

Comment rédiger un rapport avec la commande **Sweave()** de

Pr JEAN R. LOBRY



Cette fiche donne quelques indications pour rédiger un rapport intégrant des analyses statistiques et des graphiques produits par .
On utilise ici l'encodage UTF-8 pour les caractères.

Table des matières

1	Introduction	2
2	Approche artisanale : copier/coller manuel	2
3	Approche professionnelle : utilisation de Sweave()	3
3.1	Qu'est ce que Sweave() ?	3
3.2	Pré-requis	4
3.3	Vérifier que tout marche bien	5
3.4	Intégration d'entrées et sorties de la console 	8
3.5	Incorporation de résultats dans le corps du texte	9
3.6	Inclusion de graphiques	12
3.7	Contrôle de la taille des graphiques	15
3.8	Guillemets à la française	17
	Références	18

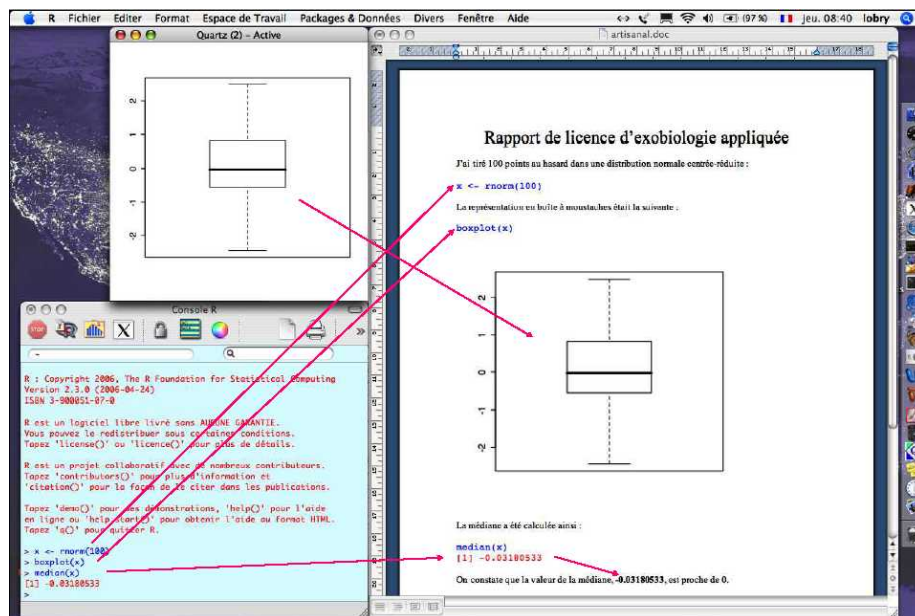
1 Introduction

REMARQUE liminaire : si vous souhaitez utiliser Sweave() dans un environnement hétérogène je recommande vivement l'utilisation de RStudio qui simplifie beaucoup les choses en homogénéisant l'environnement de travail¹. Il existe deux grandes approches pour rédiger un rapport faisant appel à des analyses et à des graphiques produits sous R.

1. Une première approche, que nous qualifierons d'artisanale, sans aucun sens péjoratif, est destinée à la production d'un nombre très restreint de rapports tous très courts. Cette approche est adaptée si vous n'envisagez de produire que des rapports de façon sporadique et que vous n'êtes pas intéressé par une carrière scientifique au sens très large du terme².
2. Une deuxième approche, que nous qualifierons de professionnelle, est plus adaptée à la production de nombreux rapports, ou d'un rapport de taille conséquente. Elle consiste à intégrer directement le code R dans le rapport et à utiliser R pour produire les graphiques et les analyses à intégrer directement dans le document final (oui c'est possible!). Ceci garanti la parfaite *reproductibilité* de vos résultats.

2 Approche artisanale : copier/coller manuel

Cette approche consiste simplement à ouvrir simultanément R et votre traitement de texte favori pour copier/coller les résultats qui vous intéressent :



1. Voir la fiche <http://pbil.univ-lyon1.fr/R/pdf/tdr781.pdf>
2. Contrairement à ce que l'on pourrait penser, de nombreux domaines des sciences dites, à tort, molles sont extrêmement exigeants pour ce qui est de la reproductibilité des résultats des analyses statistiques produites. Cette exigence n'est pas l'apanage de la recherche académique, elle concerne tous les secteurs où le *contrôle de la qualité* a un intérêt stratégique.

- Voici quelques indications utiles pour obtenir un résultat acceptable :
- Pour les graphiques, commencez par re-dimensionner la fenêtre graphique dans `R` jusqu'à ce que vous soyez satisfait du résultat. Copier/coller ensuite le graphique dans votre traitement de texte, puis re-dimensionnez le, de façon proportionnelle en x et y pour ne pas introduire de distorsions, pour obtenir la taille finale désirée.
 - Pour les commandes `R` entrées dans la console, copiez/collez les dans le traitement de texte puis supprimez le caractère d'invite de commande (par défaut le caractère '>') de la console `R`. Ceci vous permettra ultérieurement de facilement copier/coller dans l'autre sens, de votre traitement de texte vers `R`³. Définissez un style pour mettre en évidence qu'il s'agit de commandes `R`, par exemple en utilisant une police de caractères **non proportionnelle**, éventuellement de **couleur rouge** pour signifier qu'il s'agit de commandes `R` en entrée.
 - Pour les résultats obtenus dans la console, définissez un style similaire à celui des entrées, mais changez la couleur, par exemple en **couleur bleue**, pour mettre en évidence qu'il s'agit de résultats, et non d'entrées.
 - Pour intégrer des résultats numériques dans le corps du texte, il vous faudra faire un copier/coller de plus, veillez à maintenir la cohérence de l'ensemble !



