Jean-Pierre Cabannes
(www.cabannes.org)
2005-2013

Démarrer en R

Présentation

Installation

Principes et commandes de base

- Console
- Help
- Caractères
- Commentaires
- Commandes
- Affectation
- La commande « summary »
- Graphiques
- Objets disponibles/nettoyage
- Corrections
- Scripts
- Sorties
- Répertoires

Données, régression linéaire, un exemple

- La commande « read table »
- La commande « attach »
- La commande « lm »
- Compléments

Autres procédures, packages complémentaires, méthodes avancées

- Installation d'un package
- Chargement d'un package
- Package « stats »
- Package « tseries »
- Package « systemfit »
- Package « Rcmdr »
- Package « FactoMineR »

Textes et sites

Données pour Klein (fichier « data.txt »)

Présentation

R est un logiciel statistique, ou plus exactement un langage, introduit en 1993 par deux chercheurs de l'université d'Auckland (Nouvelle-Zélande) : Robert Gentleman et Ross Ihaka. C'est un logiciel libre, développé à présent par la communauté des participants au sein du R-project et proposé pour les trois systèmes d'exploitation : Unix/Linux, Windows et MacOS. C'est enfin un logiciel modulaire, de nombreux *packages* complémentaires offrent une grande variété de procédures mathématiques ou statistiques, incluant les méthodes économétriques complexes, le traitement des séries chronologiques, l'analyse des données, etc.

On présente quelques éléments de base permettant de comprendre le fonctionnement de R sous Windows et d'effectuer les calculs économétriques classiques sur ses données en s'inspirant des exemples proposés.

Installation

À partir du site : http://www.r-project.org/, du projet R, on va sur l'un des sites d'archives (*CRAN* pour Comprehensive R Archive Network) chercher le fichier d'installation de la dernière version de R; pour Windows, c'est actuellement le fichier « R-3.0.0-win.exe », d'environ 52 mégabytes.

On exécute le programme, ce qui installe R.

R s'installe par défaut dans la même langue que Windows, ce qui détermine les intitulés des menus et sous-menus, mais non toutes les commandes et réponses du système...

Les modules complémentaires désirés seront directement installés à partir de R via la liaison internet.

Principes et commandes de base

À la différence de logiciels tels Excel ou StatBox, plus conviviaux mais moins souples et moins puissants, R, est au départ un langage à ligne de commande, comme Stata. C'est cet usage qui va être décrit, bien que le « R-Commander » offre une interface graphique plus aisée à utiliser pour les traitements de base de statistique, d'économétrie et d'analyse des données.

Console : lorsque l'on lance R, une fenêtre, appelée la *console*, s'ouvre, elle affiche en bleu quelques informations de démarrage, puis en rouge l'invite ou

prompt « > » qui attend l'entrée d'une commande. Plus généralement, les commandes s'affichent en rouge et les réponses du système en bleu.

Aide : l'aide en ligne, menu « Aide », est très bien faite. La sous-commande « Aide HTML » ouvre par exemple l'aide dans le navigateur, tandis que « Fonctions R (text)… » ouvre une fenêtre de texte sur la fonction choisie – par exemple la fonction « lm », pour *linear model*.

On remarque que ces demandes génèrent à la console les instructions R équivalentes (ici « help.start() » et « help("lm") » respectivement) qu'on peut taper directement, mais il est évidemment plus simple de passer par les menus.

Caractères : attention R distingue minuscules et majuscules ; ainsi « help(lm) » appelle l'aide sur la commande lm tandis que « help(LM) » ou « HELP(lm) » produisent des messages d'erreur, de même on peut définir un objet « a » et un objet « A » qui n'ont aucune raison d'être identiques. Le plus simple est de n'utiliser que des minuscules pour désigner des objets, mais ce n'est pas un impératif…

Commentaires : une ligne commençant par « # » est un commentaire, il peut également suivre une instruction :

```
> # ceci est une ligne de commentaire
> … # un autre commentaire
```

Commandes : les commandes élémentaires sont soit des expressions, qui sont alors évaluées immédiatement, affichées et perdues :

```
> 3.14159          # on affiche les premières décimales de pi
3.14159
> (2*3)+(4*5)      # on calcule et affiche l'expression donnée
26
```

soit des *affectations*.

