Chapitre 1

R, généralités et installation

1.1. Histoire

R est apparu en 1975 comme boîte à outils de traitement statistique (S développé par Bell Labs). Cette « *toolbox* » a été commercialisée sous le nom de S-plus. En guise de contestation, Robert Gentleman a décidé de la rendre à nouveau publique en redéveloppant une version open-source en 1996 (voir R Core Team, 2013).

Aujourd'hui R c'est (voir German *et al*., 2013) :

– un noyau de trente développeurs (*R core team*) ;

– une communauté d'environ 3 000 développeurs actifs ;

– 100 packages en 2002. Au 13 novembre 2013, 5 012 packages, soit environ 50 000 scripts.

C'est aussi (voir Smith, 2014) :

– des dizaines de sociétés (SAS, SPSS, Oracle, Facebook, Google, ThomasCook, Bing, etc.) qui rendent leurs produits interopérables avec R pour leurs utilisateurs ;

– environ 1,2 million d'utilisateurs dans le monde (utilisateurs directs) ;

– 15 000 visites par jour sur le site http://cran.r-project.org/.

Il existe aussi des blogs, des forums d'aide en ligne et une revue scientifique *R journal* (ISSN : 2073-4859, Impact factor 0.8 en 2012).

Journal of Statistical Software (ISSN : 1548-7660, Impact factor 4.910 en 2012) publie aussi des articles liés aux packages de R.

L'histoire sur le plan des analyses de textes est très récente sous R. Jean-Pierre Muller (université de Genève) a développé un module appelé « ttda » (*tools for textual data analysis*) en mars 2004. A ce jour, il n'est plus maintenu pour la version de R3.0.2.

Depuis 2008, le package « tm » fait référence en exploration de corpus. D'autres packages viennent se rajouter à la liste avec certaines spécificités comme le package « twitteR » qui récupère des données du réseau social Twitter pour pouvoir les analyser.

1.2. Installation de R

Le site où se trouve le plus d'informations sur R (téléchargement, manuels) est http://cran.r-project.org/. Les sites d'aide en ligne sont http://tolstoy.newcastle.edu.au/R/ et http://stackoverflow.com (figure 1.1).

Quand on va sur le site CRAN.R (cran.r-project.org), on peut télécharger la version correspondant à son système d'exploitation : Windows, Linux ou Mac.

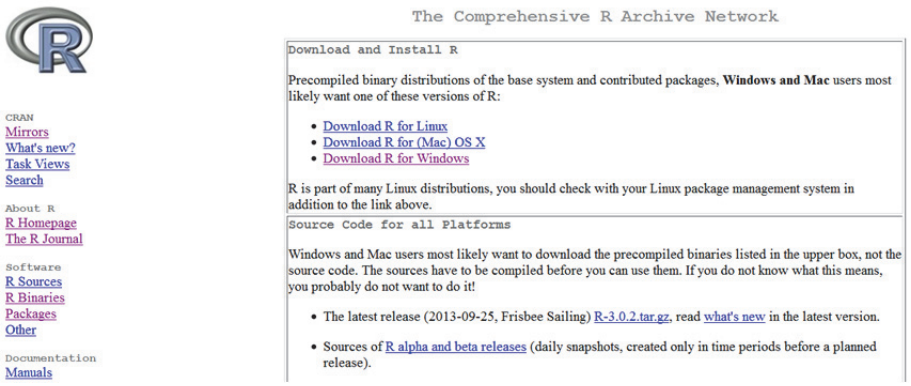

Figure 1.1. *Interface du site CRAN*

On télécharge. Il faut veiller régulièrement à mettre à jour sa version (figure 1.2).