CETTE approche permet de générer facilement de petits documents de qualité, mais elle trouve très rapidement ses limites car il est extrêmement fastidieux de maintenir la cohérence entre les différents éléments du rapport. Supposez par exemple que vous ayez envie de modifier un graphique, pour par exemple lui rajouter un titre. Il vous faudra alors générer le graphique modifié sous `R`, mais comme vous avez oublié de noter quelque part la dimension du périphérique utilisé dans la version précédente, vous allez perdre du temps pour retrouver un résultat acceptable. Puis il faudra copier/coller le graphique dans le traitement de texte, et de nouveau, le re-dimensionner à la main (et encore une fois vous n'avez pas noté ce que vous aviez fait la dernière fois, encore du temps perdu à tâtonner pour retrouver quelque chose d'à peu près similaire). Mais ce n'est pas fini : le code `R` n'est plus cohérent avec le graphique, il faut une nouvelle opération manuelle de copier/coller, et de remise en forme dans le traitement de texte, pour assurer la cohérence. Et, bien entendu, si vous avez intégré des résultats numériques dans le corps du texte, il faudra une troisième opération manuelle de copier/coller pour mettre tout le document en cohérence. On arrive très rapidement aux limites de l'exercice.

3 Approche professionnelle : utilisation de Sweave()

3.1 Qu'est ce que Sweave() ?

LA fonction `Sweave()` est une commande standard de `R` qui permet d'automatiser la production de documents en intégrant directement le code source `R` dans le document, la cohérence globale est alors automatiquement assurée. Comme `Sweave()` est une commande standard, il n'y a rien de particulier à installer si vous avez déjà `R` d'installé. D'une façon générale, il est beaucoup

³. Vous pouvez le faire directement au niveau de la console `R` en entrant la commande `options(prompt = " ")`.

plus simple de travailler sous des systèmes d'exploitations de type unix que sous Windows dès que l'on veut utiliser  de façon professionnelle. Ce n'est pas que cela soit impossible, le présent document est compilable sous Windows, mais il y a des analyses coût-bénéfice à prendre en compte. Ces dernières sont assez bien résumées de façon humoristique dans le folklore de  :

```
library(fortunes)
fortune("Knoppix")
Benjamin Lloyd-Hughes: Has anyone had any joy getting the rgdal package to compile
under windows?
Roger Bivand: The closest anyone has got so far is Hisaji Ono, who used MSYS
(http://www.mingw.org/) to build PROJ.4 and GDAL (GDAL depends on PROJ.4, PROJ.4
needs a PATH to metadata files for projection and transformation), and then
hand-pasted the paths to the GDAL headers and library into src/Makevars, running
Rcmd INSTALL rgdal at the Windows command prompt as usual. All of this can be
repeated, but is not portable, and does not suit the very valuable standard binary
package build system for Windows. Roughly: [points 1 to 5 etc omitted]
Barry Rowlingson: At some point the complexity of installing things like this for
Windows will cross the complexity of installing Linux... (PS excepting live-Linux
installs like Knoppix)
-- Benjamin Lloyd-Hughes, Roger Bivand, and Barry Rowlingson
R-help (August 2004)
```

Dans la suite de ce document, nous supposons que vous travaillez sous un système de type unix. Il en existe des distributions libres.


3.2 Pré-requis

VOUS n'avez pas besoin de connaître \LaTeX pour utiliser `Sweave()`, mais vous avez besoin d'avoir les outils nécessaires pour transformer un fichier source \LaTeX en un document PDF, par exemple `pdf \LaTeX` , qui est fourni avec la plupart des distributions \LaTeX . Vérifiez que vous êtes capable de compiler le petit fichier (`mini.tex`) source \LaTeX suivant :

```
mini.tex
\documentclass[french,a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{babel}
\begin{document}
\begin{Huge}Bonjour, Ô mon$\delta$e !\end{Huge}
\end{document}
```

Pour ce faire suivez les étapes suivantes :

1. Allez sur le site <https://pbil.univ-lyon1.fr/R/donnees/> pour enregistrer une copie du fichier `mini.tex` dans votre dossier de travail. Si vous n'avez pas de dossier de travail, créez en un.
2. Ouvrez le fichier `mini.tex` présent dans votre dossier de travail avec votre éditeur de texte favori. Le codage des caractères est en UTF-8, il peut être nécessaire de l'indiquer au moment de l'ouverture du fichier.
3. Vérifiez que dans votre éditeur de texte que le caractère « Ô » est bien présent dans la quatrième ligne : `\begin{Huge}Bonjour, Ô monδe !\end{Huge}`. Si ce n'est pas le cas, c'est sans doute que votre éditeur de texte n'a pas ouvert le fichier en UTF-8. Essayez encore.
4. Lancez la compilation en entrant la commande `pdflatex mini.tex` dans un terminal⁴.