(Pour la commodité de la lecture, on présente en police *courrier*, de taille fixe, les réponses du système.)

Affectation : la syntaxe est « *objet* = *expression* », qui range l'évaluation de l'expression dans l'objet désigné qui est stocké et conservé :

```
> a = 3.14159      # on place la valeur 3,14159 dans l'objet a
> a                # on affiche l'objet a
[1] 3.14159
> b = 2.71828
> b
```

```
[1] 2.71828
> c = a+b+1        # on place l'évaluation de a+b+1 dans l'objet c
> c                # on affiche c
[1] 6.85987
> a                # on rappelle a qui est toujours là
[1] 3.14159
```

Mais R n'est pas simplement une calculette ! Une autre structure de données est par exemple le vecteur numérique, saisi par la commande « c » :

```
> # on définit la série y
> y = c(1, 2, 3, 5, 7, 11)
> y
[1] 1 2 3 5 7 11
```

On ne développe pas davantage les différents types de données ni leurs opérateurs ; on verra plus loin comment récupérer directement un ensemble de séries depuis un tableau rectangulaire usuel.

Remarque : on pourra rencontrer dans certains traités ou exemples « <- » en place de « = » pour indiquer l'affectation :

```
> y <- c(1, 2, 3, 5, 7, 11)          # par exemple
```

c'est intellectuellement préférable et reconnu par R, l'opération d'affection étant autre chose que le prédicat binaire d'égalité, mais néanmoins peu utilisé et nous gardons ici la notation traditionnelle.

La commande « summary » : la commande « summary » permet d'afficher les principales informations concernant un objet :

```
> summary(a)       # sans surprises…
   Min.   1st Qu.  Median    Mean  3rd Qu.    Max.
  3.142    3.142   3.142   3.142   3.142    3.142
> summary(y)       # plus intéressant
   Min.   1st Qu.  Median    Mean  3rd Qu.    Max.
  1.000    2.250   4.000   4.833   6.500   11.000
```

Graphiques : R offre une grande variété de graphiques, principalement via la commande « plot » et ses nombreuses variantes :

```
> plot(y)                  # le graphique de la série y
> plot(y, type = "l")      # le même en joignant les points
> lines(y)                 # une commande équivalente
> z = c(1, 1, 2, 3, 5, 8)  # une nouvelle série
> plot(z)
> plot(y,z)                # un graphique croisé
```

Ils apparaissent dans la fenêtre graphique, chacun y remplaçant le précédent. À moins qu'on ne se satisfasse de leur examen à l'écran, on peut sauvegarder les graphiques successivement obtenus, dans le format désiré, par la commande « Sauver sous » du menu « Fichier » de la fenêtre graphique, ou via le menu contextuel ouvert par un clic-droit, ou encore passer par les commandes du menu « Historique », après y avoir sélectionné l'option « Enregistrer ».

La commande « demo(graphics) » donne un large aperçu des riches possibilités de R dans le domaine graphique.

Objets disponibles/nettoyage : les commandes appropriées du menu « Misc » permettent d'afficher la liste des objets disponibles, ainsi éventuellement que de les supprimer en cas de tâtonnements et d'essais répétés. Il peut être également commode d'effacer la console par la commande convenable du menu « Edition », ou plus simplement via le menu contextuel ouvert par un clic-droit.

Corrections : les flèches de direction permettent de rappeler les commandes précédentes, et de s'y déplacer pour les modifier ou les corriger.

Scripts : un traitement de données ou la mise en œuvre d'une méthode économétrique demande généralement plus d'une commande, il est possible de soumettre globalement une suite de commandes, appelée *programme* ou *script*.

On peut ouvrir un script existant, coller un script copié ailleurs - tel un exemple d'un manuel en ligne - ou taper directement un programme dans la *fenêtre de script*, les commandes appropriées du menu « Edition » (ou du menu contextuel proposé par le clic-droit) permettent alors d'en faire exécuter tout ou partie pour voir les résultats dans la console.