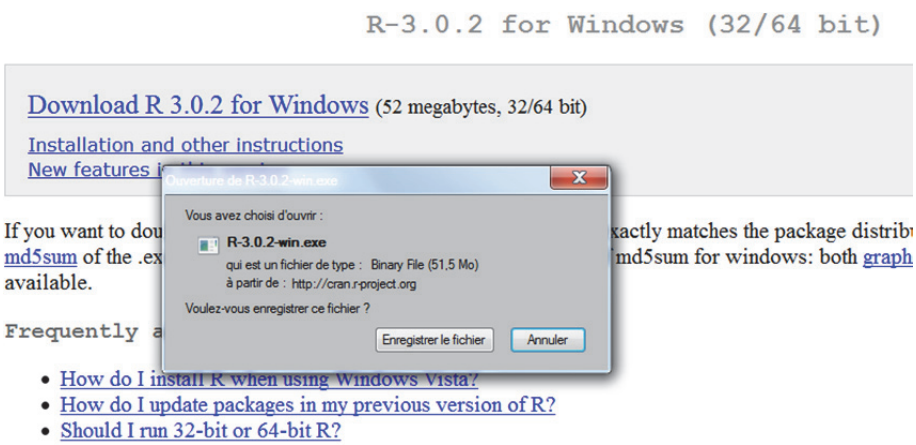

Figure 1.2. *Téléchargement de R*

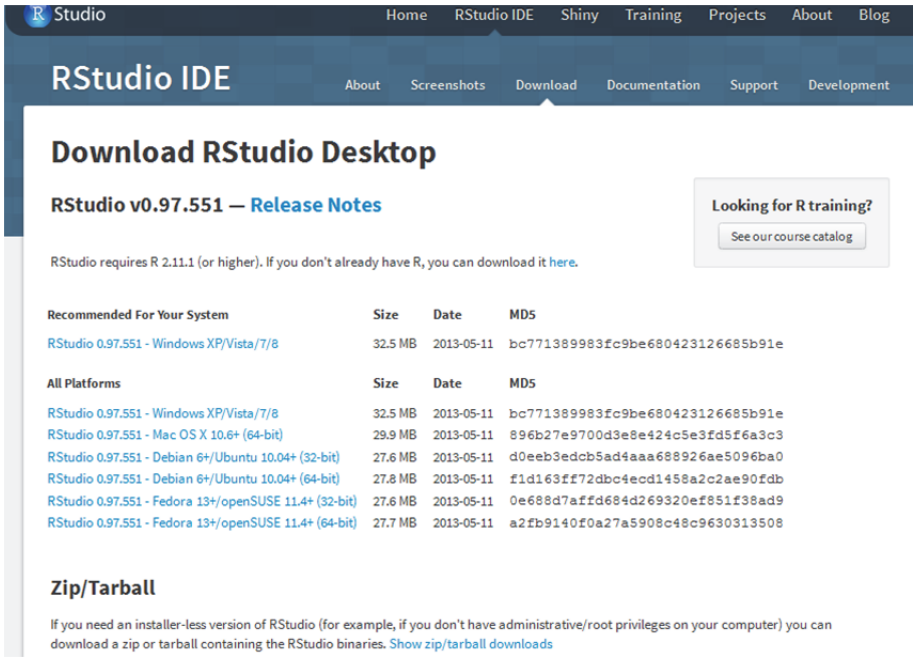

Figure 1.3. *Téléchargement de RStudio*

Une interface graphique de développement conviviale et spécifique à R (aussi appelée IDE pour *Integrated Development Interface*) a vu le jour : RStudio. Cette interface permet de visualiser le script en cours d'édition, les résultats graphiques et l'exécution en mode console, d'un seul coup d'œil. En jouant aussi sur une activité en mode projet. Cette interface s'installe indépendamment de R (dans un répertoire à part) à partir du site web : www.rstudio.com/ (figure 1.3).

Sur la figure 1.4, on peut voir l'interface une fois téléchargée et activée.

