Fiche TD avec le logiciel R : tdr781

Écrire un rapport avec `Sweave()` Tutoriel des deuxièmes rencontres R, 26 juin 2013

A.B. Dufour, S. Mousset
inspirés de J.R. Lobry

Ce tutoriel est une introduction à l'utilisation de `Sweave` pour générer des rapports intégrant des analyses statistiques et des graphiques produits par R. Elle utilise le logiciel `RStudio`, une interface graphique et multiplateforme pour le logiciel R.

Table des matières

1 Introduction

`Sweave()` est une commande standard de R qui permet d'intégrer du code R directement dans le code source d'un document (par exemple un rapport scientifique) écrit en LaTeX.
Le principe est le suivant :

- ⋆ Du code R est incorporé dans le code source d'un document écrit en LaTeX. Deux syntaxes sont proposées (syntaxe "noweb" et syntaxe LaTeX) qui diffèrent par le type de balises utilisées pour permettre d'identifier le code R. Le nom du fichier du document source contenant le code R est traditionnellement nommé avec l'une des extensions suivantes : `.Rnw`, `.rnw`, `.Snw`, `.snw` (documents écrits avec une syntaxe "noweb") ou `.Rtex`, `.rtex`, `.Stex`, `.stex` (documents écrits avec une syntaxe LaTeX).
- ⋆ À l'aide de la commande `Sweave()` exécutée sous R, le code R du document est exécuté et les résultats (texte, figures...) sont intégrés dans un fichier standard LaTeX (extension `.tex`).
- ⋆ La composition de texte (*typesetting*) à partir du fichier généré par `Sweave()` est ensuite confiée à LaTeX, par exemple avec la commande `pdflatex`.

Bien que l'ensemble de ces étapes puisse être réalisé en ligne de commande R, il peut être plus simple d'utiliser une interface graphique pour R qui permettra de réaliser l'ensemble de ces étapes de façon transparente pour l'utilisateur. Nous avons choisi pour cela d'utiliser `RStudio`().

2 Les logiciels requis

2.1 Installation

Les logiciels à installer dépendent des systèmes d'exploitation

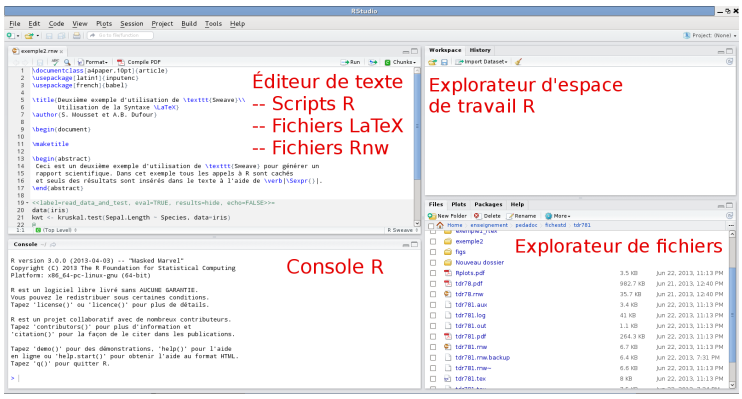
- ⋆ LaTeX : sous linux, nous utiliserons la distribution TeXLive (à utiliser avec le gestionnaire de paquet de votre distribution), sous Windows nous utiliserons MiKTeX (`http://www.miktex.org/`), sous Mac OS X on utilisera la distribution MacTeX (`http://tug.org/mactex/`).
- ⋆ Bien entendu, R `3.0.0` ou une version plus récente est disponible sur `http://www.r-project.org`.
- ⋆ Pour simplifier le travail, une interface graphique pour le logiciel R peut être très utile. Nous utiliserons le logiciel `RStudio`, un environnement graphique et multiplateforme pour R (`http://www.rstudio.com/`)

2.2 Fonctionnement de `RStudio`

`RStudio` est un environnement graphique permettant d'exécuter du code R et qui possède un éditeur de texte adapté à l'édition de scripts R, de fichiers LaTeX et de fichiers noweb combinant des commandes R dans un document LaTeX.
À l'ouverture de `RStudio`, la fenêtre est séparée en 4 parties (voir la figure 1) :

- ⋆ Un éditeur de texte où vous pourrez éditer vos sources.