Sorties : les résultats des commandes successives sortent par défaut sur la console ; mais il est possible de les envoyer en mode texte sur un fichier externe par la commande « sink("fichier") » :

```
> # on expédie les résultats dans le fichier result.txt
> sink("result.txt")
> objects()          # plus rien ne sort sur la console
> summary(y)
> sink()             # retour à la console
> summary(y)         # test…
…
```

Il ne faut pas oublier les guillemets encadrant le nom du fichier, qui est de fait une chaîne de caractères…

Répertoires : par défaut, les fichiers de scripts, de sorties, ou encore de données, sont supposés dans le répertoire « Mes documents » de l'utilisateur, mais il est possible voire conseillé de leur consacrer un répertoire propre qu'on spécifie par la commande « Changer le répertoire courant... » du menu « Fichier ».

Données, régression linéaire, un exemple

On présente maintenant un exemple simple de lecture de données et d'estimation d'une relation linéaire par les moindres carrés ordinaires.

La commande « read.table » : souvent, les données se présentent sous la forme de tableaux rectangulaires portant les séries en colonnes, tel celui des séries économiques utilisées dans le célèbre modèle de Klein (Lawrence R. Klein, 1920-…) :

```
year  cons   p    plag  wtot  wpri wpub  inv   klag   x     g     t    ylag
1921  41.9  12.4  12.7  28.2  25.5  2.7 -0.2  182.8  45.6   3.9   7.7  44.9
1922  45.0  16.9  12.4  32.2  29.3  2.9  1.9  182.6  50.1   3.2   3.9  45.6
...
1941  69.7  23.5  21.1  61.8  53.3  8.5  4.9  204.5  88.4  13.8  11.6  75.7
```

On suppose le tableau dans un fichier texte, il est lu et rangé dans un objet R de type liste de données par la commande « read.table("fichier", options) », ainsi si le fichier se nomme « data.txt » :

```
> # on copie et stocke les données de data.txt dans l'objet « klein »
> klein = read.table("data.txt", header=TRUE)
> klein       # pour voir…
   year cons    p plag wtot wpri wpub  inv  klag    x    g    t xlag
1  1921 41.9 12.4 12.7 28.2 25.5  2.7 -0.2 182.8 45.6  3.9  7.7 44.9
2  1922 45.0 16.9 12.4 32.2 29.3  2.9  1.9 182.6 50.1  3.2  3.9 45.6
   ...
21 1941 69.7 23.5 21.1 61.8 53.3  8.5  4.9 204.5 88.4 13.8 11.6 75.7
> summary(klein)
...
```

L'option « header=TRUE », « header=T », ou simplement « h=T », indique que la première ligne donne les noms des colonnes.

On peut encore indiquer que l'une des colonnes, désignée par son nom, servira d'identifiant :

```
> klein = read.table("data.txt", header=TRUE, row.names="year")
> klein       # vérification...
```

Cette commande est également adaptée au classique format texte universel : « csv », utilisé par les tableurs :

```
> klein = read.table("data.csv", h=T, sep=";", dec=",")
```

si le séparateur est le point-virgule et le séparateur décimal la virgule (csv français).

La commande permet également de lire un fichier directement sur le Net :

```
> klein = read.table("http://cabannes.org/data.csv", h=T, sep=";", dec=",")
```

La commande « write.table » permet l'opération inverse...

```
> write.table(klein, file="mesdata.txt")
```

La commande « attach » : cette commande permet de lier un objet R de type liste de données de manière à pouvoir appeler ses composants directement par leur nom; sa syntaxe est « attach(nom) » où "nom" est le nom de l'objet R considéré :

```
> summary(wtot)          # appel nominatif impossible
  Error in summary(wtot) : object "wtot" not found
> attach(klein)
> summary(wtot)          # wtot est à présent connu
   Min.   1st Qu.  Median    Mean  3rd Qu.    Max.
  28.20    37.00   40.70   41.48   45.30   61.80
```

