Tips on Using *R*

Jin-Lung Lin

Lastly updated: September, 2009

1 What is R?

R is an open-source (GPL) object oriented statistical package started by Robert Gentleman and Ross Ihaka of the Statistics Department of the University of Auckland in 1995 and currently maintained by the R core-development team, a hard-working, international team of volunteer developers. Syntactically and functionally R is very similar (if not identical) to S/Splus, the popular statistics packages.

2 Why using R?

According to Julian J. Faraway and many others, the advantages of R are:

1. Versatility
   R is a also a programming language and user is not limited by the procedures that are preprogrammed by a package. It is relatively easy to program new methods in R .

2. Interactivity
   Data analysis is inherently interactive and R allows us to make changes interactively.

3. Portability
   R is open-source software and may be freely redistributed. Linux, Macintosh, Windows and other UNIX versions are maintained and can be obtained from the R-project at www.r-project.org.

4. Popularity
   R is most popular with researchers in Statistics and finance. There are already a lot of library for all areas in statistics and the number keeps increasing every day.

5. Interface
   Foreign language interface is available.

6. Low (or no) cost
   R is free (Who says there is no freee lunch ?).

Disvantages of R

1. It is not so easy to learn to use R as it is command-driven rather than menu-driven.
2. Inefficient in handling large dataset as compared with SAS.

3 Obtaining R

1. browse http://www.r-project.org/
2. click CRAN
3. select
   http://cran.csie.ntu.edu.tw/
4. click Windows (95 and later)
5. click base
6. download R-2.9.2-win32.exe (As of Sep. 2009)
7. also download appropriate Rblas.dll from /bin/windows/contrib/ATLAS. Be sure that you choose the right CPU version for your computer.

4 Installing R

1. Execute R-2.9.2-win32.exe. I accept all the defaults during installation process.
2. Replace the original C:/Program Files/R/R-2.9.2/bin/Rblas.dll($<$200K) with the Rblas.dll (about 2Meg) optimized for your CPU. This will typically speed up the computation twice faster.
3. Edit C:/Program Files/R/R-2.7.2/etc/Rconole by changing 'language = ' to 'language = en' provided you like the English environment.
4. Activate *R*, click on the button 'Package' and select 'Install packages'. Select 'http://cran.csie.ntu.edu.tw/' as reposit site and install all packages. It takes about 20 minutes.

5 Installing Tinn-R

Tinn-R is a small editor written specifically for running R. Obtaining Tinn-R from http://sourceforge.net/projects/tinn-r. Just run the setup file. The newest version is http://sourceforge.net/projects/ti
r 2.0.0.7 relased on Sep 5, 2008.

6 Manuals for R

Several excellent manuals are available at www.r-project.org

1. An Introduction to R:
   under 'Manuals' at Documentation

2. Simple R
   under 'contributed documentation'

3. Econometrics in R
   under contributed documentation

4. Some-useful-R-pointers
   available at C:/Program Files/R/R-2.5.1/library/bayesm/doc

5. A Brief Guide to R for Beginners in Econometrics by Mahmood Arai
   available at http://people.su.se/m̃a/R_intro/

6. R by examples
   http://www.mayin.org/ajayshah/KB/R/index.html

7 Packages for R

1. R functions for doing forecasting written by Rob Hyndman
   http://robjhyndman.com/software/forecast

2. For finance
   Rmetrics at http://www.itp.phys.ethz.ch/econophysics/R/

3. Task view on finance
   http://cran.r-project.org/src/contrib/Views/Finance.html

4. Task view on Econometrics
   http://cran.r-project.org/src/contrib/Views/Econometrics.html

5. task view on Bayesian
   http://cran.r-project.org/src/contrib/Views/Bayesian.html

6. task view on graphics
   http://cran.r-project.org/src/contrib/Views/Graphics.html

7. task view on Multivariate
   http://cran.r-project.org/src/contrib/Views/Multivariate.html

8 Obtaining help within R

1. *help* or *?*
   A useful trip is to type '?command', say ?rnorm', marked the example codes, and then paste them into R.