FIGURE 1 – L'environnement graphique `RStudio`



- ⋆ Une console R où vous pourrez exécuter des commandes.
- ⋆ Un explorateur d'espace de travail R où sont répertoriés les objets de votre espace de travail.
- ⋆ Un explorateur de fichiers.

2.3 Premiers pas avec `RStudio`

Avec `RStudio` ouvrez le premier fichier d'exemple nommé `exemple1.rnw` (ou copiez le code de la figure 2 dans un nouveau script que vous sauverez sous le nom `exemple1.rnw`). Attention, ce fichier est prévu pour utiliser l'encodage latin-1 (ISO-8859-1). Si nécessaire ré-ouvrez le avec le bon encodage (cliquez sur "File", puis "Reopen with Encoding").
Essayez de compiler ce fichier en cliquant sur "Compile PDF". Normalement, vous devriez obtenir un fichier d'une page contenant deux sections de code R.

3 Inclure du code R dans un document LaTeX

On utilise des balises dans le document source pour permettre à R de distinguer le code à exécuter du texte à ignorer. Deux types de syntaxe sont utilisables pour ces balises (même au sein du même document, ce qui n'est sans doute pas à conseiller).

3.1 La syntaxe noweb

La syntaxe noweb est une syntaxe classique utilisée permettant de combiner une documentation au format LaTeX et du code exécutable. Deux progammes utilisés sur la même source (`notangle` et `noweave`) permettent pour le premier d'extraire le code source exécutable et pour le second d'extraire le code LaTeX.

Le code source est inséré entre les deux balises suivantes
```
<< options >>=
# Code R
@
```
qui doivent être placées en début de ligne, sans être précédées par des espaces.

Cette syntaxe est utilisée dans l'exemple de la figure 2. C'est la syntaxe que nous utiliserons en général (sauf dans la section dédiée à la syntaxe LaTeX).

FIGURE 2 – Exemple d'utilisation de la syntaxe noweb

exemple1.rnw
```
\documentclass[a4paper,10pt]{article}
\usepackage[latin1]{inputenc}
\usepackage[french]{babel}

\title{Premier exemple d'utilisation de \texttt{Sweave}\\
       Utilisation de la Syntaxe noweb}
\author{S. Mousset et A.B. Dufour}

\begin{document}

\maketitle

\begin{abstract}
 Ceci est un premier exemple d'utilisation de \texttt{Sweave} pour générer un
 rapport scientifique. Dans le premier segment de code R, seules les commandes
 apparaissent. Dans le second segment, le code R et les résultats générés apparaissent.
\end{abstract}

Lecture des données:
<<label=readdata, eval=TRUE, results=hide, echo=TRUE>>=
data(iris)
@

Utilisation d'un test Kruskal-Wallis pour répondre à la question: ``Les tailles
des sépales diffèrent-elles entre espèces?''

<<label=test, eval=TRUE, results=verbatim, echo=TRUE>>=
kruskal.test(Sepal.Length ~ Species, data=iris)
@

\end{document}
```

Par convention, les noms des fichier écrits en syntaxe noweb utilisent l'une des extension suivantes : `.Snw`, `.snw`, `.Rnw` ou `.rnw`.

3.2 La syntaxe LaTeX

Une syntaxe plus proche de la syntaxe de LaTeX est aussi proposée. Le code source est alors inséré entre deux balises `\begin{Scode}` et `\end{Scode}`. Cette syntaxe est utilisée dans l'exemple de la figure 3.

Par convention, les noms des fichier écrits en syntaxe noweb utilisent l'une des extension suivantes : `.Stex`, `.stex`, `.Rtex` ou `.rtex`. Malheureusement, `RStudio` ne reconnaît pas (encore) cette syntaxe.
Le type de syntaxe utilisée peut être changé à l'intérieur d'un document en utilisant la commande
`\SweaveSyntax{SweaveSyntaxLatex}` ou `\SweaveSyntax{SweaveSyntaxNoweb}`