En l'absence de cette liaison, les colonnes peuvent être invoquées par des commandes plus lourdes telles que celle-ci :

```
> summary(klein$wtot)
...
```

La commande « lm » : la commande de la régression linéaire est « lm » (pour *linear model*), sa syntaxe d'emploi est : « lm(y~x1+x2+x3+…) » où y est la variable à expliquer, suivi du caractère *tilde* : « ~ », et des variables explicatives séparées par le signe « + ». La constante, "intercept", est prise par défaut et n'a pas à être déclarée.

```
> # estimation de la fonction de consommation par les mco
> lm(cons~p+plag+wtot)
Call:
lm(formula = cons ~ p + plag + wtot)

Coefficients:
(Intercept)            p         plag         wtot
   16.23660      0.19293      0.08988      0.79622
```

Mais plutôt que cette sortie minimale et aussitôt perdue, il est préférable de demander les résultats complets via une affectation puis une commande « summary » :

```
> reg = lm(cons~p+plag+wtot)
> reg                    # reproduit l'affichage minimal des résultats
...
> summary(reg)           # affiche les résultats détaillés
Call:
lm(formula = cons ~ p + plag + wtot)

Residuals:
     Min       1Q   Median       3Q      Max
-2.17345 -0.43597 -0.03466  0.78508  1.61650

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 16.23660    1.30270  12.464 5.62e-10 ***
p            0.19293    0.09121   2.115   0.0495 *
plag         0.08988    0.09065   0.992   0.3353
wtot         0.79622    0.03994  19.933 3.16e-13 ***
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1

Residual standard error: 1.026 on 17 degrees of freedom
Multiple R-Squared: 0.981,      Adjusted R-squared: 0.9777
F-statistic: 292.7 on 3 and 17 DF,  p-value: 7.938e-15
```

Compléments : d'autres quantités sont encore calculées et stockées dans l'objet auquel sont affectés les résultats de la commande « lm » : les résidus, les valeurs ajustées, etc. :

```
> reg$residuals              # les résidus
> reg$fitted                 # la série ajustée
> plot(reg$residuals)        # on figure les résidus
> lines(reg$residuals)       # idem avec la ligne qui les joint
> plot(wtot, reg$residuals)  # un graphique croisé
```

On peut aussi demander la *stepwise regression* :

```
> step(reg)
Start:  AIC= 4.62
 cons ~ p + plag + wtot

       Df Sum of Sq    RSS    AIC
- plag  1      1.03  18.91   3.80
<none>               17.88   4.62
- p     1      4.71  22.59   7.53
- wtot  1    417.90 435.78  69.68

Step:  AIC= 3.8
 cons ~ p + wtot

       Df Sum of Sq    RSS    AIC
<none>               18.91   3.80
- p     1     13.37  32.29  13.03
```

```
- wtot  1    440.78 459.69  68.81

Call:
lm(formula = cons ~ p + wtot)

Coefficients:
(Intercept)            p         wtot
    16.4303       0.2506       0.8036
```

Autres procédures, packages complémentaires, méthodes avancées

R offre un très grand nombre de modules ou *packages* complémentaires, permettant de mettre en œuvre des méthodes avancées ou propres à un domaine particulier. Beaucoup sont disponibles sur les sites d'archives du projet et installables sur l'ordinateur directement depuis le logiciel ; une liste partielle en est donnée via l' « Aide HTML » du logiciel.

Installation d'un package : on suppose qu'on dispose d'une connexion internet. Ayant identifié un package complémentaire intéressant, par exemple « survival » pour le traitement des séries démographiques de survie, on utilise le menu « Packages », puis « Choisir le site miroir de CRAN... » pour choisir un site de téléchargement, et « Installer le(s) package(s)... » pour l'installer.

