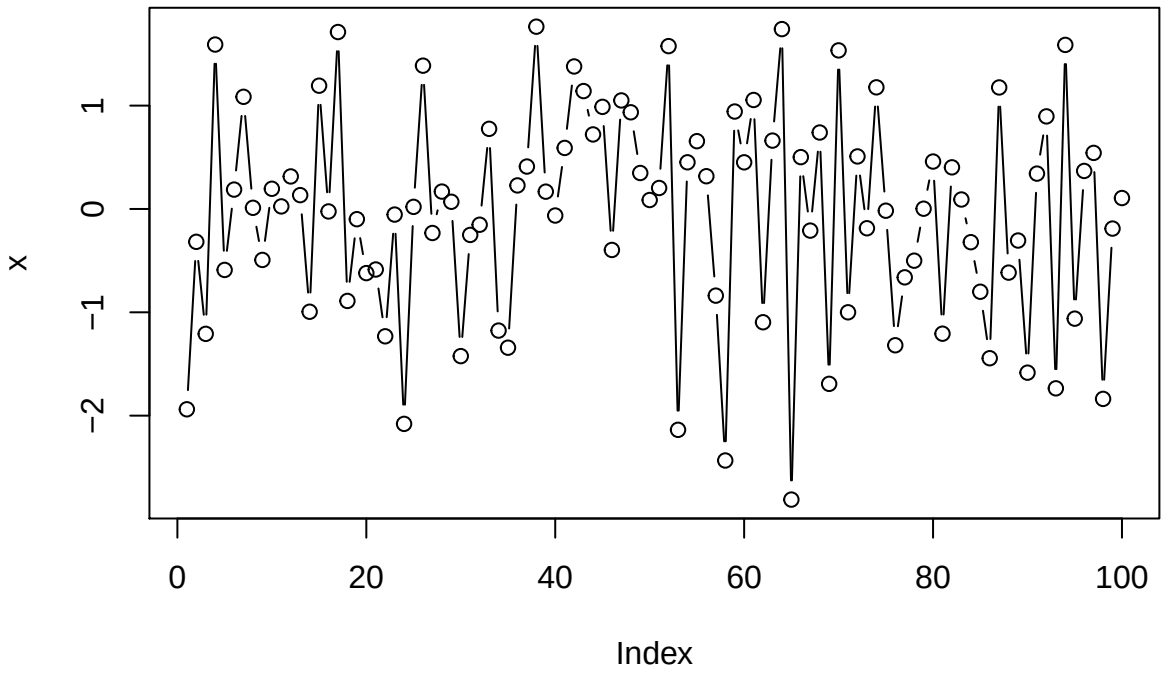

```
is.ts(x)
```

```
## [1] FALSE
```

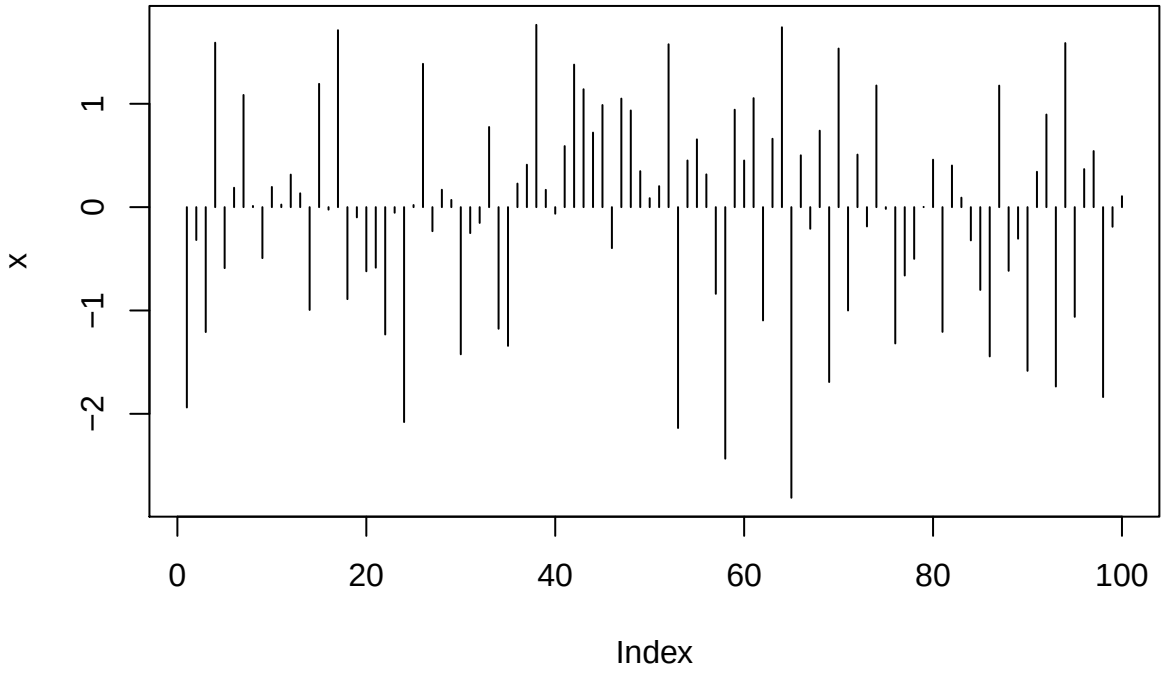