FIGURE 3 – Exemple d'utilisation de la syntaxe LaTeX

exemple1.rtex

```
\documentclass[a4paper,10pt]{article}
\usepackage[latin1]{inputenc}
\usepackage[french]{babel}

\title{Premier exemple d'utilisation de \texttt{Sweave}\\
       Utilisation de la Syntaxe \LaTeX}
\author{S. Mousset et A.B. Dufour}

\begin{document}

\maketitle

\begin{abstract}
 Ceci est un premier exemple d'utilisation de \texttt{Sweave} pour générer un
 rapport scientifique. Dans le premier segment de code R, seules les commandes
 apparaissent. Dans le second segment, le code R et les résultats générés apparaissent.
\end{abstract}

Lecture des données:
\begin{Scode}{label=readdata, eval=TRUE, results=hide, echo=TRUE}
data(iris)
\end{Scode}

Utilisation d'un test Kruskal-Wallis pour répondre à la question: ``Les tailles
des sépales diffèrent-elles entre espèces?''

\begin{Scode}{label=test, eval=TRUE, results=verbatim, echo=TRUE}
kruskal.test(Sepal.Length ~ Species, data=iris)
\end{Scode}

\end{document}
```

3.3 En dehors des *code chunks* : l'utilisation de `Sexpr{}`

Alors que jusqu'à présent nous avons inséré des sections de code R à l'intérieur du document, il peut être nécessaire d'insérer une valeur à l'intérieur du texte. Ceci peut s'effectuer à l'aide de la commande `\Sexpr{}`. Une utilisation est donnée dans l'exemple de la figure 4.

4 Les principales options de `Sweave`

Une liste exhaustive des options de `Sweave` peut être obtenue dans l'aide de `Rweavelatex` de R. Nous nous contenterons ici de décrire les options principales.

4.1 Régler des options à l'échelle du document entier

Nous avons vu dans les exemples précédents qu'il était possible de modifier les sorties de R grâce aux instructions appelées "options" placées entre les balises `<<` et `>>=`. Certaines de ces options peuvent être réglées à l'échelle d'une série de *code chunks* grâce à la commande `\\SweaveOpts{}`. Une option importante est `prefix.string` qui permet de fixer une chaîne de caractères qui servira de base pour les noms des fichiers générés. La ligne de code suivante indique par exemple que les *code chunks* doivent être évalués, que les commandes et les sorties sont affichées dans le document LaTeX et que les noms des fichiers générés commenceront par `figs/sweave` (en fait ils seront créés dans un répertoire `figs`) :

FIGURE 4 – Insertion directe de valeurs avec `Sexpr`

exemple2.rnw

```
\documentclass[a4paper,10pt]{article}
\usepackage[latin1]{inputenc}
\usepackage[french]{babel}

\title{Deuxième exemple d'utilisation de \texttt{Sweave}\\
       Utilisation de \texttt{Sexpr}}
\author{S. Mousset et A.B. Dufour}

\begin{document}

\maketitle

\begin{abstract}
 Ceci est un deuxième exemple d'utilisation de \texttt{Sweave} pour générer un
 rapport scientifique. Dans cet exemple tous les appels à R sont cachés
 et seuls des résultats sont insérés dans le texte à l'aide de \texttt{Sexpr}.
\end{abstract}

<<label=read_data_and_test, eval=TRUE, results=hide, echo=FALSE>>=
data(iris)
kwt <- kruskal.test(Sepal.Length ~ Species, data=iris)
@

On note un effet significatif de l'espèce sur la longueur moyenne des sépales des
fleurs d'iris (test de Kruskal-Wallis, $K=\Sexpr{signif(kwt[["statistic"]],digits=2)}$,
$p=\Sexpr{signif(kwt[["p.value"]],digits=2)}$).

\end{document}
```

```
\SweaveOpts{eval=TRUE, echo=TRUE, results=verbatim, prefix.string=figs/sweave}
```

4.2 Les options générales à l'échelle d'un *code chunk*

Dans les exemples précédents, nous avons déjà rencontré des exemples d'utilisation des options `label`, `eval`, `echo` et `results`. Voici quelques options utilisables dans les déclarations de *code chunk*, les valeurs par défaut sont indiquées entre parenthèses.

- `echo` : booléen (`TRUE`), indique s'il faut inclure le code source R dans le fichier de sortie.
- `eval` : booléen (`TRUE`), indique s'il faut évaluer le code source.
- `results` : chaîne de caractères (verbatim), la sortie se fait dans un environnement `Soutput` proche de l'environnement verbatim de LaTeX. Les autres possibilités sont "tex" (les commandes génèrent du code LaTeX) et "hide" (les résultats des commandes ne sont pas inclus dans le fichier de sortie).
- `term` : booléen (`TRUE`) les sorties texte ressemblent à celles d'une session R dans un terminal, les valeurs des assignations ne sont pas affichées mais les valeurs des objets le sont. Si `FALSE` alors l'affichage des sorties requiert l'utilisation de la commande `print()`.
- `split` : booléen (`FALSE`). Les sorties de texte sont incluses directement dans le fichier généré. Si `FALSE` alors un fichier séparé est créé pour les sorties de texte du *code chunk*.