Chargement d'un package : pour pouvoir être utilisé, un package installé doit en outre être chargé par la commande « Charger le package... » qui rend également disponible l'aide associée. Contrairement à la précédente, cette opération doit être répétée à chaque nouveau lancement de R, et elle peut être ordonnée par ligne de commande ou dans un script par l'instruction « library » :

```
> library(survival)        # par exemple
```

Les packages de base, notamment « stats », sont installés et chargés par défaut avec le logiciel, il est donc inutile de tenter de les installer ou les charger à nouveau… Les commandes :

```
> library()
> search()
```

permettent par ailleurs de savoir quels sont les packages installés et les packages chargés.

Package « stats » : outre la régression linéaire multiple et la méthode stepwise déjà signalées, ce package contient de nombreuses procédures, incluant l'analyse en composantes principales (procédure princomp) et les méthodes de base de traitement des séries chronologiques (méthodes ar et arima par exemple, et procédures de mêmes noms).

Parmi de nombreuses options, la procédure « glm » (pour *generalized linear model*) permet l'estimation du modèle logit :

```
> # lecture de données d'abonnement
> abonnement = read.table("abon.dat", header=T)
> attach(abonnement)
> # on estime le modèle logit expliquant l'abonnement à une
> # certaine revue (abo=1) par l'âge et le sexe
> mlogit = glm(abo~age+sexe, family=binomial(link="logit"))
> summary(mlogit)
Call:
glm(formula = abo ~ age + sexe, family = binomial
(link = "logit"))

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.55961  -0.74655   0.06624   0.88512   1.84192

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  8.18443    3.09644   2.643  0.00821 **
age         -0.16491    0.06519  -2.530  0.01142 *
sexe        -2.42241    0.95590  -2.534  0.01127 *
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 55.452  on 39  degrees of freedom
Residual deviance: 38.981  on 37  degrees of freedom
AIC: 44.981

Number of Fisher Scoring iterations: 5
```

Package « tseries » : au-delà de celles de base, des méthodes plus avancées d'étude des séries chronologiques sont disponibles dans le package « tseries », tels les tests de stationnarité de Dickey-Fuller ou KPSS (procédures adf.test et kpss.test), l'estimation des modèles GARCH (procédure garch), etc.

Package « systemfit » : ce package permet l'estimation des modèles à équations simultanées, la procédure du même nom effectue notamment l'estimation globale par les doubles ou encore les triples moindres carrés. À titre d'exemple on donne un programme d'estimations du modèle à équations simultanées de Klein par diverses méthodes :

```
# lecture des données puis estimations du modèle de Klein
klein = read.table("data.txt", header=TRUE)
attach(klein)
```

```
# equations
conso =  cons ~ p + plag + wtot
wage = wpri ~ x + xlag + year
invest = inv ~ p + plag + klag
modklein = list(consommation = conso, salaires = wage, investissement =
invest)

# chargement du package « systemfit » supposé installé
library(systemfit)

# estimations par les mco
fitols = systemfit(modklein)
summary(fitols)

# estimations par la méthode "sur"
fitsur = systemfit(modklein, "SUR")
summary(fitsur)

# estimations par les doubles moindres carrés
inst = ~ klag + plag + xlag + wpub + g + tax + year
fitdmc = systemfit(modklein, "2SLS", inst=inst)
summary(fitdmc)

# estimations par les doubles moindres carrés
# avec instruments différents
inst1 = ~ klag + plag + xlag + tax + year
inst2 = ~ klag + plag + xlag + wpub + g + tax + year
inst3 = ~ klag + plag + xlag + wpub + g + tax
instlist = list(inst1, inst2, inst3)
fitdmc2 = systemfit(modklein, "2SLS", inst=instlist)
summary(fitdmc2)

# estimations par les triples moindres carrés
inst = ~ klag + plag + xlag + wpub + g + tax + year
fittmc = systemfit(modklein, "3SLS", inst=inst)
summary(fittmc)
```

Package « Rcmdr » : ce package offre à R une interface graphique, le R-Commander, ayant l'ambition de rendre plus conviviales les commandes de manipulation de données et la mise en œuvre des méthodes statistiques de base, incluant la régression linéaire et l'estimation des modèles dérivés.

