

Chapitre 1

Présentation du langage R

Objectifs du chapitre

1. Comprendre ce qu'est un langage de programmation interprété.
2. Connaître la provenance du langage R et les principes ayant guidé son développement.
3. Mettre en place sur son poste de travail un environnement de développement en R.
4. Démarrer une session R et exécuter des commandes simples.
5. Utiliser des fichiers de script R de manière interactive.
6. Créer, modifier et sauvegarder ses propres fichiers de script R.

1.1 Bref historique

À l'origine fut le S, un langage pour « programmer avec des données » développé chez Bell Laboratories à partir du milieu des années 1970 par une équipe de chercheurs menée par John M. Chambers. Au fil du temps, le S a connu quatre principales versions communément identifiées par la couleur du livre dans lequel elles étaient présentées : version « originale » (Brown Book ; Becker et Chambers, 1984), version 2 (Blue Book ; Becker et collab., 1988), version 3 (White Book ; Chambers et Hastie, 1992) et version 4 (Green Book ; Chambers, 1998) ; voir aussi Chambers (2000) et Becker (1994) pour plus de détails.

Dès la fin des années 1980 et pendant près de vingt ans, le S a principalement été popularisé par une mise en œuvre commerciale nommée S-PLUS. En 2008, Lucent Technologies a vendu le langage S à Insightful Corporation, ce qui a effectivement stoppé le développement du langage par ses auteurs originaux. Aujourd'hui, le S est commercialisé de manière relativement

1.2 Description sommaire de R

confidentielle sous le nom Spotfire S+ par TIBCO Software.

Ce qui a fortement contribué à la perte d'influence de S-PLUS, c'est une nouvelle mise en œuvre du langage développée au milieu des années 1990. Inspirés à la fois par le S et par Scheme (un dérivé du Lisp), Ross Ihaka et Robert Gentleman proposent un langage pour l'analyse de données et les graphiques qu'ils nomment R (Ihaka et Gentleman, 1996). À la suggestion de Martin Maechler de l'ETH de Zurich, les auteurs décident d'intégrer leur nouveau langage au projet GNU(1), faisant de R un logiciel libre.

Ainsi disponible gratuitement et ouvert aux contributions de tous, R gagne rapidement en popularité là même où S-PLUS avait acquis ses lettres de noblesse, soit dans les milieux académiques. De simple dérivé « not unlike S », R devient un concurrent sérieux à S-PLUS, puis le surpasse lorsque les efforts de développement se rangent massivement derrière le projet libre. D'ailleurs John Chambers place aujourd'hui ses efforts de réflexion et de développement dans le projet R (Chambers, 2008).

1.2 Description sommaire de R

R est un environnement intégré de manipulation de données, de calcul et de préparation de graphiques. Toutefois, ce n'est pas seulement un « autre » environnement statistique (comme SPSS ou SAS, par exemple), mais aussi un langage de programmation complet et autonome.

Tel que mentionné précédemment, le R est un langage principalement inspiré du S et de Scheme (Abelson et collab., 1996). Le S était à son tour inspiré de plusieurs langages, dont l'APL (autrefois un langage très prisé par les actuaire) et le Lisp. Comme tous ces langages, le R est interprété, c'est-à-dire qu'il requiert un autre programme — l'interprète — pour que ses commandes soient exécutées. Par opposition, les programmes de langages compilés, comme le C ou le C++, sont d'abord convertis en code machine par le compilateur puis directement exécutés par l'ordinateur.

Cela signifie donc que lorsque l'on programme en R, il n'est pas possible de plaider l'attente de la fin de la phase de compilation pour perdre son temps au travail.

Le programme que l'on lance lorsque l'on exécute R est en fait l'interprète. Celui-ci attend qu'on lui soumette des commandes dans le langage R, commandes qu'il exécutera immédiatement, une à une et en séquence. Par analogie, Excel est certes un logiciel de manipulation

1.3 Interfaces

de données, de mise en forme et de préparation de graphiques, mais c'est aussi au sens large un langage de programmation interprété. On utilise le langage de programmation lorsque l'on entre des commandes dans une cellule d'une feuille de calcul. L'interprète exécute les commandes et affiche les résultats dans la cellule.

Le R est un langage particulièrement puissant pour les applications mathématiques et statistiques (et donc actuarielles) puisque précisément développé dans ce but. Parmi ses caractéristiques particulièrement intéressantes, on note :

- langage basé sur la notion de vecteur, ce qui simplifie les calculs mathématiques et réduit considérablement le recours aux structures itératives (boucles for, while, etc.);
- pas de typage ni de déclaration obligatoire des variables; programmes courts, en général quelques lignes de code seulement;
- programmes courts, en général quelques lignes de code seulement;
- temps de développement très court.