⋆ `include` : booléen (`TRUE`). Indique si la sortie textuelle (si `split=TRUE`) ou graphique (si `fig=TRUE` doit être incluse automatiquement dans le fichier source à l'emplacement du *code chunk* (par un `\input` ou un `\includegraphics{}`).

Dans l'exemple de la figure 5, les sorties ne sont pas directement incluses à la place du *code chunk* (option `include=FALSE`) mais enregistrées dans des fichiers séparés (option `split=TRUE`) situés eux-mêmes dans un répertoire séparé (option `prefix.string=figs/sweave`).

FIGURE 5 – Inclusion séparée des sorties de *code chunk*

exemple4.rnw

```
\documentclass[a4paper,10pt]{article}
\usepackage[latin1]{inputenc}
\usepackage[french]{babel}

\title{Quatrième exemple d'utilisation de \texttt{Sweave}\\
       Sortie dans des fichiers inclus séparément\\
       (option \texttt{include=FALSE}).}
\author{S. Mousset et A.B. Dufour}

\SweaveOpts{eval=TRUE, echo=TRUE, results=verbatim, prefix.string=figs/sweave}

\begin{document}

\maketitle

\begin{abstract}
 Ceci est un quatrième exemple d'utilisation de \texttt{Sweave} pour générer un
 rapport scientifique. Dans cet exemple les sorties des codes R sont enregistrées
 dans des fichiers \LaTeX séparés qui sont inclus ensuite manuellement dans le document.
\end{abstract}

Une analyse de variance permet de montrer que la taille moyenne des sépales n'est
pas égale entre les espèces. Les commandes R utilisées pour procéder à l'analyse
et le résultat de l'analyse de variance sont présentés dans la figure \ref{fig:anova}.

<<label=analyse, split=TRUE, term=FALSE, results=hide, include=FALSE>>=
data(iris)
modele <- lm(Sepal.Length ~ Species, data=iris)
anova(modele)
@
<<label=anova, eval=TRUE, echo=FALSE, results=tex, include=FALSE, split=TRUE>>=
library(xtable)
analyse <- anova(modele)
tableau <- xtable(analyse)
print(tableau, floating=FALSE)
@

\begin{figure}[ht!]
\caption{Commandes R et résultats de l'analyse de variance}\label{fig:anova}
\vspace{0.5 cm}
\begin{minipage}{\textwidth}
  \small
  \input{figs/sweave-analyse.tex}
  \vspace{0.5 cm}
  \input{figs/sweave-anova.tex}
\end{minipage}
\end{figure}

\end{document}
```

4.3 Génération de figures dans un *code chunk* et inclusion de figures dans un document

Bien sûr, l'inclusion de figures générées avec R est possible (et nous avons gardé le meilleur pour la fin). Ces figures peuvent être enregistrées sous différents formats et intégrées automatiquement ou non dans le document LaTeX créé.

Les options à utiliser pour les sorties graphiques sont décrites ci-dessous (les valeurs par défaut sont indiquées entre parenthèses) :

- ⋆ `fig` : booléen (`FALSE`). Indique si le *code chunk* produit une sortie graphique.
- ⋆ `eps` : booléen (`FALSE`). Indique si la sortie graphique doit être enregistrée au format `eps`.
- ⋆ `pdf` : booléen (`TRUE`). Indique si la sortie graphique doit être enregistrée au format `pdf`.
- ⋆ `png` : booléen (`FALSE`). Indique si la sortie graphique doit être enregistrée au format `png`.
- ⋆ `jpeg` : booléen (`FALSE`). Indique si la sortie graphique doit être enregistrée au format `jpeg`.
- ⋆ `width` : valeur numérique (`6`). Indique la largeur (exprimée en pouces) de la fenêtre graphique utilisée par R.
- ⋆ `height` : valeur numérique (`6`). Indique la hauteur (exprimée en pouces) de la fenêtre graphique utilisée par R.
- ⋆ `resolution` : valeur numérique (`300`). Indique la résolution (en points par pouces) utilisée pour générer les images `jpeg` ou `png`.