Pour un travail limité à ces méthodes, l'utilisateur n'a donc pas à apprendre la syntaxe des commandes sous-jacentes et le logiciel peut paraître aussi simple à utiliser que Excel ou Eviews.

Pour la mise en œuvre de méthodes plus complexes, telles celles liées aux équations simultanées ou aux séries temporelles, la programmation par ligne de commande décrite dans le présent document reste pour l'instant nécessaire.

Package « FactoMineR » : ce package propose les méthodes factorielles classiques d'analyse des données, telles l'ACP, l'AFC et AFCM, etc. « à la française ».

En suivant les indications de ses auteurs (voir sur le site indiqué *infra*), on peut installer une version du R-Commander incluant l'appel de ces méthodes.

Textes et sites

Site du projet R
www.r-project.org

Initiation générale à R
www.ceremade.dauphine.fr/~xian/Noise/R.pdf

Initiation à l'économétrie en R
http://cran.r-project.org/doc/contrib/Farnsworth-EconometricsInR.pdf

Site de FactoMineR
http://factominer.free.fr/index_fr.html

Des données et des scripts sur mon site
http://www.cabannes.org

Données pour Klein (fichier « data.txt »)

year	cons	p	plag	wtot	wpri	wpub	inv	klag	x	g	t	xlag
1921	41.9	12.4	12.7	28.2	25.5	2.7	-0.2	182.8	45.6	3.9	7.7	44.9
1922	45.0	16.9	12.4	32.2	29.3	2.9	1.9	182.6	50.1	3.2	3.9	45.6
1923	49.2	18.4	16.9	37.0	34.1	2.9	5.2	184.5	57.2	2.8	4.7	50.1
1924	50.6	19.4	18.4	37.0	33.9	3.1	3.0	189.7	57.1	3.5	3.8	57.2
1925	52.6	20.1	19.4	38.6	35.4	3.2	5.1	192.7	61.0	3.3	5.5	57.1
1926	55.1	19.6	20.1	40.7	37.4	3.3	5.6	197.8	64.0	3.3	7.0	61.0
1927	56.2	19.8	19.6	41.5	37.9	3.6	4.2	203.4	64.4	4.0	6.7	64.0
1928	57.3	21.1	19.8	42.9	39.2	3.7	3.0	207.6	64.5	4.2	4.2	64.4
1929	57.8	21.7	21.1	45.3	41.3	4.0	5.1	210.6	67.0	4.1	4.0	64.5
1930	55.0	15.6	21.7	42.1	37.9	4.2	1.0	215.7	61.2	5.2	7.7	67.0
1931	50.9	11.4	15.6	39.3	34.5	4.8	-3.4	216.7	53.4	5.9	7.5	61.2
1932	45.6	7.0	11.4	34.3	29.0	5.3	-6.2	213.3	44.3	4.9	8.3	53.4
1933	46.5	11.2	7.0	34.1	28.5	5.6	-5.1	207.1	45.1	3.7	5.4	44.3
1934	48.7	12.3	11.2	36.6	30.6	6.0	-3.0	202.0	49.7	4.0	6.8	45.1
1935	51.3	14.0	12.3	39.3	33.2	6.1	-1.3	199.8	54.4	4.4	7.2	49.7
1936	57.7	17.6	14.0	44.2	36.8	7.4	2.1	197.7	62.7	2.9	8.3	54.4
1937	58.7	17.3	17.6	47.7	41.0	7.7	2.0	199.8	65.0	4.3	6.7	62.7
1938	57.5	15.3	17.3	45.9	38.2	7.8	-1.9	201.8	60.9	5.3	7.4	65.0
1939	61.6	19.0	15.3	49.4	41.6	7.8	1.3	199.9	69.5	6.6	8.9	60.9
1940	65.0	21.1	19.0	53.0	45.0	8.0	3.3	201.2	75.7	7.4	9.6	69.5
1941	69.7	23.5	21.1	61.8	53.3	8.5	4.9	204.5	88.4	13.8	11.6	75.7

-----==ooΩoo==-----

(12 avril 2013)