



sdoc : rédaction d'un document long sous RStudio

Mode d'emploi

Joseph Larmarange

2013-12-15

sdoc permet de combiner la puissance de divers outils pour rédiger un document long directement dans *R Studio* en utilisant le format *markdown*. Par rapport au support natif du format *R markdown*, `sdoc` ajoute :

- la numérotation automatique des titres, tables et figures ;
- la génération d'une table des matières, d'une liste des tables et d'une liste des figures ;
- l'ajout automatique des numéros de titre, table ou figure aux liens internes ;
- amélioration de la ponctuation (limitée aux règles anglaises) ;
- la génération d'un fichier HTML avec une mise en forme moderne, incluant un accès facilité à la table des matières et aux notes de bas de page ;
- la génération d'une version PDF avec une mise en forme classique et l'ajout des numéros de page aux liens internes.

◆ R, RStudio, knitr, pandoc, Prince XML

Sommaire

1 À propos de sdoc	2
2 Pré-requis	2
2.1 R Studio	2
2.2 pandoc	2
2.3 Prince XML	3
3 Installation et utilisation	3
3.1 Installation manuelle	3
3.2 Utilisation	3
4 Paramètres du document	3
4.1 Bloc YAML de configuration	3
4.2 Autres paramètres de configuration	4
5 Format markdown	5
6 Titres et liens internes	5
Titre non numéroté	5

6.1 Titre avec identifiant	5
7 Inclure des chunk R	6
7.1 Bloc de code	6
7.2 Code en ligne	7
8 Tableaux	7
8.1 Fonction kable	7
8.2 Package xtable	8
8.3 Package tables	8
9 Graphiques	9
10 Notes de bas de page	11
11 Équations	11
12 Ponctuation intelligente	12
Liste des figures	12
Liste des tables	12

1 À propos de *scdoc*

scdoc est développé par Joseph Larmarange sur GitHub  :
<https://github.com/larmarange/scdoc>

Pour toute question, rapport de bug, proposition d'amélioration :
<https://github.com/larmarange/scdoc/issues>

2 Pré-requis

2.1 R Studio

scdoc nécessite que plusieurs outils soient installés sur votre système. En premier lieu, *scdoc* s'utilise dans la cadre d'un projet *R* géré avec l'interface de développement *R Studio*. Cette dernière, disponibles pour Windows Mac et Linux, peut être téléchargée à <http://www.rstudio.com/ide/>.

2.2 pandoc

La transformation du fichier *markdown* au format *HTML* est réalisée avec *pandoc*. Ce dernier peut être téléchargé à <http://johnmacfarlane.net/pandoc/installing.html>. Il importe que *pandoc* soit accessible en ligne de commande¹. ATTENTION : en raison de certains changement dans *pandoc*, veuillez utiliser une version récente (≥ 1.12).


1. Sous Windows, vous devrez peut-être ajouter manuellement le chemin de *pandoc* à la variable `path` du système. Pour vérifier si *pandoc* est accessible, ouvrez l'invite de commande Windows et entrez la commande `pandoc --help`. Si un message vous informe que `pandoc` n'est pas une commande reconnue, recherchez le répertoire où est situé le fichier `pandoc.exe` (probablement quelque chose comme `C:\Users\mon_nom\AppData\Local\Pandoc\`). Ajouter ce répertoire au `path` Windows (voir <http://sametmax.com/ajouter-un-chemin-a-la-variable-d'environnement-path-sous-windows/>).

2.3 Prince XML

Pour générer un fichier PDF², *Prince XML* doit être installé sur votre ordinateur. Il peut être téléchargé à <http://www.princexml.com/download/>. Comme pour *pandoc*, *Prince XML* doit être accessible en ligne de commande³.

3 Installation et utilisation

3.1 Installation manuelle

1. Téléchargez la dernière version de *scdoc* sur Github  (<https://github.com/larmarange/scdoc/archive/master.zip>).
2. Dézippez l'archive.
3. Copiez le sous-répertoire `/scdoc/` dans le répertoire de votre projet.

3.2 Utilisation

Au début de chaque session, exécutez la commande suivante :

```
R> source("scdoc/scdoc.r")
```

Et c'est tout ! Éditez votre fichier *R markdown* (`.Rmd`) dans *R Studio*. Cliquez sur le bouton **Knit HTML** pour produire la version HTML⁴ (et PDF le cas échéant) de votre document.

4 Paramètres du document

Note : n'hésitez pas à utiliser ce document comme modèle pour vos propres documents.

4.1 Bloc YAML de configuration

De nombreuses options peuvent être définies au début du document dans un bloc au format YAML. Ce bloc⁵ doit impérativement commencer par une ligne contenant uniquement `---` et se terminer par une ligne contenant uniquement `...`. Tous les paramètres sont optionnels.

2. La génération d'un PDF est optionnelle, voir section 4 page 3.

3. Sous Windows, le répertoire `C:\Program Files (x86)\Prince\Engine\bin` devra être ajouté à la variable d'environnement `path`.