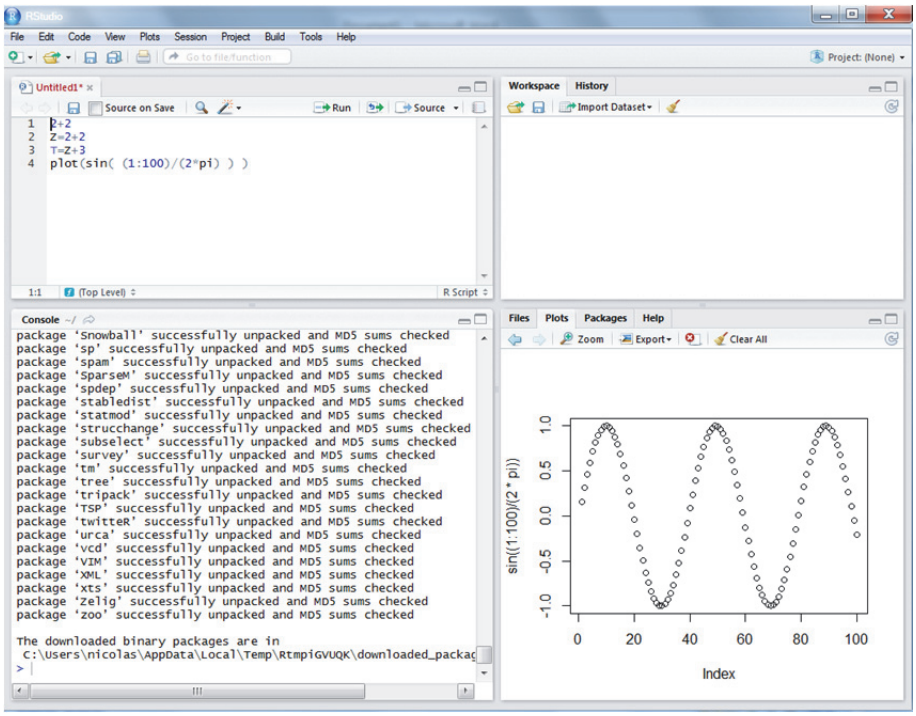

Figure 1.4. *Interface graphique de RStudio*

1.3. Mise en route de R

Lorsque l'on exécute R, on voit apparaître la fenêtre générale avec un menu et une console. On ouvre un éditeur de script en allant dans le menu fichier > nouveau script. Une nouvelle fenêtre apparaît. Dans cette fenêtre, on va écrire les commandes qui nous seront utiles pour faire un ou plusieurs traitements (figure 1.5).

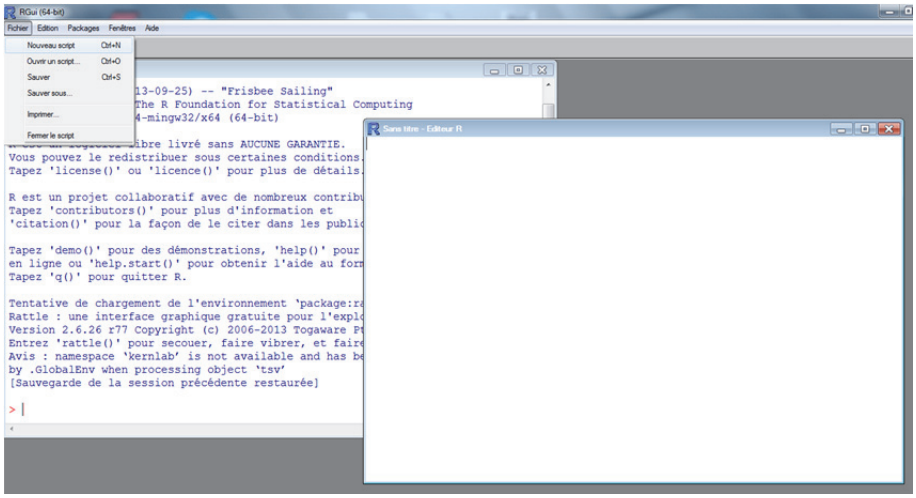

Figure 1.5. *Interface utilisateur en ligne de commande*

On tape ce code qui doit nous afficher une sinusoïde :

```
> Z=2+2;
> Z;
> A=Z*10;
> A;
> S = sin( (1:A)/(2*pi) )
> S;
> plot( S )
```

La ligne 1 signifie que l'on déclare une variable Z à laquelle on affecte le calcul de 2+2.