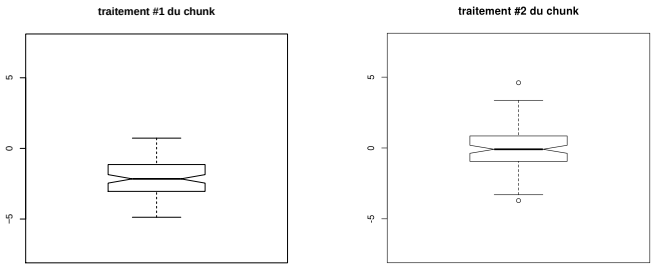
Attention : Un *code chunk* pour lequel `fig=TRUE` est exécuté autant de fois qu'il est demandé de formats de sortie. S'il n'en est pas tenu compte, cela peut conduire à des incohérences dans les figures générées (voir la figure 6).

La taille des graphiques inclus dans le document LaTeX peut être modifiée à l'aide de la variable `Gin` définie par `Sweave`. La valeur par défaut correspond à une largeur de 80% de la largeur de texte définie de la façon suivante : `\setkeys{Gin}{width=0.8\textwidth}`
Dans l'exemple de la figure 7, une figure de 9×6 pouces est générée au format `pdf` et incluse automatiquement après le *code chunk*. La variable `Gin` est modifiée en début de document pour que les images insérées dans le document final aient la largeur de la zone de texte.
Il peut être préférable de générer un graphique et de ne l'inclure que manuellement dans un environnement particulier. Un tel exemple est proposé dans la figure 8.

FIGURE 6 – Réutilisation du *code chunk* pour plusieurs formats graphiques.

Bug : les graphiques générés diffèrent.

Image au format `pdf` Image au format `png`



Origine du bug : l'entrée traitée par `Sweave` modifie les données du graphique :

hardbug.rnw

```
<<label=bug, fig=TRUE, pdf=TRUE, png=TRUE, echo=FALSE, results=hide>>=
if (exists("run")) {
  run <- run+1
} else {
  run <- 1
}
mu <- runif(1, min=-5, max=5)
sigma <- runif(1, min=0.5, max=2)
a <- rnorm(100, mean=mu, sd=sigma)
boxplot(a, notch=TRUE,
        main=paste("traitement #", run," du chunk", sep=""),
        ylim=c(-7.5,7.5))
@
```

FIGURE 7 – Génération et inclusion de graphiques

exemple5.rnw

```
\documentclass[a4paper,10pt]{article}
\usepackage[latin1]{inputenc}
\usepackage[french]{babel}
\usepackage{graphicx}

\title{Cinquième exemple d'utilisation de \texttt{Sweave}\\
       Inclusion de graphiques (option \texttt{fig=TRUE}).}
\author{S. Mousset et A.B. Dufour}

\SweaveOpts{eval=TRUE, echo=TRUE, results=verbatim, prefix.string=figs/sweave}

\begin{document}

\setkeys{Gin}{width=\textwidth}

\maketitle

\begin{abstract}
 Ceci est un cinquième exemple d'utilisation de \texttt{Sweave} pour générer un
 rapport scientifique. Dans cet exemple une figure est générée et automatiquement
 incluse dans le fichier \LaTeX{}.
\end{abstract}

La visualisation graphique des tailles de sépales à l'aide de boîtes à moustaches
permet de mettre en évidence des différences de longueur moyenne entre espèces.

<<label=boxplot, echo=TRUE, results=verbatim, fig=TRUE, pdf=TRUE, include=TRUE, width=9, height=6>>=
data(iris)
boxplot(Sepal.Length ~ Species, data=iris, notch=TRUE)
@

\end{document}
```

