CUHK RMSC4002 Tutorial 3

Benjamin Chan

October 9, 2018

Read in and Manipulate Data

The file `stock_2018.csv` contains information about the stock HSBC (0005), CLP (0002) and Cheung Kong (0001) from 1 Sep 2014 to 31 Aug 2018. Data are downloaded at Yahoo Finance (https://finance.yahoo.com/).

```r
# Read in data (a CSV file) under Dataset
d <- read.csv("./../Dataset/stock_2018.csv")

# as.ts: coerce an object to a time-series
t1 <- as.ts(d$HSBC)        # For stock HSBC (0005)
t2 <- as.ts(d$CLP)         # For stock CLP (0002)
t3 <- as.ts(d$CK)          # For stock Cheung Kong (0001)

# Compute daily percentage return
u1 <- (lag(t1)-t1)/t1      # lag: compute a lagged version of a time series
u2 <- (lag(t2)-t2)/t2
u3 <- (lag(t3)-t3)/t3

u <- cbind(u1, u2, u3)     # Combine into matrix u
```

Check for Multivariate Normal Distribution

- For univariate normal distribution $N(\mu, \sigma^2)$, the term

$$\left(\frac{x-\mu}{\sigma}\right)^2 = (x-\mu)(\sigma^2)^{-1}(x-\mu)$$

  measures the square of the distance from x to μ in standard deviaton units.
- For multivariate normal distribution $N_p(\mu, \Sigma)$, the term

$$(x-\mu)'\Sigma^{-1}(x-\mu)$$

  is the square of the generalized distance from x to μ, or the Mahalanobis distance.
- If $u = (u_1, u_2, u_3)'$ follows a multivariate normal distribution, then the quadratic form

$$d^2 = (u-\bar{u})'S^{-1}(u-\bar{u}),$$

  where $\bar{u}$ is the sample mean vector and S is the sample covariance matrix, follows approximately χ^2_3.
- In general, if $u = (u_1, \ldots, u_p)' \sim N_p(\mu_u, \Sigma_u)$, then d^2 should follow approximately a chi-squared distribution with p degrees of freedom.

```r
n2 <- nrow(u)                 # Number of rows in u
n1 <- n2-180+1                # Starting index: 180th obs before n2
u180 <- u[n1:n2,]             # Save the most recent 180 days to u180

(m <- apply(u180, 2, mean))   # Compute mean vector of u180
```

```
          u1            u2            u3 
-0.0002815488  0.0011469403 -0.0002769128 
```

```r
# Transform m into 180x3 matrix (each row is the column mean)
m <- matrix(m, nr = 180, nc = 3, byrow = T)

(s <- var(u180))              # Compute covariance matrix of u180
```

```
             u1           u2           u3
u1 8.798931e-05 2.593044e-05 5.134027e-05
u2 2.593044e-05 9.257047e-05 4.930806e-05
u3 5.134027e-05 4.930806e-05 1.103498e-04
```

```r
sinv <- solve(s)              # Compute inverse of s

dim(u180)                     # 180x3 matrix
```

```
[1] 180   3
```

```r
dim(m)                        # 180x3 matrix
```

```
[1] 180   3
```

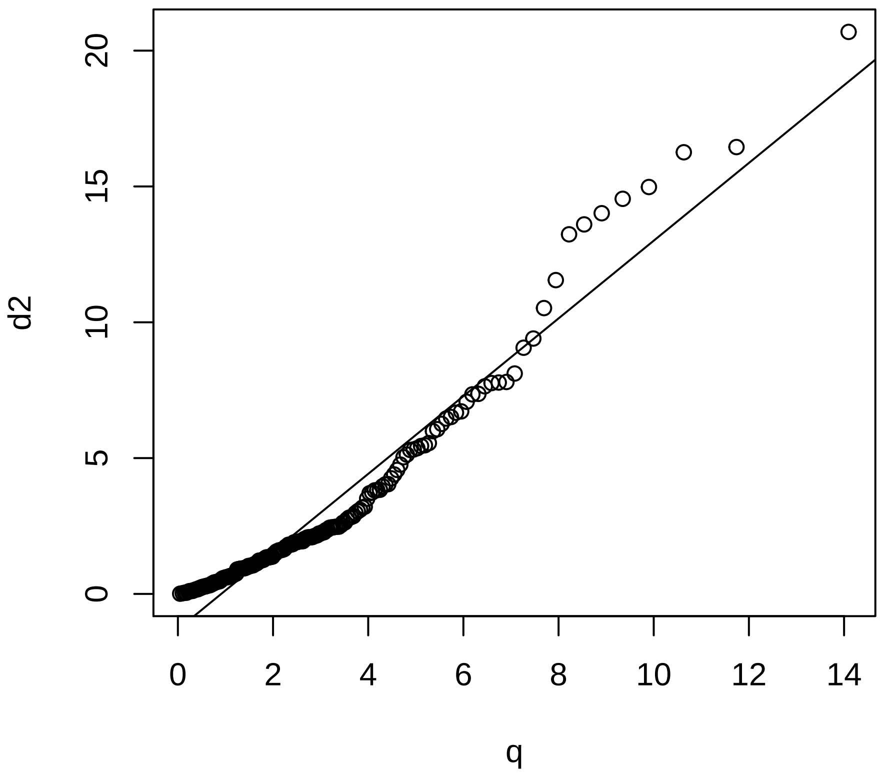
```r
# Compute the squared generalized distance
d2 <- diag((u180-m) %*% sinv %*% t(u180-m))
length(d2)                    # Length of d2 is 180
```

```
[1] 180
```