```
plot(x,type='p')
```



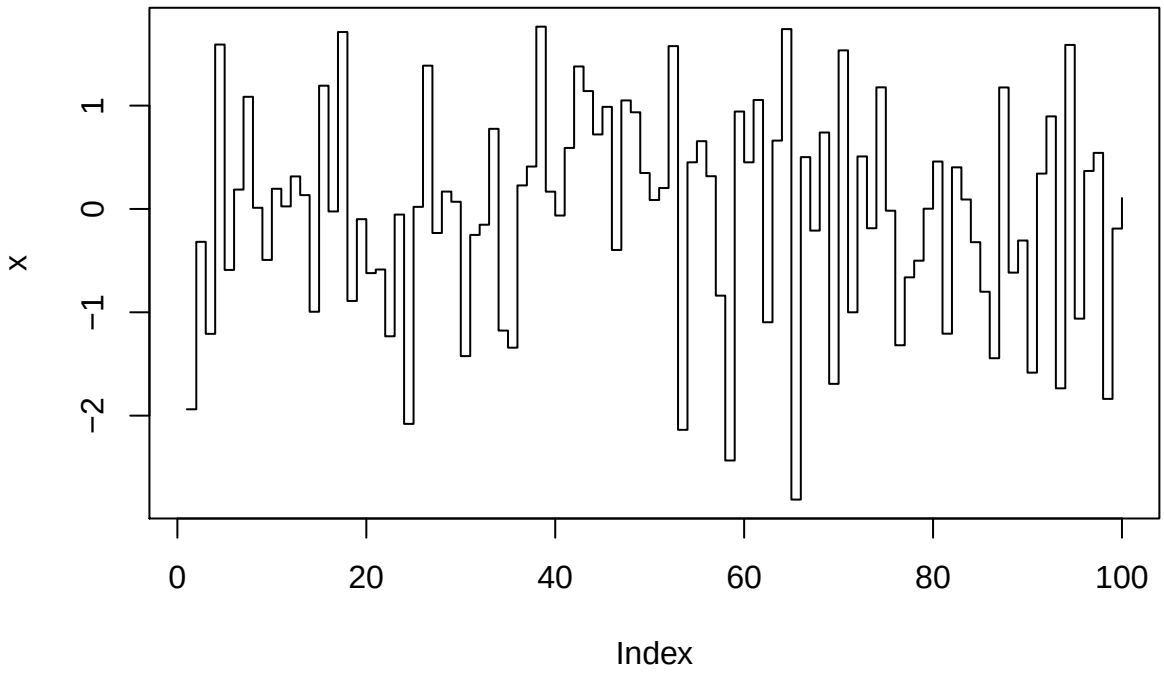

```
plot(x,type='b')
```




```
plot(x,type='h')
```