La ligne 2 signifie que l'on affiche le contenu de Z.

La ligne 3 signifie que l'on multiplie le contenu de Z par 10 et que l'on affecte le résultat à la variable A.

La ligne 4 signifie que l'on affiche le contenu de A.

La ligne 5 signifie que l'on affecte à S le calcul de la fonction sinus à la série de nombres dont le premier est 1 divisé par 2xp le deuxième 2 divisé par 2p, le dernier 40 divisé par 2p.

La ligne 6 affiche la série S.

La ligne 7 affiche le graphe dont l'abscisse est de 1/2p à max(A)/2p, l'ordonnée est la valeur de S correspondante.

Pour exécuter l'ensemble des lignes, on sélectionne tout (en surbrillance) et on clique sur le bouton exécuter. Le résultat s'affiche (graphe et console) après exécution. Si l'on clique sur le bouton sans sélection, seule la dernière ligne du script s'exécute (voir figure 1.6).

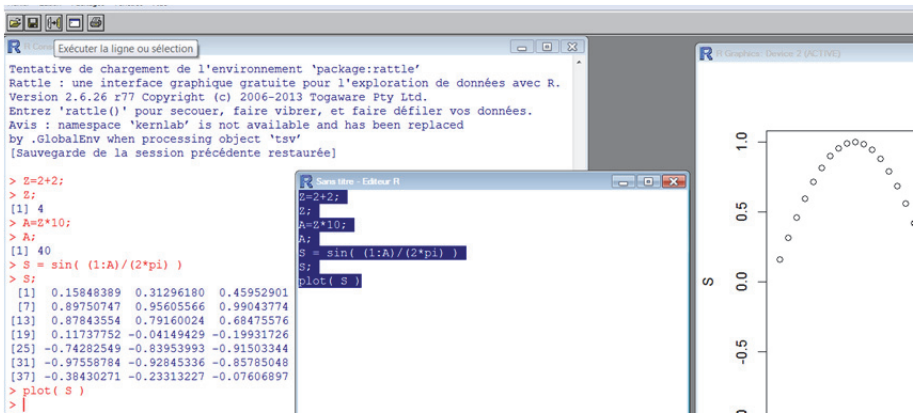

Figure 1.6. *Exécution d'un script*

1.4. Evaluation du temps de calcul

La fonction *proc.time()* permet de récupérer les données de l'horloge. Par différence entre deux données, on obtient la durée d'un intervalle de temps et donc du temps de calcul d'un processus. Dans R les boucles sont plutôt coûteuses. Il est en général recommandé d'exécuter les commandes R des différents packages qui sont optimisées en temps d'exécution. Dans le script suivant, on teste la durée d'exécution d'une boucle qui incrémente de 1 la valeur aléatoire d'un tableau et on affiche la durée d'exécution. La durée écoulée est exactement ce que prend le système pour terminer montre en main :

```
> # Stop the clock
> h <- rep(NA, 1000000)
>
> # Start the clock!
> ptm <- proc.time()
>
> # Loop through the vector, adding one
> for (i in 1:1000){
+ j = sample(1:1000000, 1)
```

```
+ h[ j ] <- h[j] + 1
+ }
>
> # Stop the clock
> proc.time() - ptm
utilisateur   système   écoulé
       2.85      1.62     6.21
>
```

1.5. Installation d'un package

Le tableau 1.1 montre les différentes packages pour réaliser une analyse de corpus. Il existe plus de vingt packages spécialisés. Il est aussi possible d'exploiter l'export de ces packages pour ensuite rebondir sur des packages de traitement plus transversaux (clustering, classification, visualisation, etc.). Cette opportunité offre à ce moment-là plus de 5 500 packages.