2. *help.search, apropos, or find*
   Useful when not sure which command to look at. For example,
   apropos("cor")

```
 [1] ".__C__recordedplot" "cancor"             "cor"
 [4] "cor.test"           "cov2cor"            "Harman23.cor"
 [7] "Harman74.cor"       "recordGraphics"     "recordPlot"
```

3. Searching for help over the net
   For example, *RSiteSearch("Kalman Filter")*

4. Read FAQs at www.r-project.org

5. Use the R site search at www.r-project.org for answers.

9 Language fundamentals

9.1 vector, matrix, and manupulations

```
> f1 <- c(7.5,6,5)
```

```
>  f1
[1] 7.5 6.5
> t(f1)
 [,1] [,2]
[1,]  7.5  6.5
> f2 = 1:5
> f2
[1] 1 2 3 4 5
> f2[c(3,4)]
[1] 3 4
> f2[-3]
[1] 1 2 4 5

> A=matrix(1:12,ncol=3)
> B=matrix(1:12,ncol=3,byrow=TRUE)
> A
      [,1] [,2] [,3]
 [1,]    1    5    9
 [2,]    2    6   10
 [3,]    3    7   11
 [4,]    4    8   12

> B
     [,1] [,2] [,3]
 [1,]    1    2    3
 [2,]    4    5    6
 [3,]    7    8    9
 [4,]   10   11   12

> B[1:2,2:3]
     [,1] [,2]
[1,]    2    3
[2,]    5    6

> C=A %*% t(B)
> C
```

```
     [,1] [,2] [,3] [,4]
 [1,]   38   83  128  173
 [2,]   44   98  152  206
 [3,]   50  113   76  239
 [4,]   56  128  200  272

   A*B
     [,1] [,2] [,3]
 [1,]    1   10   27
 [2,]    8   30   60
 [3,]   21   56   99
 [4,]   40   88  144

 > A[A>6]
[1]  7  8  9 10 11 12

> A^2
  [,1] [,2] [,3]
 [1,]    1   25   81
 [2,]    4   36  100
 [3,]    9   49  121
 [4,]   16   64  144


> D = (A*A)[1:3,]
> D
 [,1] [,2] [,3]
[1,]    1   25   81
[2,]    4   36  100
[3,]    9   49  121

 > round(3.2)
 [1] 3

 > A[2*round(A/2)==A]
 [1]  2  4  6  8 10 12
```

```
 > A[round(A[,2]/2)*2==A[,2],]
     [,1] [,2] [,3]
[1,]    2    6   10
[2,]    4    8   12


 > A[A[,2] %in% c(5,8),] # keep the rows where the second column is either 5
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    4    8   12

> A[!A[,2] %in% c(5,8),] # remove the rows where the second column is either
     [,1] [,2] [,3]
[1,]    2    6   10
[2,]    3    7   11



> solve(D)
solve(D)
        [,1]     [,2]    [,3]
 [1,]  1.0625 -1.84375  0.8125
 [2,] -0.8125  1.18750 -0.4375
 [3,]  0.2500 -0.34375  0.1250
> D %*% solve(D)
 [,1]          [,2]          [,3]
 [1,]  1.000000e+00 8.881784e-16  3.885781e-16
 [2,] -4.440892e-16 1.000000e+00 -2.220446e-16
 [3,] -1.110223e-16 4.440892e-15 1.000000e+00

> X0=matrix(rnorm(300),ncol=3)
> X0[1:10,1:3]
> X[1:10,1:3]
            [,1]       [,2]       [,3]
 [1,] -0.4902088  0.1832561  0.6332573
```

```
 [2,]  1.3560325 -0.0926615 -0.8946232
 [3,] -0.5155720 -0.1521368 -0.4719691
 [4,] -1.5855754  1.0109394 -1.1772363
 [5,]  0.6012783 -1.0358522 -0.6055515
 [6,] -0.4393713  1.4475326 -0.8644015
 [7,] -1.5715612 -1.0192620 -1.0258046
 [8,] -0.8501274 -1.5664882 -1.3506368
 [9,] -0.9879753 -0.7191517  0.5440574
[10,]  0.2217093  0.1398276 -0.4755406

> X=cbind(1,X0)
> X
 X[1:10,1:4]
      [,1]       [,2]       [,3]       [,4]
 [1,]    1 -0.4902088  0.1832561  0.6332573
 [2,]    1  1.3560325 -0.0926615 -0.8946232
 [3,]    1 -0.5155720 -0.1521368 -0.4719691
 [4,]    1 -1.5855754  1.0109394 -1.1772363
 [5,]    1  0.6012783 -1.0358522 -0.6055515
 [6,]    1 -0.4393713  1.4475326 -0.8644015
 [7,]    1 -1.5715612 -1.0192620 -1.0258046
 [8,]    1 -0.8501274 -1.5664882 -1.3506368
 [9,]    1 -0.9879753 -0.7191517  0.5440574
[10,]    1  0.2217093  0.1398276 -0.4755406

> beta=c(2,0.1,-2,10)
> y = X%*% beta + 0.1 *rnorm(100)
> b= solve(t(X) %*% X) %*% t(X) %*% y
> b
 [,1]
 [1,]  1.9995154 [2,]  0.1167715 [3,] -1.9823865 [4,] 10.0072089
```