⁴ Remarque : on peut aussi le faire directement sans sortir de  avec `system("pdflatex mini.tex")`


Vous devez obtenir un document PDF d'une page au format A4, numérotée 1 en bas, et sur le haut de laquelle est écrit en gros caractères :

Bonjour, Ô monde!

Ce petit test vous permet de vérifier deux choses :

1. Vous êtes capable de compiler un fichier source L^AT_EX qui contient des caractères accentués, comme ici le caractère « Ô ».
2. Le caractère grec delta minuscule, δ , est plaisant à lire. En effet, par rapport à la version initiale de 1986, le rendu du caractère δ a été fortement amélioré en 1992⁵. Si le rendu du caractère δ n'est pas bon (*cf.* ci-contre) c'est sans doute que votre version de L^AT_EX est légèrement obsolète et qu'il faut absolument faire une mise à jour.

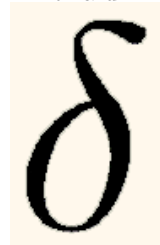
3.3 Vérifier que tout marche bien

Le logiciel  contient en standard un fichier d'exemple pour `Sweave()` qui s'appelle `Sweave-test-1.Rnw`. Lancez les exemples de `Sweave()` avec :

⁵. Voir <http://www-cs-faculty.stanford.edu/~uno/cm.html>



δ laid



δ beau

```

example(Sweave)

Sweave testfile <- system.file("Sweave", "Sweave-test-1.Rnw", package = "utils")
Sweave ## Don't show:
Sweave oldwd <- setwd(tempdir()) # so we will write only to a temp directory

Sweave ## End(Don't show)
Sweave
Sweave ## enforce par(ask = FALSE)
Sweave options(device.ask.default = FALSE)

Sweave ## create a LaTeX file - in the current working directory, getwd():
Sweave Sweave(testfile)
Writing to file Sweave-test-1.tex
Processing code chunks with options ...
 1 : keep.source print term verbatim (Sweave-test-1.Rnw:15)
 2 : keep.source term hide (Sweave-test-1.Rnw:17)
 3 : echo keep.source print term verbatim (Sweave-test-1.Rnw:22)
 4 : keep.source term verbatim (Sweave-test-1.Rnw:30)
 5 : echo keep.source term verbatim (Sweave-test-1.Rnw:45)
 6 : echo keep.source term verbatim pdf (Sweave-test-1.Rnw:53)
 7 : echo keep.source term verbatim pdf (Sweave-test-1.Rnw:63)

You can now run (pdf)latex on 'Sweave-test-1.tex'

Sweave ## This can be compiled to PDF by
Sweave ## tools::texi2pdf("Sweave-test-1.tex")
Sweave ## or outside R by
Sweave ## R CMD texi2pdf Sweave-test-1.tex
Sweave ## which sets the appropriate TEXINPUTS path.
Sweave ## create an R source file from the code chunks
Sweave Stangle(testfile)
Writing to file Sweave-test-1.R

Sweave ## which can be sourced, e.g.
Sweave source("Sweave-test-1.R")
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
[1] -1.16541359 -1.10186840 0.09354542 0.71054761 -0.29935222 -0.51535572
[7] 0.53660531 1.81081755 -1.32809173 -0.95501258 -1.43628366 -1.00165685
[13] 0.91139330 -1.54760437 -1.48602097 2.21096723 -0.37816369 1.31995096
[19] -0.30615810 -0.11758333

      One Sample t-test


data: x
t = -0.80061, df = 19, p-value = 0.4333
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -0.7309442 0.3264705
sample estimates:
mean of x
-0.2022369

Sweave ## Don't show:
Sweave if(!interactive()) unlink("Sweave-test-1*")

Sweave setwd(oldwd)

Sweave ## End(Don't show)
Sweave
Sweave
Sweave

```

Si tout s'est bien passé, vous devez avoir maintenant dans votre dossier courant un fichier `Sweave-test-1.tex`, ainsi qu'un certain nombre de fichiers graphiques au format pdf et un fichier de commandes . Utilisez pdfL^AT_EX pour compiler le fichier `Sweave-test-1.tex`, vous devez obtenir un fichier au format PDF de trois pages ressemblant à ceci :

3.4 Intégration d'entrées et sorties de la console

Enregistrez une copie du fichier `mini2.rnw` dans votre dossier de travail, disponible au même endroit ⁶.

```

mini2.rnw
\documentclass[french,a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{babel}
\begin{document}
Essayons de calculer 2 + 2 :
<<essai2plus2>>=
2 + 2
@
\end{document}
    
```

L'extension `rnw` désigne par convention un fichier destiné à être traité par la commande `Sweave()` de . Lancez la commande `Sweave()` en lui donnant comme argument le nom de ce fichier ⁷ :

```

Sweave("mini2.rnw")
Writing to file mini2.tex
Processing code chunks with options ...
1 : echo keep.source term verbatim (label = essai2plus2, mini2.rnw:8)
You can now run (pdf)latex on 'mini2.tex'
    
```

Vous pouvez alors compiler le fichier intermédiaire `mini2.tex` qui a été généré par `Sweave()` pour générer un document PDF contenant le texte suivant :

```

Essayons de calculer 2 + 2 :

> 2 + 2

[1] 4
    
```

Le fragments de code à intégrer dans le document final (ici `2 + 2`) doit être encadré au minimum par `<<>>=` au début et `@` à la fin, et il ne doit pas il y avoir de caractères en début de ligne. Le résultat (ici `[1] 4`) sera automatiquement intégré dans le document final. Ici, nous avons en plus donné un nom au fragment de code en utilisant `<<essai2plus2>>=` pour pouvoir le repérer facilement lors de la phase de compilation avec `Sweave()`.