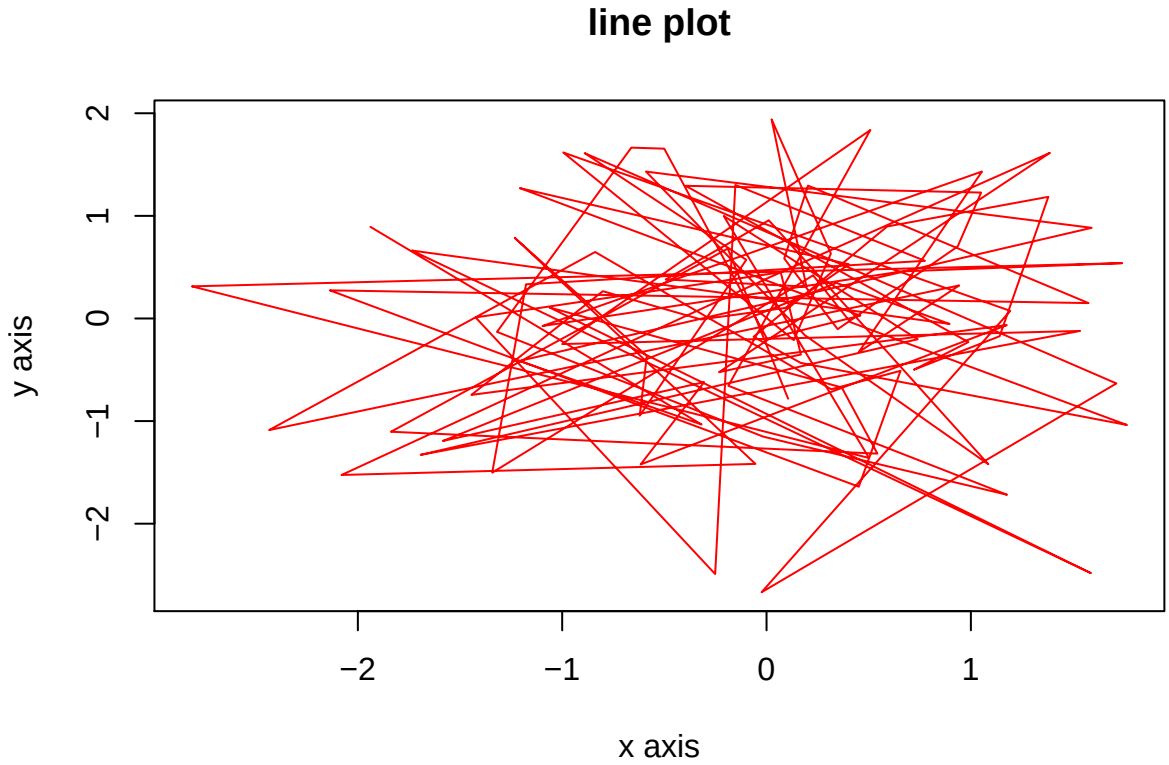



```
plot(x,type='s')
```

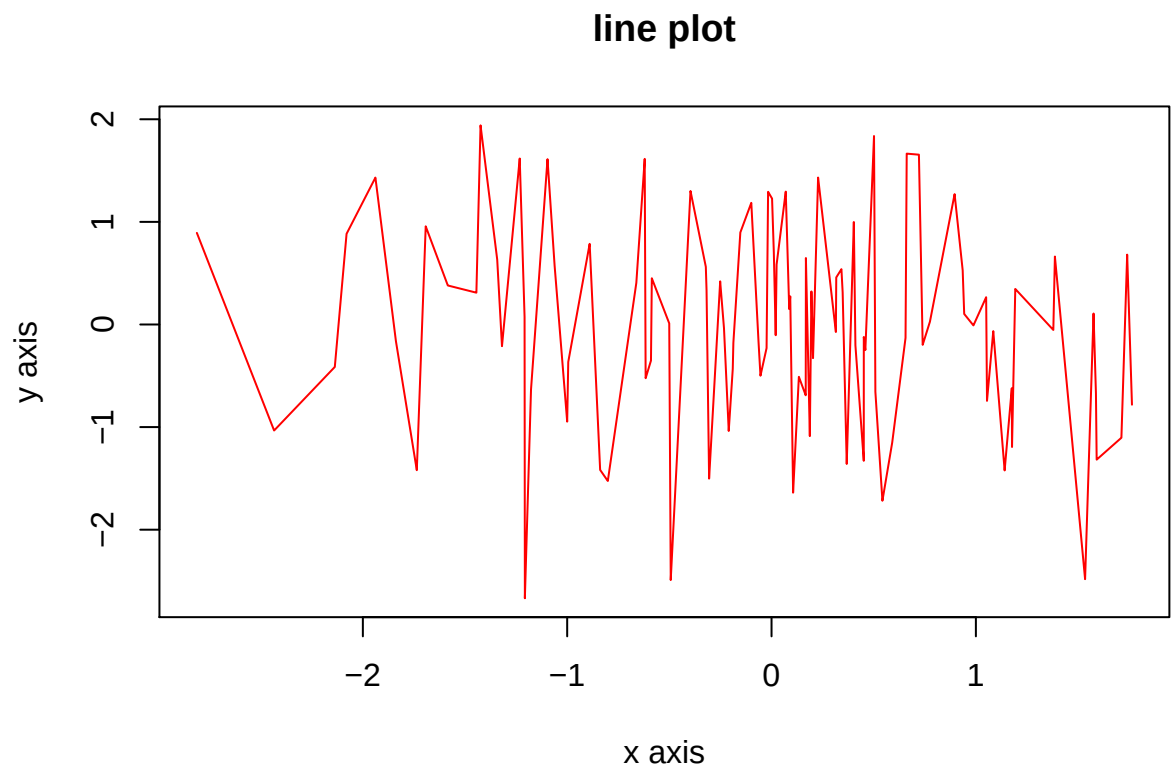


Annotate the plots

```
plot(x,y,type='l',xlab='x axis',ylab='y axis',main='line