9.2 Reading Data and Data structure

- *read.table*

- *scan*

```
Content of data.txt

 UNIT Y X1 X2
 A 1 0.23815 0.4373
 A 2 0.55508 0.47938
 A 3 3.03399 -2.17571
 A 4 -1.49488 1.66929
 B 10 -1.74019 0.35368
 B 9 1.40533 -1.2612
 B 8 0.15628 -0.27751
 B 7 -0.93869 -0.0441
 B 6 -3.06566 0.14486

> df=read.table("data.txt",header=TRUE)
> df
UNIT Y X1 X2
 1 A 1 0.23815 0.43730 2
 2 A 2 0.55508 0.47938 3
 3 A 3 3.03399 -2.17571 4
 4 A 4 -1.49488 1.66929 5
 5 B 10 -1.74019 0.35368 6
 6 B 9 1.40533 -1.26120 7
 7 B 8 0.15628 -0.27751 8
 8 B 7 -0.93869 -0.04410 9
 9 B 6 -3.06566 0.14486 1

> df$Y
[1] 1 2 3 4 10 9 8 7 6
> mode(df$Y)
[1] "numeric"
> df[,2]
 [1] 1 2 3 4 10 9 8 7 6

> lmout=lm(Y ~ X1 + X2, data=df)
```

```
> names(lmout)
[1] "coefficients" "residuals" "effects" "rank" [5] "fitted.values"
"assign" "qr" "df.residual" [9] "xlevels" "call" "terms" "model"
> print(lmout)
Call: lm(formula = Y ~ X1 + X2, data = df)

Coefficients: (Intercept) X1 X2 5.084 -1.485 -2.221

> summary(lmout)
Call: lm(formula = Y ~ X1 + X2, data = df)
Residuals: Min 1Q Median
3Q Max -3.3149 -2.4101 0.4034 2.5319 3.2022
Coefficients: Estimate
Std. Error t value Pr(>|t|) (Intercept) 5.0839 1.0194 4.987 0.00248
** X1 -1.4851 0.8328 -1.783 0.12481 X2 -2.2209 1.3820 -1.607 0.15919
---
 Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
 Residual standard error: 2.96 on 6 degrees of freedom
 Multiple R-Squared: 0.3607, Adjusted R-squared: 0.1476
 F-statistic: 1.693 on 2 and 6 DF, p-value: 0.2612

> ls()
[1] "df" "lmout" This doesnt tell us too much about the objects.



> str(df)
'data.frame': 9 obs. of 4 variables:
 $ UNIT: Factor w/ 2 levels "A","B": 1 1 1 1 2 2 2 2 2
 $ Y : int 1 2 3 4 10 9 8 7 6
 $ X1 : num 0.238 0.555 3.034 -1.495 -1.740 ...
 $ X2 : num 0.437 0.479 -2.176 1.669 0.354 ...

> df$X1
[1] 0.23815 0.55508 3.03399 -1.49488 -1.74019 1.40533 0.15628 -
0.93869 -3.06566
> length(df$X1)
```

```
[1] 9
> is.vector(df$X1)
[1] TRUE

  ldata=list(A=dfmat[dfmat[,1]=="A",2:4],B=dfmat[dfmat[,1]=="B",2:4])
> ldata
 $A Y X1 X2
 1 " 1" " 0.23815" " 0.43730"
 2 " 2" " 0.55508" " 0.47938"
 3 " 3" " 3.03399" "-2.17571"
 4 " 4" "-1.49488" " 1.66929"

 $B Y X1 X2
 5 "10" "-1.74019" " 0.35368"
 6 " 9" " 1.40533" "-1.26120"
 7 " 8" " 0.15628" "-0.27751"
 8 " 7" "-0.93869" "-0.04410"
 9 " 6" "-3.06566" "0.14486"

> ldata[1]
  Y X1 X2
 1 " 1" " 0.23815" " 0.43730"
 2 " 2" " 0.55508" " 0.47938"
 3 " 3" " 3.03399" "-2.17571"
 4 " 4" "-1.49488" " 1.66929"

> lmout=NULL
> for (i in 1:2) {
+ lmout[[i]]=lm(Y ~ X1+X2,data=ldatadf[[i]])
+ print(lmout[[i]]) + }

Call: lm(formula = Y ~ X1 + X2, data = ldatadf[[i]])

 Coefficients:
  (Intercept) X1 X2 4.494 -2.860 -3.180
```

```
  Call:
  lm(formula= Y ~ X1 + X2, data = ldatadf[[i]])

  Coefficients: (Intercept) X1 X2
  9.309 1.051 1.981
```

9.3 Programming in R

Task 1: Sum $1^2 + 2^2 + \cdots + 500^2$
Solution 1:

```
   > sum=0
>    for (i in 1:500)
 +    {
 +     sum=sum+i*i
 +    }
>    sum;
[1] 41791750
```

