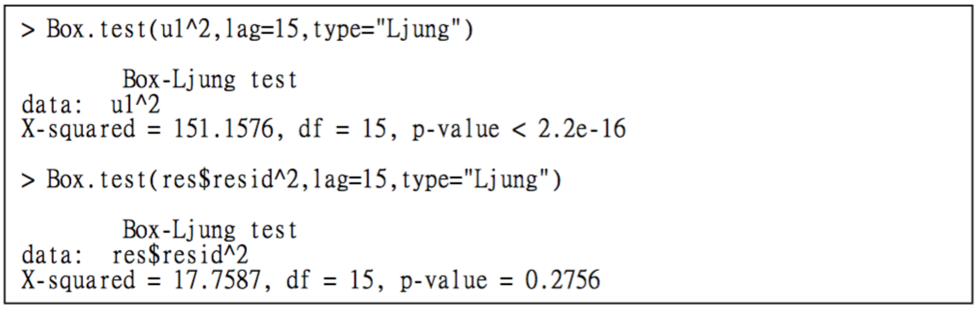
RMSC4002 Tutorial 4

Chapter 2

October 10, 2017

1 Box-Ljung test



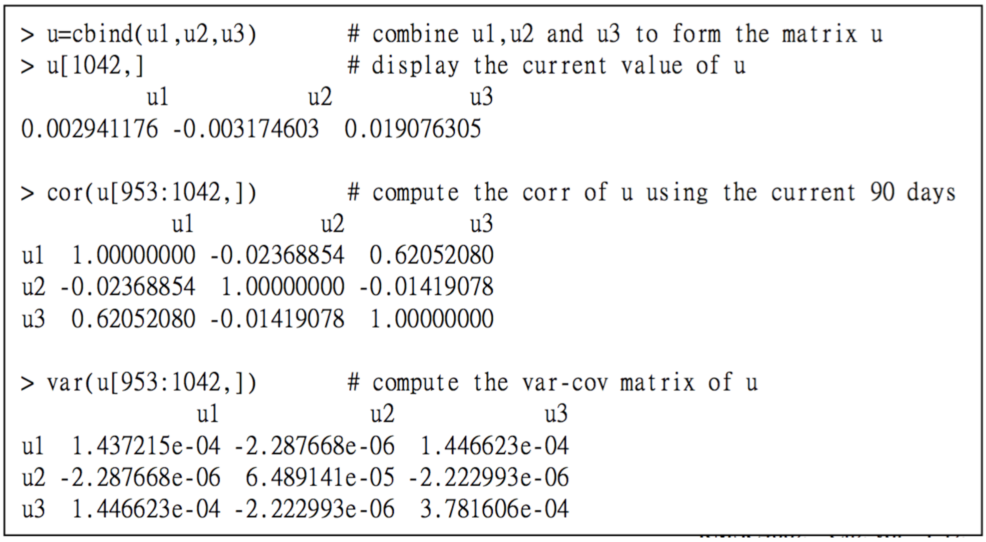
```
> Box.test(u1^2,lag=15,type="Ljung")

        Box-Ljung test
data:  u1^2
X-squared = 151.1576, df = 15, p-value < 2.2e-16

> Box.test(res$resid^2,lag=15,type="Ljung")

        Box-Ljung test
data:  res$resid^2
X-squared = 17.7587, df = 15, p-value = 0.2756
```