FIGURE 8 – Génération et inclusion séparées de graphiques et de texte

exemple6.rnw

```
\documentclass[a4paper,10pt]{article}
\usepackage[latin1]{inputenc}
\usepackage[french]{babel}
\usepackage{graphicx}

\title{Sixième exemple d'utilisation de \texttt{Sweave}\\
       Inclusion manuelle de graphiques\\
       (options \texttt{fig=TRUE, include=FALSE}).}
\author{S. Mousset et A.B. Dufour}

\SweaveOpts{eval=TRUE, echo=TRUE, results=verbatim, prefix.string=figs/sweave}
<<label=options, echo=FALSE, results=hide>>=
options(encoding="latin1")
@
\begin{document}

\maketitle

\begin{abstract}
 Ceci est un sixième exemple d'utilisation de \texttt{Sweave} pour générer un
 rapport scientifique. Dans cet exemple une figure est générée et une analyse
 statistique est réalisée. Les commandes R, le graphique et le résultat de
 l'analyse sont présentés dans une figure séparée.
\end{abstract}

À l'aide de la fonction \texttt{boxplot} de R, nous avons obtenu une représentation
graphique de la distribution des longueurs des sépales d'iris selon les espèces. Une
analyse de la variance a confirmé l'impression que la taille moyenne des sépales
diffère entre les espèces. Les commandes utilisées, la représentation graphique et
le résultat de l'analyse de variance sont présentés sur la figure \ref{fig:results}

<<label=analyse, echo=TRUE, results=hide, split=TRUE, fig=TRUE, pdf=TRUE, include=FALSE>>=
data(iris)
boxplot(Sepal.Length ~ Species, data=iris, notch=TRUE,
        main="Longueur des sépales")
modele <- lm(Sepal.Length ~ Species, data=iris)
anova(modele)
@

<<label=anova, echo=FALSE, split=TRUE, results=tex, include=FALSE>>=
library(xtable)
tableau <- xtable(anova(modele))
print(tableau, floating=FALSE)
@

  \begin{figure}[ht!]\caption{Étude de la taille des sépales d'iris avec R}\label{fig:results}
    \begin{center}
      \begin{tabular}{c c}
        \begin{minipage}{0.45\textwidth}
          \includegraphics[width=\textwidth]{figs/sweave-analyse.pdf}
        \end{minipage} &
        \begin{minipage}{0.45\textwidth}
          \small
          \input{figs/sweave-analyse.tex}
          \vspace{0.5 cm}
          \input{figs/sweave-anova.tex}
        \end{minipage} \\
      \end{tabular}
    \end{center}
  \end{figure}
\end{document}
```

5 Pour aller plus loin

Pour mieux comprendre les sorties de `Sweave`, on peut inspecter la sortie générée par `Sweave` lorsqu'il traite le *code chunk* simpliste de la figure 9 : les entrées et sorties sont insérées dans des environnements `Sinput` et `Soutput`, eux-mêmes insérés dans un environnement `Schunk`.

FIGURE 9 – Traitement d'un *code chunk* par R

Entrée traitée par `Sweave` :

basicchunk.rnw

```
<<label=basicchunk, term=TRUE, echo=TRUE, results=verbatim>>=
rnorm(5)
@
```

Sortie LaTeX de `Sweave` :

basicchunk.tex

```
\begin{Schunk}
\begin{Sinput}
> rnorm(5)
\end{Sinput}
\begin{Soutput}
[1] -1.048165086 -1.831860646 -1.417091737 -0.335820600  0.007615001
\end{Soutput}
\end{Schunk}
```

5.1 Personnaliser les entrées et sorties de R

Afin d'identifier clairement les entrées et les sorties de R, on peut redéfinir les environnements `Sinput`, `Soutput`, et `Schunk`. L'exemple de la figure 10 reprend l'exemple de la figure 8 pour lequel ces environnements ont été redéfinis (notez aussi le `\usepackage{color}` dans le préambule du document). Notez que dans cet exemple, le prompt classique de R a aussi été modifié dans le *code chunk* "options" pour rendre la sortie plus lisible.

5.2 Les problèmes d'encodage

En manipulant et en générant des fichiers de texte contenant des caractères spéciaux vous serez un jour confronté aux problèmes d'encodage. `Sweave` peut détecter sensible l'encodage utilisé par votre fichier source dans la ligne d'appel au package `inputenc` (ici pour un document utilisant l'encodage en latin 1) :
`\usepackage[latin1]{inputenc}`
Il est aussi utile d'indiquer à R quel jeu de caractères utiliser en fixant les options de R dans l'un des premiers *code chunk* à la façon des exemples des figures 8 et 10. Essayez de supprimer le *code chunk* "options" de ces deux figures et vous pourriez constater des dysfonctionnements surprenants selon votre système d'exploitation et le jeu de caractères utilisé par défaut par votre version de R. Des problèmes d'encodage de caractères peuvent survenir avec les apostrophes générées par R insérées dans les environnements `verbatim` :dans l'exemple de la figure 10, la commande `options(show.signif.stars=FALSE)` a été utilisée car ces caractères créaient des erreurs lors de la compilation LaTeX sous windows.