plot',col='red',font.main=2)
```



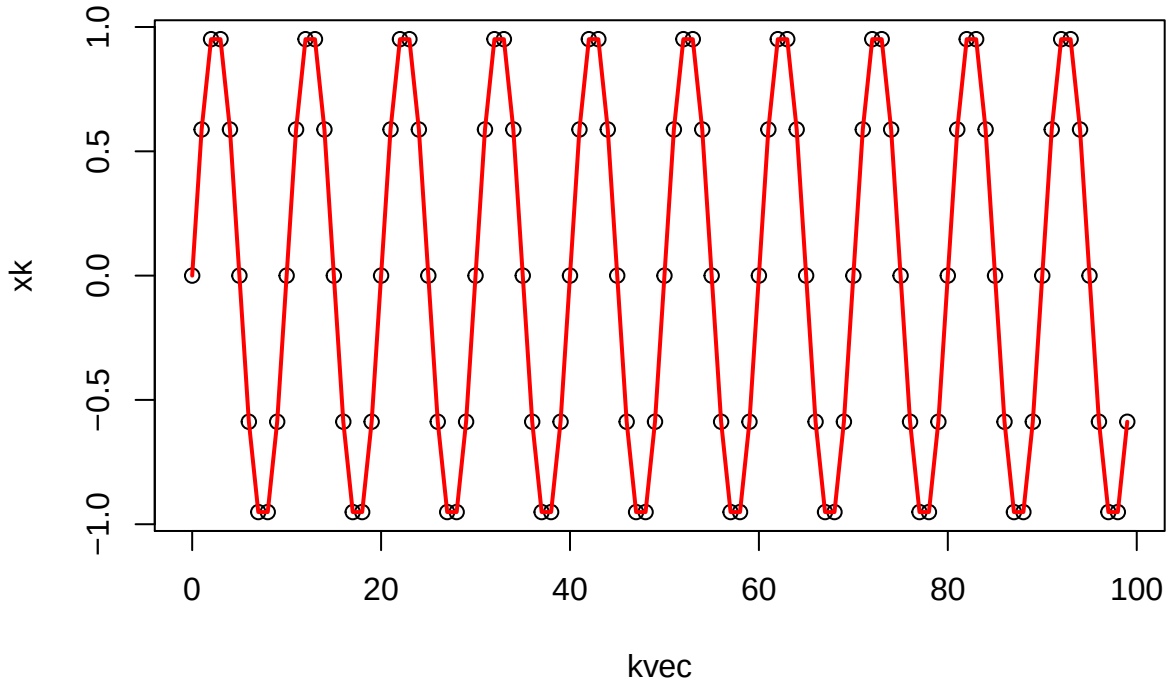

```
plot(sort(x),y,type='l',xlab='x axis',ylab='y axis',main='line plot',col='red',font.main=2)
```