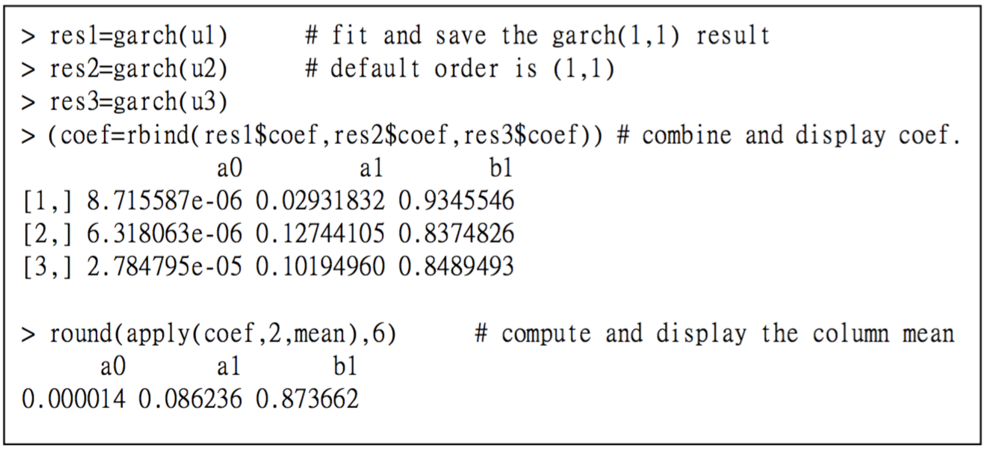
2 The Estimation of Correlation



```
> u=cbind(u1,u2,u3)       # combine u1,u2 and u3 to form the matrix u
> u[1042,]                # display the current value of u
         u1           u2          u3
0.002941176 -0.003174603  0.019076305

> cor(u[953:1042,])       # compute the corr of u using the current 90 days
            u1          u2          u3
u1  1.00000000 -0.02368854  0.62052080
u2 -0.02368854  1.00000000 -0.01419078
u3  0.62052080 -0.01419078  1.00000000

> var(u[953:1042,])       # compute the var-cov matrix of u
              u1            u2            u3
u1  1.437215e-04 -2.287668e-06  1.446623e-04
u2 -2.287668e-06  6.489141e-05 -2.222993e-06
u3  1.446623e-04 -2.222993e-06  3.781606e-04
```



```
> res1=garch(u1)      # fit and save the garch(1,1) result
> res2=garch(u2)      # default order is (1,1)
> res3=garch(u3)
> (coef=rbind(res1$coef,res2$coef,res3$coef)) # combine and display coef.
                a0         a1        b1
[1,] 8.715587e-06 0.02931832 0.9345546
[2,] 6.318063e-06 0.12744105 0.8374826
[3,] 2.784795e-05 0.10194960 0.8489493

> round(apply(coef,2,mean),6)      # compute and display the column mean
      a0       a1       b1
0.000014 0.086236 0.873662
```

$$\omega = 0.000014 \quad \alpha = 0.086236 \quad \beta = 0.873662$$

$$\sigma_1^2 = \omega + \alpha(0.00294)^2 + \beta(0.0001437) = 0.00014$$

$$\sigma_2^2 = \omega + \alpha(\text{-}0.00317)^2 + \beta(0.0000649) = 0.000072$$

$$\sigma_3^2 = \omega + \alpha(0.01908)^2 + \beta(0.0003782) = 0.000376$$

$$\mathrm{cov}_{12} = \omega + \alpha(0.00294)(-0.00317) + \beta(-0.00000229) = 0.0000112$$

$$\mathrm{cov}_{13} = \omega + \alpha(0.00294)(0.01908) + \beta(0.000145) = 0.000135$$

$$\mathrm{cov}_{23} = \omega + \alpha(-0.00317)(0.01908) + \beta(-0.00000222) = 0.000012$$

$$\rho_{12} = \mathrm{cov}_{12}/(\sigma_1\sigma_2) = 0.1117$$

$$\rho_{13} = \mathrm{cov}_{13}/(\sigma_1\sigma_3) = 0.5886$$

$$\rho_{23} = \mathrm{cov}_{23}/(\sigma_2\sigma_3) = 0.07353$$

3 GARCH using EXCEL

The EXCEL we are using does not have the built-in GARCH or EWMA function. However, we can use the solver function to find the MLE of GARCH and EWMA. Let us illustrate this by fitting a GARCH(1,1) model to estimate the volatility of HSBC.