Remark: u180 and m are 180x3 matrices while sinv is a 3x3 matrix. Hence (u180-m) %*% sinv %*% t(u180-m) is a 180x180 matrix. Its diagonal is a vector of length 180 equal to the squared generalized distance from each observation to the sample mean.

```r
d2 <- sort(d2)                # Sort d2 in ascending order
i <- ((1:180)-0.5)/180        # Create a vector of percentiles
q <- qchisq(i, 3)             # Compute quantile of chisq(3)

par(mfrow = c(1, 1))
qqplot(q, d2)                 # QQ-chisquare plot
abline(lsfit(q, d2))          # Add least squares fit line
```




```
ks.test(d2, pchisq, 3)        # KS chisquare test for d2
```

```
    One-sample Kolmogorov-Smirnov test

data:  d2
D = 0.14609, p-value = 0.0009213
alternative hypothesis: two-sided
```

From the plot, the distribution of u180 is close to multivariate normal. However, the p-value is small, so formally speaking, you should reject the null hypothesis that d2 comes from a chi-squared distribution.

Famous statistician George Box said "All models are wrong but some are useful". Let us assume that $u = (u_1, u_2, u_3)'$ follows a trivariate normal distribution for the time being.

Generate Multivariate Normal Random Vector

- Note that any linear combination of normal random variables/vectors are also normally distributed.
- For univariate normal distribution, if

$$Z \sim N(0, 1),$$

then

$$X = \mu + \sigma Z \sim N(\mu, \sigma^2).$$

Reasoning:

$$E(X) = \mu + \sigma E(Z) = \mu$$

and

$$\mathrm{Var}(X) = \sigma^2 \mathrm{Var}(Z) = \sigma^2.$$

- For multivariate normal distribution, if

$$Z = (Z_1, \ldots, Z_p)' \sim N_p(0_p, I_p), \text{ i.e. } Z_i \overset{\text{i.i.d.}}{\sim} N(0,1)$$

and

$$C'C = \Sigma,$$

then

$$X = \mu + C'Z \sim N_p(\mu, \Sigma).$$

Reasoning:

$$E(X) = \mu + C'E(Z) = \mu$$

and

$$\mathrm{Cov}(X) = C'\mathrm{Cov}(Z)C = C'C = \Sigma.$$

- The matrix C is called the Cholesky decomposition of Σ. See https://en.wikipedia.org/wiki/Cholesky_decomposition for more details.

```
# Setting random seed in simulation ensures that
# the same set of pseudo random numbers is generated each time.
# It is an important element in reproducible research.
set.seed(7)                     # Set random seed
mu <- apply(u180, 2, mean)      # Compute mean vector
sigma <- var(u180)              # Compute covariance matrix
C <- chol(sigma)                # Cholesky decomposition of sigma

# Sanity check: C'C = sigma
t(C)%*%C
```

```
             u1           u2           u3
u1 8.798931e-05 2.593044e-05 5.134027e-05
u2 2.593044e-05 9.257047e-05 4.930806e-05
u3 5.134027e-05 4.930806e-05 1.103498e-04
```

```
sigma
```

```
             u1           u2           u3
u1 8.798931e-05 2.593044e-05 5.134027e-05
u2 2.593044e-05 9.257047e-05 4.930806e-05
u3 5.134027e-05 4.930806e-05 1.103498e-04
```

```
Tm <- cbind(t1,t2,t3)           # Combine t1, t2, t3 to form Tm
T0 <- Tm[nrow(Tm),]             # Set T0 to be the most recent prices

# Simulate prices for the future 90 days
for (i in 1:90) {
    Z <- rnorm(3)               # Generate normal random vector
    v <- mu + t(C)%*%Z          # Transform to multivariate normal
    T1 <- T0*(1 + v)            # Predict new stock prices
    Tm <- rbind(Tm, t(T1))      # Append T1 to Tm
    T0 <- T1                    # Update T0
}
```

Plot Simulation Results

```r
library(plotly)               # Create Interactive Web Graphics via 'plotly.js'
library(tidyr)                # Easily Tidy Data with 'spread()' and 'gather()' Functions
library(dplyr)                # A Grammar of Data Manipulation

colnames(Tm) <- c("HSBC", "CLP", "CK")      # To be appeared in plot

# %>%: pipe operator
newseries <- as.data.frame(Tm) %>%
    gather(type, value) %>%
    mutate(time = rep(time(Tm), 3))

# Alternatively
temp1 <- as.data.frame(Tm)
temp2 <- gather(temp1, type, value)
temp3 <- mutate(temp2, time = rep(time(Tm), 3))

head(newseries)
```

```
  type    value time
1 HSBC 65.69888    1
2 HSBC 65.38452    2
3 HSBC 65.73816    3
4 HSBC 65.46311    4
5 HSBC 65.69888    5
6 HSBC 65.18806    6
```

```r
head(temp3)
```

```
  type    value time
1 HSBC 65.69888    1
2 HSBC 65.38452    2
3 HSBC 65.73816    3
4 HSBC 65.46311    4
5 HSBC 65.69888    5
6 HSBC 65.18806    6
```

```r
tail(newseries)
```

```
     type    value time
3232   CK 94.32474 1074
3233   CK 95.11284 1075
3234   CK 94.54062 1076
3235   CK 95.48517 1077
3236   CK 95.62374 1078
3237   CK 96.34513 1079
```

```r
tail(temp3)
```

```
     type    value time
3232   CK 94.32474 1074
3233   CK 95.11284 1075
3234   CK 94.54062 1076
3235   CK 95.48517 1077
3236   CK 95.62374 1078
```

```
3237   CK 96.34513 1079
```

```
# plot_ly(x = newseries$time, y = newseries$value, color = newseries$type, mode = 'lines')
# plot_ly is not supported by LaTeX. Please refer to HTML version for the plot.
```