###

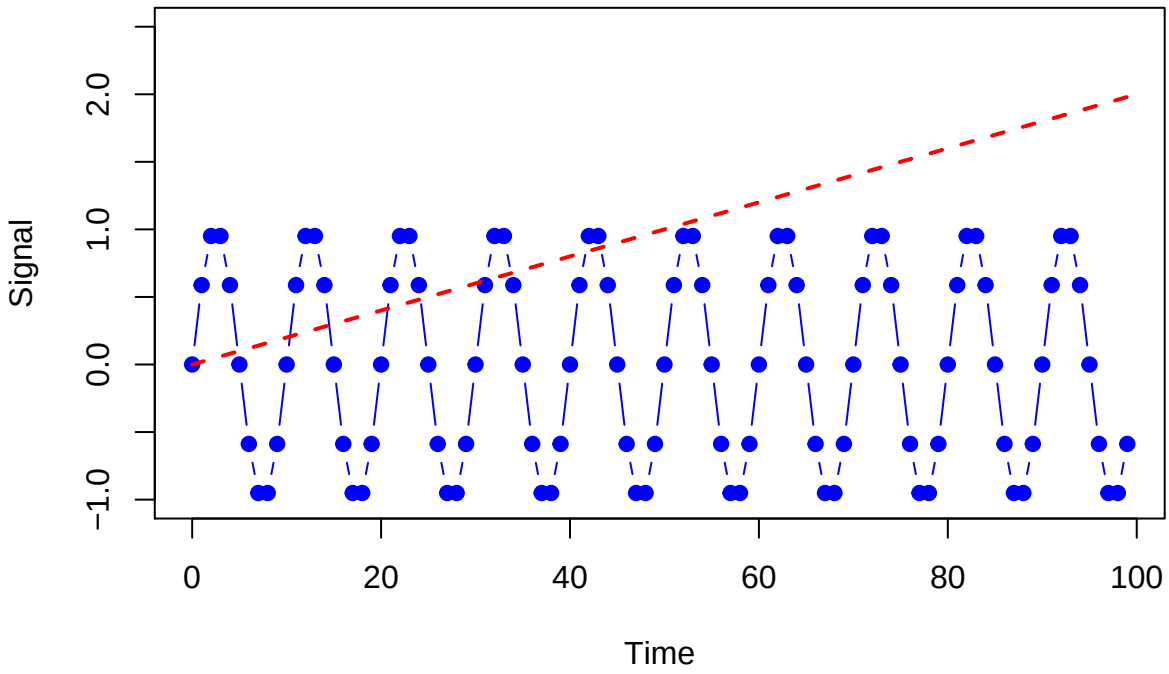
Overlays

```
kvec = (0:99)
xk = sin(2*pi*0.1*kvec)
plot(kvec,xk) # Cant reverse, first plot()
lines(kvec,xk,col='red',lwd=2)
```




```
# plot(kvec,xk,col='red',lwd=2,type='l')
```

```r
kvec = (0:99)
xk1 = sin(2*pi*0.1*kvec)  # Sinusoid
xk2 = 0.02*kvec # Line
plot(kvec,xk1,type='b',pch=19,ylim=c(-1,2.5),xlab='Time',ylab='Signal',col='blue')
lines(kvec,xk2,col='red',lty=2,lwd=2)
```