1.3 Interfaces

R est d'abord et avant tout une application n'offrant qu'une invite de commande du type de celle présentée à la ???. En soi, cela n'est pas si différent d'un tableur tel que Excel : la zone d'entrée de texte dans une cellule n'est rien d'autre qu'une invite de commande, par ailleurs aux capacités d'édition plutôt réduites.



```
RGui (64-bit) - [R Console]
Fichier Edition Voir Misc Packages Fenêtres Aide

R version 4.0.3 (2020-10-10) -- "Bunny-Wunnies Freak Out"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R est un logiciel libre livré sans AUCUNE GARANTIE.
Vous pouvez le redistribuer sous certaines conditions.
Tapez 'license()' ou 'licence()' pour plus de détails.

R est un projet collaboratif avec de nombreux contributeurs.
Tapez 'contributors()' pour plus d'information et
'citation()' pour la façon de le citer dans les publications.

Tapez 'demo()' pour des démonstrations, 'help()' pour l'aide
en ligne ou 'help.start()' pour obtenir l'aide au format HTML.
Tapez 'q()' pour quitter R.

[Sauvegarde de la session précédente restaurée]
> |
```

FIGURE 1.1 – Fenêtre de la console sous OS X au démarrage de R

1.4 Stratégies de travail

- Sous Windows, une interface graphique plutôt rudimentaire est disponible. Elle facilite certaines opérations telles que l'installation de packages externes, mais elle offre autrement peu de fonctionnalités additionnelles pour l'édition de code R.
- L'interface graphique de R sous Mac OS X est la plus élaborée. Outre la console présentée à la figure ??, l'application R.app comporte de nombreuses fonctionnalités, dont un éditeur de code assez complet.
- Sous Unix et Linux, R n'est accessible que depuis la ligne de commande du système d'exploitation (terminal). Aucune interface graphique n'est offerte avec la distribution de base de R.

1.4 Stratégies de travail

Dans la mesure où R se présente essentiellement sous forme d'une invite de commande, il existe deux grandes stratégies de travail avec cet environnement statistique.

1. On entre des expressions à la ligne de commande pour les évaluer immédiatement :
On peut également créer des objets contenant le résultat d'un calcul. Ces objets

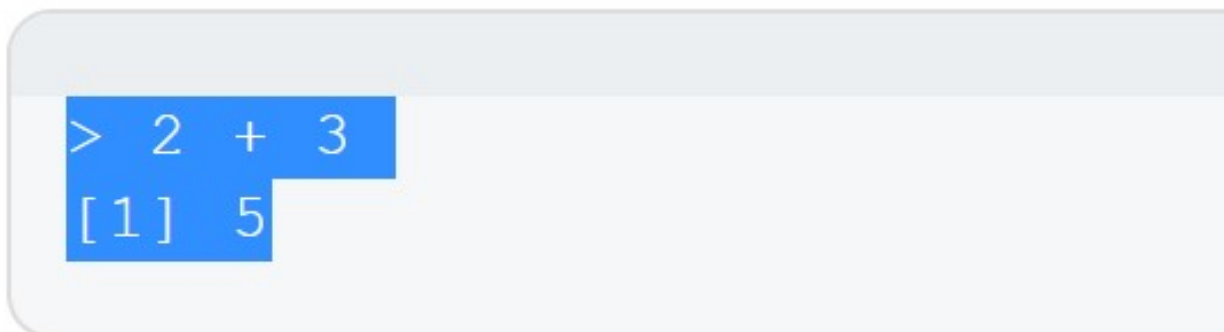


FIGURE 1.2 – Fenêtre de la console sous OS X au démarrage de R

sont stockés en mémoire dans l'espace de travail de R : Lorsque la session de travail est terminée, on sauvegarde une image de l'espace de travail sur le disque dur de l'ordinateur afin de pouvoir conserver les objets pour une future séance de travail : Par défaut, l'image est sauvegardée dans un fichier nommé `.RData` dans le dossier de travail actif (voir la section 1.7) et cette image est automatiquement chargée en mémoire au prochain lancement de R, tel qu'indiqué à la fin du message d'accueil : Cette approche, dite de « code virtuel et objets réels » a un gros inconvénient : le

1.4 Stratégies de travail

```
> x <- exp(2)
> x
[1] 7.389056
```

FIGURE 1.3 – Fenêtre de la console sous OS X au démarrage de R

```
> save.image()
```

FIGURE 1.4 – Fenêtre de la console sous OS X au démarrage de R

```
[Sauvegarde de la session précédente restaurée]
```

FIGURE 1.5 – Fenêtre de la console sous OS X au démarrage de R

code utilisé pour créer les objets n'est pas sauvegardé entre les sessions de travail. Or, celui-ci est souvent bien plus compliqué que l'exemple ci-dessus. De plus, sans accès au code qui a servi à créer l'objet `x`, comment savoir ce que la valeur 7.389056 représente au juste ?

2. L'approche dite de « code réel et objets virtuels » considère que ce qu'il importe de conserver d'une session de travail à l'autre n'est pas tant les objets que le code qui a servi à les créer. Ainsi, on sauvegardera dans ce que l'on nommera des fichiers

1.4 Stratégies de travail

de script nos expressions R et le code de nos fonctions personnelles. Par convention, on donne aux fichiers de script un nom se terminant avec l'extension `.R`. Avec cette approche, les objets sont créés au besoin en exécutant le code des fichiers de script. Comment ? Simplement en copiant le code du fichier de script et en le collant dans l'invite de commande de R.