Dans l'installation de base, les packages d'analyse de corpus ne sont pas installées. Il existe plus de vingt packages concernant le traitement de documents en langage naturel (voir tableau 1.1).

Nom	Thème	Maintenance	Contact
DeducerText	interface graphique qui utilise tm	Ian Fellows	<ian@fellstat.com>
koRpus	indices de lisibilité d'un corpus	m.eik michalke	<meik.michalke@hhu.de>
LanguageR	modèles de langage	R. H. Baayen	<harald.baayen@uni-tuebingen.de>
maptpx	sélection de modèle pour topics latents dans les textes	Matt Taddy	<taddy@chicagobooth.edu>
maxent	classification de textes	Timothy P. Jurka	<tpjurka@ucdavis.edu>
openNLP	extraction d'entités nommées	Kurt Hornik	<Kurt.Hornik@R-project.org>
qdap	analyse quantitative du discours	Tyler Rinker	<tyler.rinker@gmail.com>
Rcmdr Plugin.temis	plug-in à R commander ; extraction de co-occurrents ; classifications de termes	Milan Bouchet-Valat	<nalimilan@club.fr>
RTextTools	apprentissage automatique pour la classification automatique	Timothy P. Jurka	<tpjurka@ucdavis.edu>

Nom	Thème	Maintenance	Contact
smdc	similarité entre documents	Masaaki TAKADA	<tkdmah@gmail.com>
stm	modèle structurel des topics	Brandon Stewart	<bstewart@fas.harvard.edu>
stringdist	similarité entre chaînes	Mark van der Loo	<mark.vanderloo@gmail.com>
textcat	catégorisation de documents par n-grammes	Kurt Hornik	<Kurt.Hornik@R-project.org>
textir	analyse de sentiment par régression logistique	Matt Taddy	<taddy@chicagobooth.edu>
textometry	scores de spécificité d'association	Matthieu Decorde	<matthieu.decorde@ens-lyon.fr>
tm	matrice termes documents ; co-occurrence de mots	Ingo Feinerer	<ingo.feinerer@tuwien.ac.at>
tm.plugin.factiva	parsing du format factiva	Milan Bouchet-Valat	<nalimilan@club.fr>
topicmodels	*Latent Dirichlet Allocation* (LDA) *models* et *Correlated Topics Models* (CTM)	Bettina Grün	<Bettina.Gruen@jku.at>
lsa	analyse sémantique latente	Fridolin Wild	<f.wild@open.ac.uk>
twitteR	téléchargement de tweets	Jeff Gentry	<geoffjentry@gmail.com>
wordnet	ontologie lexicale	Kurt Hornik	<Kurt.Hornik@R-project.org>
x-ent	extraction d'entités nommées generaliste	Nicolas Turenne et Tien Phan	<nicolas.inra@yahoo.fr> <phantien84@gmail.com>
zipfR	modèle de distribution de grands nombre d'événements rares ; interpolation	Stefan Evert et Marco Baroni	<stefan.evert@uos.de> <marco.baroni@unitn.it>

Tableau 1.1. *Liste des packages R qui traitent des données textuelles (DeducerText, koRpus, LanguageR, maptpx, maxent, openNLP, qdap, RcmdrPlugin.temis, RTextTools, smdc, stm, stringdist, textcat, textir, textometry, tm, tm.plugin.factiva, topicmodels, lsa, twitteR, wordnet, x-ent, zipfR).*