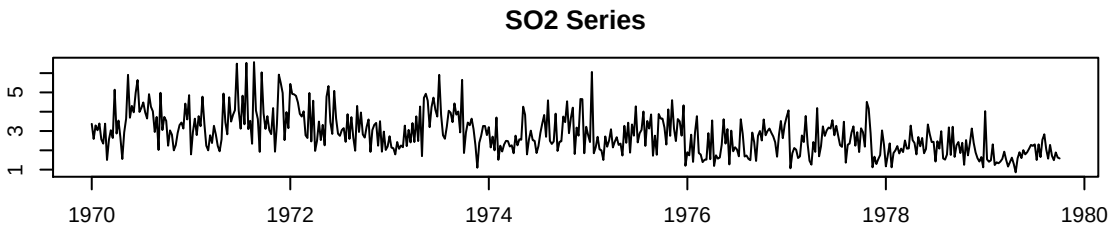


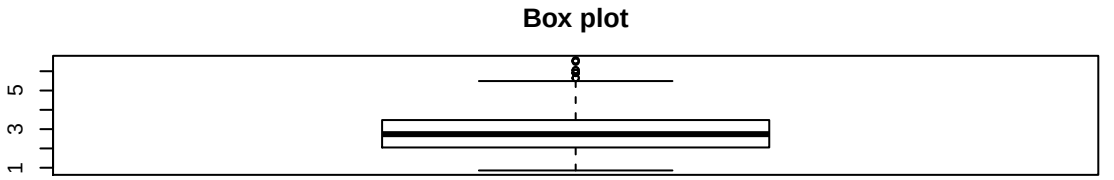

```r
legstr = c("Sine","Line")
# Look up how to place legends
```

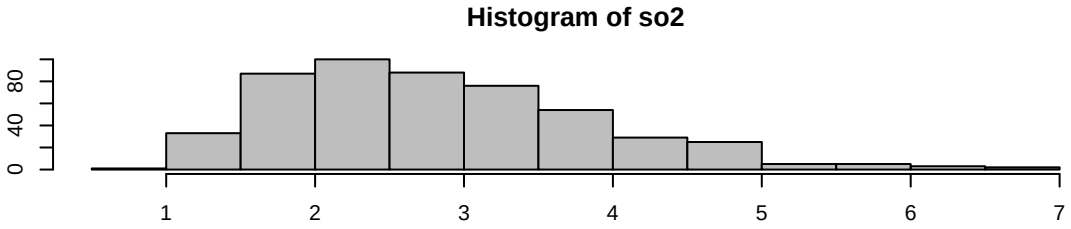
Multiple plots

```r
library(astsa)
```

```r
par(mfrow=c(3,1)) # 3 rows
par(mar=c(3,3,3,3)) # Margins LRTB, # lines
par(oma=c(1,1,1,1)) # Paddings
plot(so2,main="SO2 Series",xlab="Year")
boxplot(so2, main="Box plot")
hist(so2,col="grey")
```