5.3 Réutiliser un *code chunk*

Il est possible de réutiliser un *code chunk* précédemment défini à l'intérieur d'un nouveau *code chunk*, à condition de l'avoir nommé avec l'option `label`. Le mode de rappel d'un *code chunk* dépend de la syntaxe utilisée. La figure 11 donne un exemple de réutilisation dans les deux syntaxes.

FIGURE 10 – Redéfinition des environnements `Sinput` et `Soutput`

exemple7.rnw

```
\documentclass[a4paper,10pt]{article}
\usepackage[latin1]{inputenc}
\usepackage[french]{babel}
\usepackage{graphicx}
\usepackage{color}

\title{Septième exemple d'utilisation de \texttt{Sweave}\\
       Personnalisation des environnements \\
       \texttt{Schunk}, \texttt{Sinput} et \texttt{Soutput}.}
\author{S. Mousset et A.B. Dufour}

\SweaveOpts{eval=TRUE, echo=TRUE, results=verbatim, prefix.string=figs/sweave}
<<label=options, echo=FALSE, results=hide>>=
options(encoding="latin1", prompt=" ", continue=" ", width = 85)
options(show.signif.stars=FALSE)
@

\begin{document}
\definecolor{Soutput}{rgb}{0,0,0.56}
\definecolor{Sinput}{rgb}{0.56,0,0}
\DefineVerbatimEnvironment{Sinput}{Verbatim}
{formatcom={\color{Sinput}},fontsize=\footnotesize, baselinestretch=0.75}
\DefineVerbatimEnvironment{Soutput}{Verbatim}
{formatcom={\color{Soutput}},fontsize=\footnotesize, baselinestretch=0.75}
% Code de Duncan Murdoch posté sur R-help le 07-MARS-2008 :
% This removes the extra spacing after code and output chunks in Sweave,
% but keeps the spacing around the whole block.
\fvset{listparameters={\setlength{\topsep}{0pt}}}
\renewenvironment{Schunk}{\vspace{\topsep}}{\vspace{\topsep}}

\maketitle

\begin{abstract}
 Ceci est un septième exemple d'utilisation de \texttt{Sweave} pour générer un
 rapport scientifique. Cet exemple reprend l'exemple 6, mais présente directement
 le tableau d'analyse de la variance proposé par R. Les entrées et sorties de R
 sont présentées dans des environnements \texttt{Sinput} et \texttt{Soutput}
 personnalisés. Les commandes R et le graphique sont présentés dans une figure séparée.
\end{abstract}

À l'aide de la fonction \texttt{boxplot} de R, nous avons obtenu une représentation
graphique de la distribution des longueurs des sépales d'iris selon les espèces. Une
analyse de la variance a confirmé l'impression que la taille moyenne des sépales
diffère entre les espèces. Les commandes utilisées, la représentation graphique et
le résultat de l'analyse de variance sont présentés sur la figure \ref{fig:results}

<<label=analyse, echo=TRUE, results=verbatim, split=TRUE, fig=TRUE, pdf=TRUE, include=FALSE>>=
data(iris)
boxplot(Sepal.Length ~ Species, data=iris, notch=TRUE,
        main="Longueur des sépales")
modele <- lm(Sepal.Length ~ Species, data=iris)
anova(modele)
@

  \begin{figure}[ht!]\caption{Étude de la taille des sépales d'iris avec R}\label{fig:results}
    \begin{center}
      \begin{tabular}{c c}
        \begin{minipage}{0.45\textwidth}
          \includegraphics[width=\textwidth]{figs/sweave-analyse.pdf}
        \end{minipage} &
        \begin{minipage}{0.45\textwidth}
          \small
          \input{figs/sweave-analyse.tex}
        \end{minipage} \\
      \end{tabular}
    \end{center}
  \end{figure}
\end{document}
```

FIGURE 11 – Réutilisation de *code chunks*

Réutilisation d'un *code chunk* en syntaxe noweb :

rappel.rnw

```
<<label=chunk1>>=
# Premier chunk
@

<<label=chunk2>>=
<<chunk1>>
# Le deuxième chunk commence par appeler les commandes du premier
@
```

Réutilisation d'un *code chunk* en syntaxe LaTeX :

rappel.rtex

```
\begin{Scode}{label=chunk1}
# Premier chunk
\end{Scode}

\begin{Scode}{label=chunk2}
\Scoderef{chunk1}
# Le deuxième chunk commence par appeler les commandes du premier
\end{Scode}
```