Pour y voir plus clair, on décide de supprimer le caractère d'invite de commande de la console (`>`) et de colorier les entrées et sorties de façon différente. Récupérez le fichier `mini3.rnw` :

```

mini3.rnw
\documentclass[french,a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
    
```

6. <https://pbil.univ-lyon1.fr/R/donnees/>

7. Sous RStudio il suffit de cliquer sur « Compile PDF » pour produire le document final.


```

\usepackage{babel}
\usepackage{color}
\usepackage{pdfcolmk}

\begin{document}

\DefineVerbatimEnvironment{Sinput}{Verbatim}{formatcom = {\color[rgb]{0, 0, 0.56}}}
\DefineVerbatimEnvironment{Soutput}{Verbatim}{formatcom = {\color[rgb]{0.56, 0, 0}}}

<<options, echo = FALSE>>=
options(prompt = " ", continue = " ", width = 85)
@

Essayons de calculer 2 + 2 en couleur :
<<essai2plus2>>=
2 + 2
@
\end{document}
    
```

Lancez `Sweave()` sur ce fichier :

```

Sweave("mini3.rnw")
Writing to file mini3.tex
Processing code chunks with options ...
 1 : keep.source term verbatim (label = options, mini3.rnw:14)
 2 : echo keep.source term verbatim (label = essai2plus2, mini3.rnw:19)
You can now run (pdf)latex on 'mini3.tex'
    
```

Notez qu'il y a maintenant deux fragments de code :

1. Le premier qui s'appelle `options` et qui sert en autres à supprimer le caractère d'invite de commande.
2. Le deuxième qui s'appelle `essai2plus2` et qui calcule $2 + 2$ comme précédemment.

Compilez le fichier `mini3.tex`, vous devez obtenir le résultat suivant :


```

          Essayons de calculer 2 + 2 en couleur :

          2 + 2

          [1] 4
    
```

NOTEZ que le premier fragment de code (`options`) n'apparaît pas dans le document final, c'est parce que nous l'avons commencé avec `<<options, echo = FALSE>>=`, et que `echo = FALSE` signifie que l'on ne veut pas que le code apparaisse dans le document final.

LES autres modifications que nous avons apportées s'adressent à L^AT_EX, nous avons demandé à utiliser les bibliothèques `color` et `pdfcolmk` pour pouvoir avoir des couleurs dans le document final. Nous avons aussi redéfini les styles `Sinput` et `Soutput` pour formater les entrées et sorties de code .

3.5 Incorporation de résultats dans le corps du texte

Il est possible avec la commande `\Sexpr{}` d'insérer dans du texte la valeur d'une variable. Par exemple, le fichier source `mini4.rnw` :

Le volume V d'une sphère de rayon r est donné par :

$$V(r) = \frac{4}{3}\pi r^3$$

```
V <- function(r) 4/3 * pi * r^3
V(1)
```

```
[1] 4.18879
```

Le volume d'une sphère de rayon unité est donc de 4.18879020478639.

FIGURE 1 – Le syndrome de l'encodage de la mort qui tue. Si dans votre document terminal vous obtenez des caractères étranges en lieu et place des caractères accentués, comme ci-dessus, c'est qu'un des acteurs impliqué dans sa production n'est pas au courant du fait qu'il faut utiliser l'encodage UTF-8. Qui est le coupable ? Il y a trois acteurs impliqués. Le premier c'est votre éditeur de texte favori, mais il ne peut pas être incriminé puisque vous avez déjà réussi à éditer un texte avec des accents (*cf.* pré-requis). Le deuxième, L^AT_EX, ne peut pas non plus être accusé puisque vous avez déjà réussi à produire un texte avec des accents (*cf.* pré-requis). Reste **R**. Quand vous utilisez la commande `Sweave()`, celle-ci va lire le fichier `*.rnw` et écrire le fichier `*.tex`. Il faut aussi lui dire d'utiliser l'encodage UTF-8. Pour ce faire il suffit d'entrer la commande `options(encoding = "UTF-8")` dans **R** avant de lancer `Sweave()`.