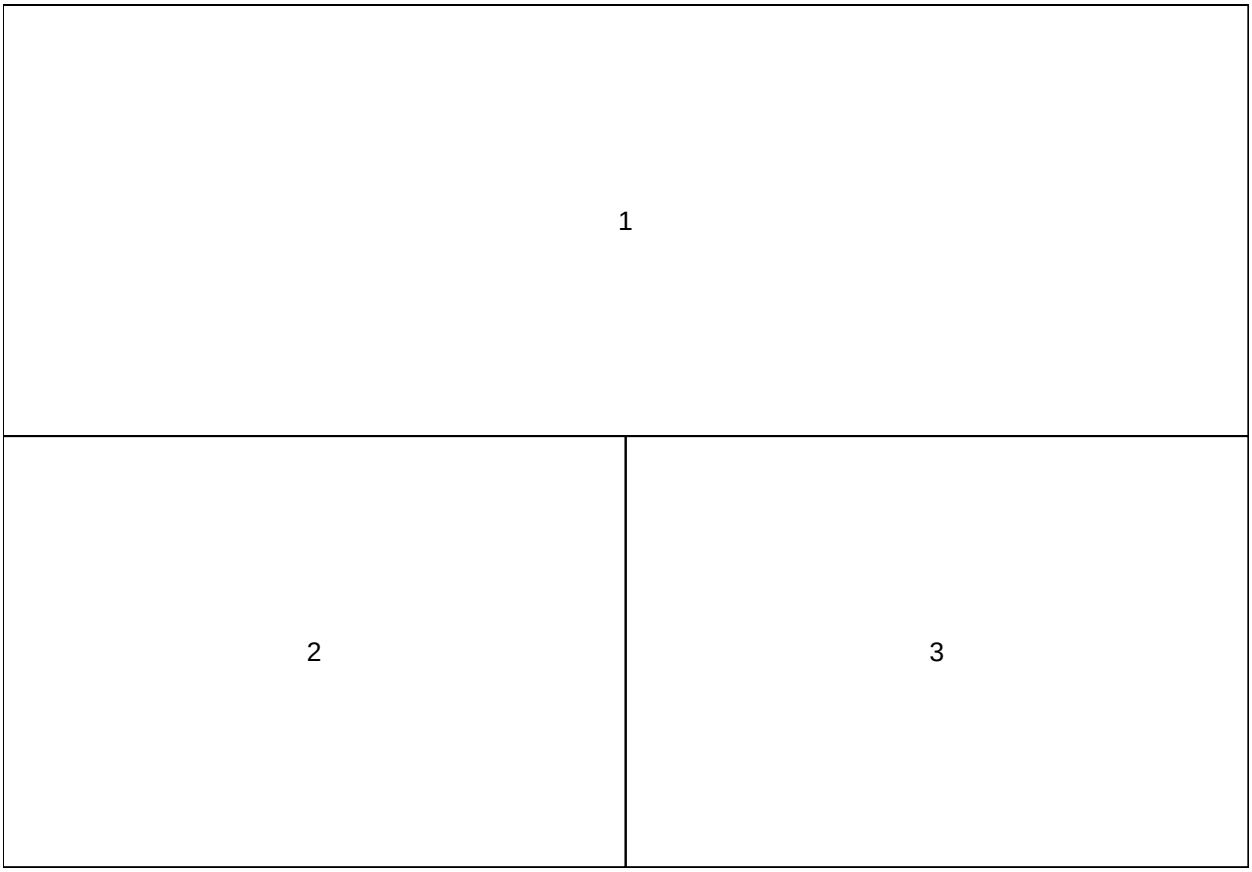




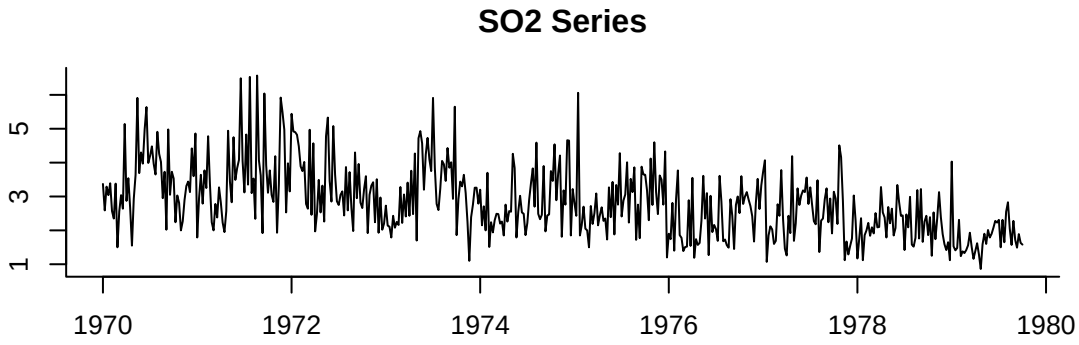
Fancier, using layout

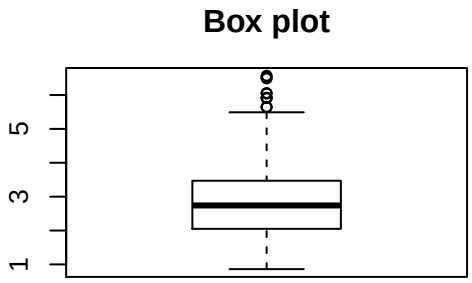
```
M <- rbind(c(1,1),c(2,3))
layout(M)
layout.show(2)
layout.show(3)
```