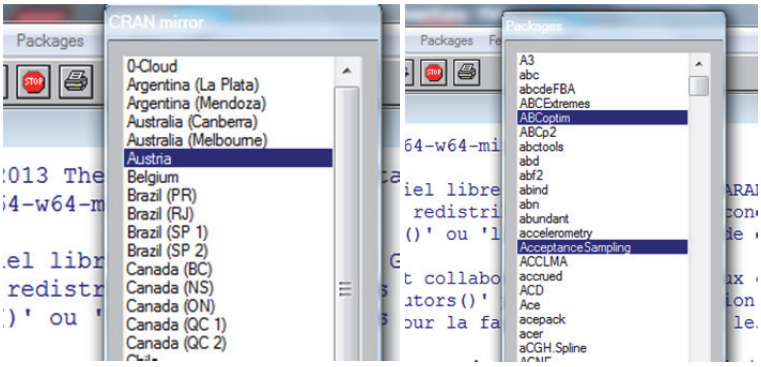

Figure 1.7. *Installation d'un package*

Pour installer une ou plusieurs packages, aller dans le menu supérieur aller dans « Packages », sélectionner d'abord « choisir le site miroir de CRAN » valider, ensuite revenir dans « Packages » et sélectionner « installer les packages » (figure 1.7). A ce niveau il faut sélectionner une ou plusieurs packages à installer avec la souris et valider.

Attention ! Avant d'installer tous les packages. Sur un Intel(R) Core(TM)2 Duo CPU P8600, horloge 2.4 GHz, le temps d'installation est de quatre heures, et occupe 2,7 Go (96 000 fichiers) d'espace disque.

Une alternative aux fenêtres est d'utiliser une installation en ligne de commande, il faut connaître le nom exact du package avec ou sans majuscules et utiliser la commande « install.package » :

```
> install.packages("stringdist")
--- SVP sélectionner un miroir CRAN pour cette session ---
essai de l'URL 'http://cran.at.r-
project.org/bin/windows/contrib/3.1/stringdist_0.9.0.zip'
Content type 'application/zip' length 148064 bytes (144 Kb)
URL ouverte
downloaded 144 Kb

le package 'stringdist' a été décompressé et les sommes MD5 ont
été vérifiées avec succès

Les packages binaires téléchargés sont dans

C:\Users\nicolas\AppData\Local\Temp\RtmpInOnZ2\downloaded_packages
>
```

On peut aussi installer un package qui n'est pas référencé dans la base CRAN en récupérant le fichier archive zip d'installation. A ce moment là l'installation se fait en utilisant la commande R CMD CHECK dans une fenêtre-terminal (figure 1.8) (ou R CMD INSTALL).

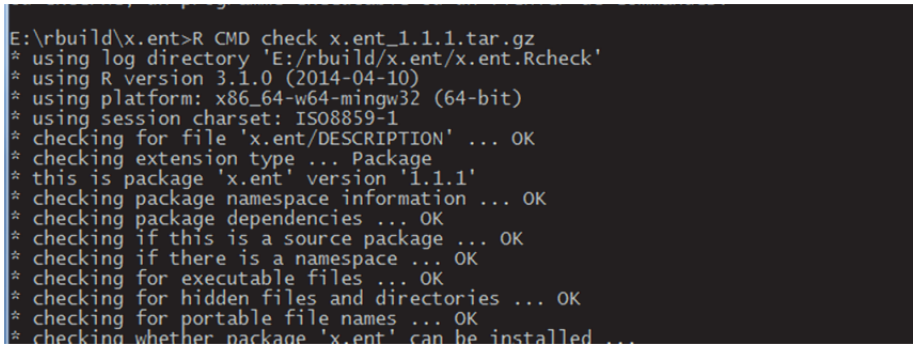

Figure 1.8. *Installation manuelle de l'archive d'un package*

La commande décompresse l'archive dans un répertoire Nom_de_package.x.ent.Rcheck, il faut ensuite copier le sous-répertoire « Nom_de_package.Rcheck/Nom_de_package » dans le répertoire des packages de R « \R\library\ ».