- Set up the stock price of HSBC in B2:B1044 and the corresponding percentage return ui in C3:C1044.
- Set up the cells J2, K2 and L2 for the parameters ω, α and β. Enter some initial values to start with. For example, set $\omega = 0.00001$, $\alpha = 0.05$, $\beta = 0.9$.
- In cell M2, enter the formula =J2/(1-K2-L2) for computing the long run variance rate using the current values of ω, α and β.
- In cell D3, enter the formula =C3ˆ2. This serves as the initial value for v.
- In cell D4, enter the formula =$J$2+$K$2*C3ˆ2+$L$2*D3. This is the GARCH(1,1) model.
- Copy the formula in cell D4 to D5:D1044.
- In cell E4, enter the formula E3 =-LN(D3)-C3ˆ2/D3. This is the first term in the summation of the log-likelihood function.
- Copy the formula in E3 to E4:E1044.

- In cell I6, enter the formula =SUM(E3:E1044). This is the value of the log-likelihood function we want to maximize.
- Now use the solver function in the Data menu. Specify I6 as the target cell and J2:L2 as the variable cells. Choose the max option and solve it. The parameter values in J2:L2 as well as columns C, D and E will change such that I6 is maximized.
- The column D will be the final series of the estimated variance rate. If we want to plot the volatilities series, we can create the column F which is the square root of D and plot the volatilities in column F.

Variance Targeting Technique:

- Set up the cells K22 and L22 for α and β with some initial values as before.
- Enter =VAR(C3:C1044) in M22 for the long-run variance rate.
- Enter =M22*(1-K22-L22) in J22 for ω.
- Set up columns G and H for v and the terms in the log-likelihood function as before but using new α and β.
- Enter =SUM(H3:H1044) in I10 for the cell to be maximized.
- Use the solver function as before to maximize I10.

Remarks:(How to load Solver Add-in?)

- In Excel2010 and later goto **File>Options**.
- Click **Add-ins**, and then in the **Manage** box, select **Excel Add-ins**.
- Click **Go**.
- In the **Add-ins available** box, select the **Solver Add-in** check box, and then click **OK**.
- After you load the Solver Add-in, the **Solver** command is available in the **Analysis** group on the **Data** tab.

4 R Program Instruction for Ch1&2

4.1 Demonstration on Stock Price Simulation, Moving Standard Deviation

(the program will not be posted to elearn system.)

- Load data from file & save data into file.
- Compute relative return of stocks.
- Find sample covariance matrix and sample mean of several stocks.
- Simulate Stock prices.
- Find value at risk of a portfolio using simulation results.
- Compute moving volatility.

4.2 Useful functions

(Will not be covered in the tutorial. For reference only.)

- help(*a*): show the manual of function *a*.
- help(package="*a*"): show the manual book of a library *a*.
- ??*a*: search manual of some key words with pattern *a*.
- install.packages("*a*"): install a library *a*.
- library(*a*): load a library *a*.
- save(...,file="a"): save R objects in a Rdata file *a* (*.rdata).
- load("*a*"): load Rdata file *a*.
- getwd(): show the current path running R.
- setwd(*a*): set the path to some local address *a*.
- str(*a*): display compacted structure or all attributes in an object *a*.
- ls.str(*a*): display the structure or all attributes in an object *a*.
- ls(): display names of all user-defined objects stored in console.
- object.size(*a*): show the size of an object *a*. Memory management in R is poor, so if you have 10000x10000 matrix, or some objects with possible huge size, check whether your memory is enough.
- rm(*a*): remove object *a*.
- head(*a*): show the first 6 elements in an object *a*.
  - If *a* is a numerical vector, the result is the first 6 numbers.
  - If *a* is a matrix or data frame, the result is the first 6 rows.
  - If *a* is a list, the result is the first 6 elements of the list.
- head(...,n): show the first n elements.
- tail(*a*), tail(*a*,n): show the last 6 or n elements of *a*.
- date(): show current time.
- system.time({...}): show the CPU time used for some expression.
- apply(*a*,*margin*,*expr*): apply functions on array margin.