mini4.rnw

```
\documentclass[french,a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{babel}
\usepackage{color}
\usepackage{pdfcolmk}

\begin{document}

\DefineVerbatimEnvironment{Sinput}{Verbatim}{formatcom = {\color[rgb]{0, 0, 0.56}}}
\DefineVerbatimEnvironment{Soutput}{Verbatim}{formatcom = {\color[rgb]{0.56, 0, 0}}}

<<options, echo = FALSE>>=
options(prompt = " ", continue = " ", width = 85)
@
Le volume  $V$  d'une sphère de rayon  $r$  est donné par :
$$
V(r) = \frac{4}{3}\pi r^3
$$
<<volsphere>>=
V <- function(r) 4/3*pi*r^3
V(1)
@
Le volume d'une sphère de rayon unité est donc de  $\text{\Sexpr{V(1)}}$ .
\end{document}
```

donne le résultat suivant :

```
options(encoding = "latin1")
Sweave("mini4.rnw")
Writing to file mini4.tex
Processing code chunks with options ...
 1 : keep.source term verbatim (label = options, mini4.rnw:14)
 2 : echo keep.source term verbatim (label = volsphere, mini4.rnw:22)
You can now run (pdf)latex on
```

Le volume V d'une sphère de rayon r est donné par :


$$V(r) = \frac{4}{3}\pi r^3$$

```
V <- fonction(r) 4/3 * pi * r^3
V(1)
```

```
[1] 4.18879
```

Le volume d'une sphère de rayon unité est donc de 4.18879020478639.

SI tout s'est bien passé, tout ne devrait pas s'être bien passé, vous devriez normalement souffrir ici du syndrome de l'encodage la mort qui tue. Corrigez le problème si besoin est (*cf.* figure 1). Ces problèmes d'encodage devraient progressivement s'atténuer avec la généralisation de l'UTF-8.

VOUS pouvez utiliser la fonction `round()` pour arrondir les résultats, par exemple `\Sexpr{round(V(1), 2)}` donnerait 4.19 dans le document final. Ceci permet donc d'insérer une, et une seule, valeur obtenue par le calcul dans  directement dans le document final. Mais il est également possible d'insérer dans le document final des tableaux, donc un grand nombre de valeurs, grâce au paquet `xtable`. Nous ne donnerons qu'un exemple simple. Le fichier source suivant :

```
----- mini5.rnw -----
\documentclass[french,a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{babel}
\usepackage{color}
\usepackage{pdfcolmk}

\begin{document}
\SweaveOpts{concordance=TRUE}

<<options, echo = FALSE>>=
library(xtable) # pour pouvoir produire des tables LaTeX dans R
data(crimtab) # pour charger le jeu de données (voir ?crimtab)
colnames(crimtab) <- as.numeric(colnames(crimtab))/2.54 # taille en pouces
rownames(crimtab) <- as.numeric(rownames(crimtab))*10 # majeur en mmm
crimtab[which(crimtab == 0, arr.ind = TRUE)] <- NA # pour virer les 0
crimtab <- crimtab[nrow(crimtab):1, ] # pour inverser l'ordre des lignes
matable <- xtable(x = crimtab, label = "Student",
  caption = "Les données utilisées par William Sealy Gosset,
  plus connu sous le nom de plume de 'Student' et du test statistique éponyme,
  en 1908 (\textit{Biometrika} \textbf{6}, 1-25). La collecte des
  données a été faite par MacDonell en 1902 (\textit{Biometrika}
  \textbf{1}, 177-227). Il s'agit d'une table de contingence obtenue
  en discrétisant la taille (exprimée ici en pouces, soit 2.54 cm, et
  portée en colonne) et la longueur du majeur de la main gauche
  (exprimée ici en mm et portée en ligne) de 3000 criminels adultes
  de sexe mâle écroués au pays de Galle et en Angleterre à la fin
  du XIX\mathrm{e} siècle. Voir \texttt{?crimtab} pour plus
```

```
d'informations.")
# Notez les doubles \\ nécessaires dans R, c'est la "double escape rule"
print(matable, file = "Student.tex", size = "tiny", NA.string = ".")
# Un veut des '.' au lieu des des NA
@
\input{Student.tex}

Les fameuses données utilisées par Student sont présentées dans la table \ref{Student}.

\end{document}
```

permet d'obtenir ceci :

```
Sweave("mini5.rnw")
Writing to file mini5.tex
Processing code chunks with options ...
1 : keep.source term verbatim (label = options, mini5.rnw:12)
You can now run (pdf)latex on 'mini5.tex'
```

	55	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	
130
134
133
132
131
130
129
128
127
126
125
124
123
122
121
120
119
118
117
116
115
114
113
112
111
110
109
108
107
106
105
104
103
102
101
100
99
98
97
96
95
94

Table 1: Les données utilisées par William Sealy Gosset, plus connu sous le nom de plume de 'Student' et du test statistique éponyme, en 1908 (*Biometrika* 6, 1-25). La collecte des données a été faite par MacDonell en 1902 (*Biometrika* 1, 177-227). Il s'agit d'une table de contingence obtenue en discrétisant la taille (exprimée ici en pouces, soit 2.54 cm, et portée en colonne) et la longueur du majeur de la main gauche (exprimée ici en mm et portée en ligne) de 3000 criminels adultes de sexe mâle écroués au pays de Galle et en Angleterre à la fin du XIX^e siècle. Voir `?crintab` pour plus d'informations.

Les fameuses données utilisées par Student sont présentées dans la table 1.