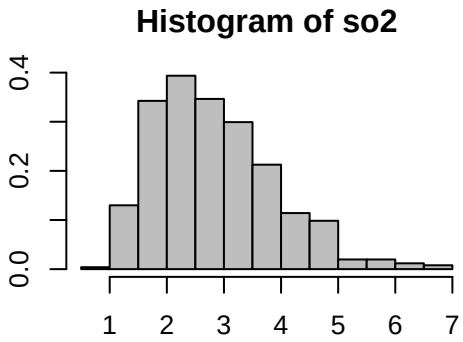

1
2




```
par(mar=c(3,3,3,3)) # Margins LRTB, # lines
par(oma=c(1,1,1,1)) # Paddings
plot(so2,main="SO2 Series",xlab="Year",bty='l')
boxplot(so2, main="Box plot")
hist(so2,col="grey",probability = T)
```







Same using split.screen

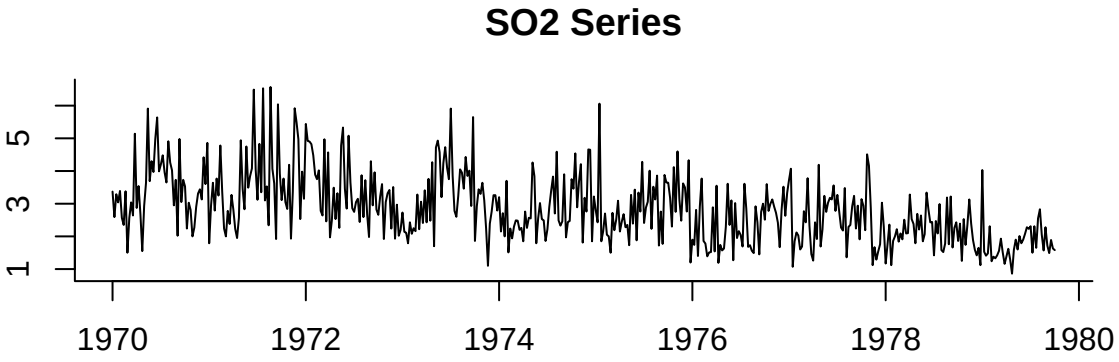
```
split.screen(c(2,1))
```

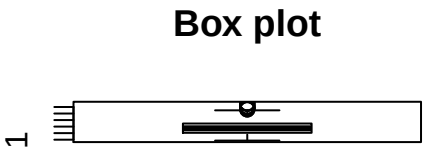
```
## [1] 1 2
```

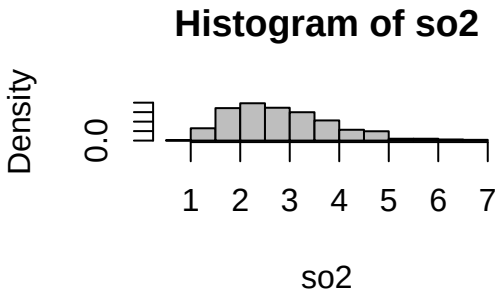
```
screen(1)
par(mar=c(3,3,3,3))
plot(so2,main="SO2 Series",xlab="Year",bty='l')
screen(2)
split.screen(c(1,2),screen=2)
```

```
## [1] 3 4
```

```
par(oma=c(1,1,1,1))
boxplot(so2, main="Box plot")
screen(4)
hist(so2,col="grey",probability = T)
```


SO2 Series
1
3
5
1970
1972
1974
1976
1978
1980


Box plot
1


Histogram of so2
Density
0.0
1
2
3
4
5
6
7
so2