Solution 2:

```
> sum((1:500)^2)
 [1] 41791750
```

Task 2: Generate a random 10 matrix A, find the column sum.
Solution

```
   > A = matrix(rnorm(8*10),ncol=8)
   > apply(A,2,sum)
  [1]  -0.4679369  6.0971916 -3.0929920 -2.1007340  0.8761865 -1.8058064 -1
```

Task 3: Write a R program to visualize the working process of central limit theorem.

```
  # Central Limit theorem
 # Written for Advanced Statistics, Econ. NTU
 # Jin-Lung Lin, April 2007
set.seed(12345678); nobs=500; nrep=500; n0=10; p=.25; muS=n0*p;
lambda0=1.5; muEx=lambda0; stdEx=lambda0; muU=0.5;
stdS=sqrt(n0*p*(1-p)); stdU=sqrt(1.0/12.0); #rS=numeric(nrep);
rU=numeric(nrep); rEx=numeric(nrep); rU1=numeric(nrep);
rU2=numeric(nrep); rU3=numeric(nrep);

MU3=numeric(1000); MU6=numeric(1000); MU20=numeric(1000);

# static  comparison
   for (irep in 1:1000)
    {  U3=rexp(3,rate=1/lambda0);
       U6=rexp(6,rate=1/lambda0)
       U20=rexp(20,rate=1/lambda0)
     MU3[irep]=sqrt(3)*(mean(U3)-muEx)/stdEx;
     MU6[irep]=sqrt(6)*(mean(U6)-muEx)/stdEx;
     MU20[irep]=sqrt(20)*(mean(U20)-muEx)/stdEx;
   }

 op=par(mfrow=c(3,1))
 hist(MU3,prob=TRUE)
 curve(dnorm(x),-5,5,col=2,add=T)
 hist(MU6,prob=TRUE)
 curve(dnorm(x),-5,5,col=2,add=T)
 hist(MU20,prob=TRUE)
 curve(dnorm(x),-5,5,col=2,add=T)
 par(op)




for (iobs in 10:nobs) {
  for ( irep in 1:nrep)
  {
```

```
#    S=rbinom(iobs,n0,p);
     Ex=rexp(iobs,rate=1/lambda0)
     U=runif(iobs);
    # rS[irep]=sqrt(iobs)*(mean(S)-muS)/sqrt(var(S));
    # rU[irep]=sqrt(iobs)*(mean(U)-muU)/sqrt(var(U));
     rEx[irep]=sqrt(iobs)*(mean(Ex)-muEx)/stdEx;
     rU[irep]=sqrt(iobs)*(mean(U)-muU)/stdU;
    }
 op=par(mfcol=c(2,2))
 plot(rEx);
 hist(rEx,ylim=c(0,1),prob=T,col="magenta");
 curve(dnorm(x),-5,5,add=TRUE);
 mtext(paste("# of observations=",iobs),side=3);
 plot(rU);
 hist(rU,ylim=c(0,1),xlim=c(-5,5),breaks=15,prob=T,col="magenta");
 curve(dnorm(x),-5,5,add=TRUE);
 mtext(paste("# of observations=",iobs),side=3);
 par(op)
}

for (iobs in 20:nobs) {
  for ( irep in 1:nrep)
  {
     U=runif(iobs);
     rU[irep]=sqrt(iobs)*(mean(U)-muU)/sqrt(var(U));
     rU1[irep]=mean(U);
     rU2[irep]=iobs*(mean(U)-muS);
     rU3[irep]=sqrt(iobs)*mean(U)/sqrt(var(U));
   }
 op=par(mfcol=c(2,2))
 hist(rU,ylim=c(0,1),xlim=c(-5,5),breaks=15,prob=T,col="magenta");
 mtext(paste("Mean of", "# of observations=",iobs),side=3);
 hist(rU1,ylim=c(0,1),xlim=c(-5,5),breaks=15,prob=T,col="magenta");
  mtext(paste("mean(X)", "# of observations=",iobs),side=3);
  hist(rU2,breaks=15,prob=T,col="magenta");
 mtext(paste("T * (mean(U)-mu)", "# of observations=",iobs),side=3);
```

```
  hist(rU3,breaks=15,prob=T,col="magenta");
 mtext(paste("T * (mean(U))", "# of observations=",iobs),side=3);
 par(op)
}
```

```
for ( i in 10:nobs)
 { x=rnorm(i);
   hist(x,ylim=c(0,1),xlim=c(-5,5),prob=T);
   curve(dnorm(x),-5,5,add=TRUE);
   mtext(paste("# of observations=",i),side=3);
   }
```