Il se peut que dans votre document final vous ayez `table ??` au lieu de `table 1`. C'est normal, nous avons fait référence à la table dans le texte avec `\ref{Student}` pour ne pas avoir à nous préoccuper de la numérotation des tables, c'est le travail de \LaTeX , pas le notre. Mais pour pouvoir faire son travail correctement, \LaTeX a besoin de compiler le document une première fois pour générer les tables, puis une deuxième fois pour résoudre les références. Il nous suffit donc d'invoquer \LaTeX une deuxième fois et les références seront résolues.

3.6 Inclusion de graphiques

La commande `Sweave()` permet d'inclure automatiquement des graphiques dans le document final (yes!). Commencez par créer dans votre dossier de

travail courant un dossier nommé `figs` pour rassembler toutes les figures au même endroit et éviter d'encombrer votre répertoire avec trop de fichiers⁸.

Le fichier source suivant :

```

----- mini6.rnw -----
\documentclass[french,a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{babel}
\usepackage{color}
\usepackage{pdfcolmk}

\begin{document}
\SweaveOpts{concordance=TRUE}

\DefineVerbatimEnvironment{Sinput}{Verbatim}{formatcom = {\color[rgb]{0, 0, 0.56}}}
\SweaveOpts{prefix.string = figs/essai, eps = FALSE, pdf = TRUE}
\setkeys{Gin}{width=0.8\textwidth}
Essays de faire un histogramme :
<<essaifig, echo = T, fig = T>>=
hist(rnorm(1000), col = "pink")
@
\end{document}


```

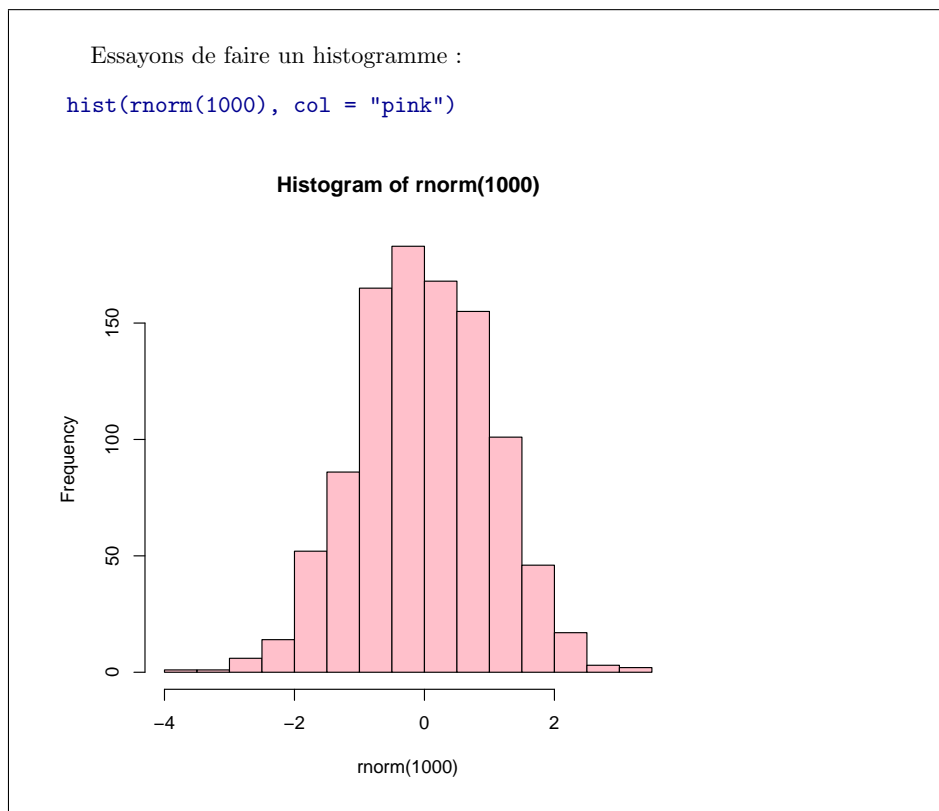
devrait vous permettre d'obtenir ceci :

```

Sweave("mini6.rnw")
Writing to file mini6.tex
Processing code chunks with options ...
 1 : echo keep.source term verbatim pdf (label = essaifig, mini6.rnw:16)
You can now run (pdf)latex on 'mini6.tex'

```

8. Mais il est aussi intéressant de désobéir à ces directives. À votre avis, si le dossier `figs` n'existe pas, il y aura-t-il un problème au moment de la compilation sous  ou bien un problème au moment de la compilation avec L^AT_EX ?



Quelques commentaires sur les instructions utilisées ici. La ligne `\SweaveOpts{prefix.string = figs/essai, eps = FALSE, pdf = TRUE}` sert à dire que nous voulons ranger les fichiers contenant les figures dans le dossier `figs` et que leur nom commence par `essai-`, vous devez normalement avoir dans ce dossier un fichier s'appelant `essai-essaifig.pdf` :

```
list.files(path = "figs", pattern = "essaifig")
[1] "empan-essaifig.pdf" "essai-essaifig.pdf"
```

le nom du fichier a été complété en ajoutant le nom du fragment de code (`essaifig`), puis l'extension `.pdf` pour signifier qu'il s'agit d'un fichier au format PDF. Notez que nous avons demandé à `Sweave()` de ne générer que des fichiers PDF avec l'option `eps = FALSE, pdf = TRUE`, parce que nous utilisons ici `pdfLATEX` et que nous n'avons pas l'utilité des fichiers au format EPS.

Le fragment de code commence ici avec la ligne `<<essaifig, echo = T, fig = T>>=` qui signifie que le fragment de code produit une figure (`fig = T`).

Une erreur fréquente consiste à commencer un fragment de code avec l'option `fig = T` mais que ce fragment de code ne produise en fait pas de figure. Le fichier PDF généré sera alors vide, ce qui conduira à une erreur quand `pdfLATEX` essaiera de l'inclure pour générer le document final.

Enfin, la ligne `\setkeys{Gin}{width=0.8\textwidth}` sert à contrôler la taille du graphique dans le document final. On signifie à L^AT_EX que l'on veut que le graphique occupe en largeur 80 % de la largeur d'une ligne courante.

3.7 Contrôle de la taille des graphiques

Voici une façon de procéder pour contrôler de façon fine et reproductible la dimension de vos graphiques. Commencez par mettre au point dans le mode interactif sous R votre graphique. Le plus simple pour cela est de créer un fichier source de code R appelé par exemple `mfigure.R` que vous exécutez avec `source("mfigure.R")`. Par exemple avec le fichier suivant :

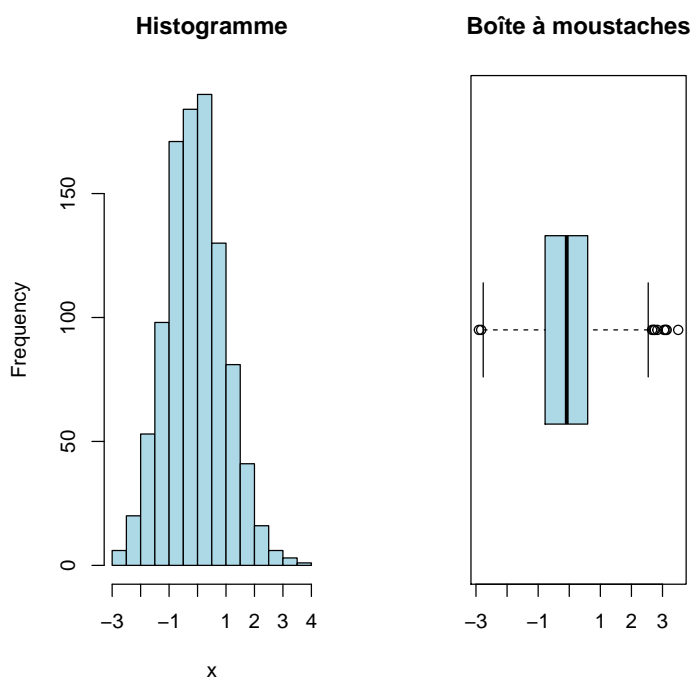
```

mfigure.R
par(mfrow = c(1,2))
x <- rnorm(1000)
hist(x, col = "lightblue", main = "Histogramme")
boxplot(x, col = "lightblue", horizontal = TRUE, main = "Boîte à moustaches")

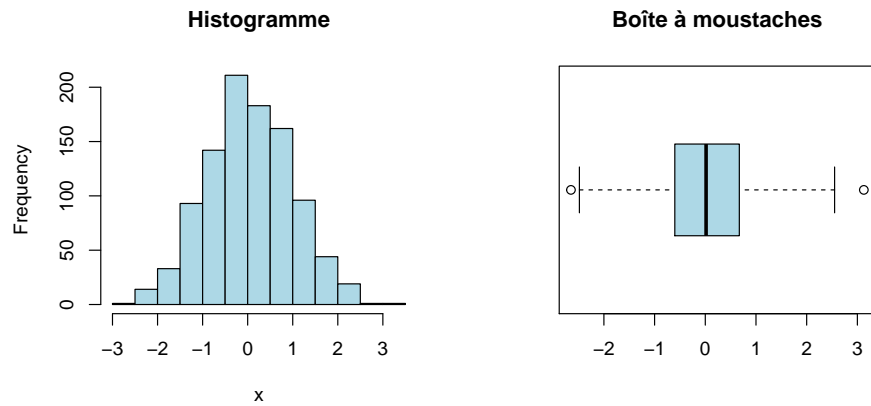
```

vous devez obtenir le résultat suivant :

```
source("mfigure.R")
```



Les dimensions générales des graphiques ne sont pas très adaptées, on préférerait avoir quelque chose de moins haut et de plus large. Quelque chose dans ce goût :



Toujours dans le mode interactif sous **R**, modifiez à la souris les dimensions de la fenêtre graphique, peaufinez le code source dans le fichier `mafigure.R`, jusqu'à ce que vous soyez satisfait du résultat. Utilisez alors la commande `par("din")` pour connaître les dimensions de la fenêtre graphique.

```
par("din")
[1] 7 7
```

La première valeur correspond à la largeur, la deuxième à la hauteur. Supposons que vous soyez satisfaits avec une largeur de 8 et une hauteur de 4. Le source suivant :

```

----- mini7.rnw -----
\documentclass[french,a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{babel}
\usepackage{color}
\usepackage{pdfcolmk}

\begin{document}
\SweaveOpts{concordance=TRUE}
\SweaveOpts{prefix.string = figs/essai, eps = FALSE, pdf = TRUE}
\setkeys{Gin}{width=\textwidth}
Je veux une figure de 8 unités de large par 4 unités de haut
et qui tienne toute la largeur de la page dans le document final.

<<essaifig, echo = F, fig = T, width = 8, height = 4>>=
par(mfrow = c(1,2))
x <- rnorm(1000)
hist(x, col = "lightblue", main = "Histogramme")
boxplot(x, col = "lightblue", horizontal = TRUE, main = "Boîte à moustaches")
@
\end{document}

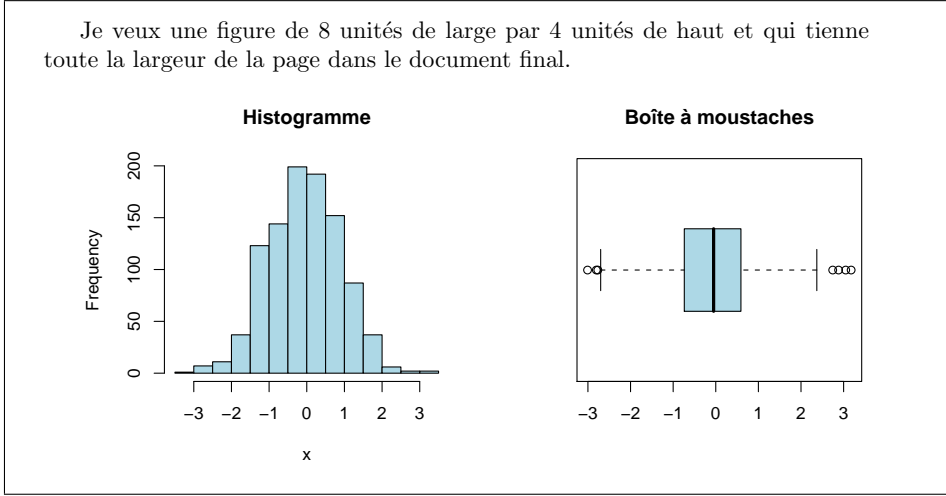
```

devrait vous permettre d'obtenir ceci :

```

Sweave("mini7.rnw")
Writing to file mini7.tex
Processing code chunks with options ...
 1 : keep.source term verbatim pdf (label = essaifig, mini7.rnw:16)
You can now run (pdf)latex on 'mini7.tex'

```

3.8 Guillemets à la française

La commande `frquote` du module `frenchb` de `babel` permet de gérer automatiquement les citations longues et imbriquées. Cette commande introduite dans la version 3.1 de `frenchb` en 2014 est beaucoup plus simple à utiliser que les `\og` et `\fg` de naguère. Si on ne précise rien on va avoir un conflit avec `Sweave()` et obtenir de gros carrés bien vilains au lieu des guillemets :

Essayons de calculer $2 + 2$ en couleur :

$2 + 2$

[1] 4

On me dit donc que ■ $2 + 2$ ■ vaut ■ 4 ■.

Le conflit vient de ce que `Sweave()` utilise par défaut le paquet `ea` qui interfère avec le module `frenchb` de `babel`. Il suffit de préciser que l'on n'en veut pas avec l'option `noae` :

```

----- mini8.rnw -----
\documentclass[french,a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage{babel}
\usepackage{color}
\usepackage{pdfcolmk}
\usepackage[noae]{Sweave}
\begin{document}

```

```

\SweaveOpts{concordance=TRUE}
\SweaveOpts{prefix.string = figs/essai, eps = FALSE, pdf = TRUE}
\DefineVerbatimEnvironment{Sinput}{Verbatim}{formatcom = {\color[rgb]{0, 0, 0.56}}}
\DefineVerbatimEnvironment{Soutput}{Verbatim}{formatcom = {\color[rgb]{0.56, 0, 0}}}

<<options, echo = FALSE>>=
options(prompt = " ", continue = " ", width = 85)
@

Essayons de calculer 2 + 2 en couleur :
<<essai2plus2>>=
2 + 2
@

On me dit donc que \frquote{2 + 2} vaut \frquote{4}.

\end{document}
    
```

```

Sweave("mini8.rnw")
Writing to file mini8.tex
Processing code chunks with options ...
 1 : keep.source term verbatim (label = options, mini8.rnw:15)
 2 : echo keep.source term verbatim (label = essai2plus2, mini8.rnw:20)
You can now run (pdf)latex on 'mini8.tex'
    
```

Essayons de calculer 2 + 2 en couleur :

2 + 2

[1] 4

On me dit donc que « 2 + 2 » vaut « 4 ».

Références

- [1] Max Kuhn and Steve Weaston. *odfWeave : Sweave processing of Open Document Format (ODF) files*, 2009. R package version 0.7.10.
- [2] F. Leisch. Sweave : Dynamic generation of statistical reports using literate data analysis. *Proceedings in Computational Statistics*, Compstat 2002 :575–580, 2002.
- [3] T. Van Zandt. *The 'fancyvrb' Package. Fancy Verbatims in L^AT_EX*, 1998. <http://ctan.tug.org/tex-archive/macros/latex/contrib/fancyvrb>.