4. Le fichier HTML produit est autonome et peut donc être diffusé tel quel.

5. Voir à titre d'exemple https://github.com/larmarange/scdoc/blob/master/Mode_d_emploi.Rmd.

Table 1. Paramètres du bloc YAML initial

lang	Langue du document au format ISO alpha 2, <code>'en'</code> pour l'anglais, <code>'fr'</code> pour le français
title	Titre du document
author	Auteur(s) du document (plusieurs entrées possibles)
date	Date du document (au format AAAA-MM-JJ, pour générer automatiquement la date du jour, saisir <code>`r format(Sys.time(), "%Y-%m-%d")`</code>)
tags	Mots-clés
abstract	Résumé (doit commencer par <code> </code> pour un résumé avec plusieurs paragraphes, ces derniers devant alors être indentés)
toc-title	Titre de la table des matières
tof-title	Titre de la table des figures
tot-title	Titre de la table des tables
prefix-fig	Préfixe pour la numérotation des figures (<i>Figure</i> par défaut)
prefix-table	Préfixe pour la numérotation des tables (<i>Table</i> par défaut)
pdf	Pour générer automatiquement un PDF avec <i>Prince XML</i> , cette option doit être égale à <code>'prince'</code>

4.2 Autres paramètres de configuration

Nous vous recommandons de placer au début de votre document le code suivant :

```
```${r configuration, echo=FALSE, message=FALSE}
opts_chunk$set(
 comment = NA, # Ne pas afficher ## devant les résultats
 dev = "svg", # Graphiques au format SVG
 cache = TRUE, # Mise en cache
 fig.width = 8, fig.height = 6 # Taille par défaut des figures
)
options(
 knitr.table.format = "html", # Tables directement au format HTML (kable)
 xtable.type = "html", # Tables en HTML (xtable)
 xtable.caption.placement = "top" # Titre avant le tableau (xtable)
)
knit_hooks$set(plot = hook_plot_html) # Figures directement au format HTML
windowsFonts(sans=windowsFont("Open Sans")) # Police par défaut des
graphiques
```
```

Il s'agit de différentes options pour *knitr*. Les options passées via `opts_chunk$set` sont optionnelles. Libre à vous de les adapter en fonction de vos besoin. Pour les graphiques, nous avons privilégié le format SVG car ce dernier permet un rendu de meilleur qualité dans le PDF produit tout en étant correctement interprété par les navigateurs modernes.

`options(knitr.table.format = "html")` et `knit_hooks$set(plot = hook_plot_html)` sont nécessaires pour une gestion adéquate des identifiants et des titres des tableaux et des figures (voir section 8 page 7 et section 9 page 9 pour plus de détails). Les options `xtable.type` et `xtable.caption.placement` ne sont nécessaires que si vous utilisez le package *xtable*.

`windowsFonts(sans=windowsFont("Open Sans"))` est totalement optionnel. Cela permet de spécifier la polices de caractères à utiliser lors de la création de graphiques.

5 Format markdown

Dans la mesure où scdoc utilise pandoc pour interpréter le code markdown, il est possible d'utiliser tous les ajouts à markdown proposés par pandoc. Ces derniers sont décrits en détails à <http://johnmacfarlane.net/pandoc/README.html#pandocs-markdown>.

6 Titres et liens internes

Les titres sont automatiquement numérotés. Il est cependant possible de spécifier qu'un titre particulier ne doit pas être numéroté en lui ajoutant la classe CSS `.unnumbered`.

```
## Titre non numéroté {.unnumbered}
```

produira :

Titre non numéroté

Pour créer un lien interne vers un titre donné, il faut en premier lieu attribuer un identifiant unique à votre titre :

```
## Titre avec identifiant {#id_titre}
```

produira :

6.1 Titre avec identifiant

Pour créer un lien interne, il suffit dès lors de cibler le titre désiré avec son identifiant.

```
Voir [section](#id_titre).
```

produira :

Voir section 6.1 page 5.

Vous remarquerez au passage que le numéro du titre a été automatiquement ajouté (ainsi que le numéro de page dans la version PDF).

7 Inclure des chunk R

7.1 Bloc de code

```
``{r}
summary(cars)
``
```

produira

```
R> summary(cars)

      speed      dist
Min.   : 4.0    Min.   : 2
1st Qu.:12.0    1st Qu.: 26
Median :15.0    Median : 36
Mean   :15.4    Mean   : 43
3rd Qu.:19.0    3rd Qu.: 56
Max.   :25.0    Max.   :120
```

Autre exemple :

```
``{r}
summary(cars)
str(cars)
sum(cars$speed)
``
```

```
R> summary(cars)

      speed      dist
Min.   : 4.0    Min.   : 2
1st Qu.:12.0    1st Qu.: 26
Median :15.0    Median : 36
Mean   :15.4    Mean   : 43
3rd Qu.:19.0    3rd Qu.: 56
Max.   :25.0    Max.   :120
```

```
R> str(cars)

'data.frame':   50 obs. of  2 variables:
 $ speed: num  4 4 7 7 8 9 10 10 10 11 ...
 $ dist : num  2 10 4 22 16 10 18 26 34 17 ...
```

```
R> sum(cars$speed)

[1] 770
```

Pour plus de détails, voir <http://yihui.name/knitr/>.

7.2 Code en ligne

8 Tableaux

Les tableaux sont automatiquement numérotés. De même, une table des tables est générée automatiquement. Le préfixe des numéros de table et le titre de la table des tables peuvent être personnalisés (voir chapitre 4.1 page 3).

Plusieurs fonctions et package R permettent de produire des tableaux correctement formatés en HTML (voir ci-après). Pour chaque package, nous présenterons comment personnaliser le titre du tableau et comment lui attribuer un identifiant. L'ajout d'un identifiant permet de réaliser un lien interne. Par exemple :

```
Voir [table](#exemple_kable).
```

produira

Voir table 2 page 8.

8.1 Fonction kable

Il vous faut une version récente de *knitr* afin que soit disponible l'argument `caption`. La dernière version de *knitr* peut-être installée avec la commande :

```
R> require(devtools)
install_github('knitr', 'yihui')
```

En premier lieu, il faut calculer un tableau (avec une fonction telle que `table` ou `xtabs`).

```
R> data(Titanic)
d <- as.data.frame(Titanic)
tab <- xtabs(Freq~Class+Survived, data=d)
```

Ensuite, on appellera la fonction `kable` dans un chunk ayant pour option `results='asis'` afin que le code produit par `kable` soit inclus tel quel. Il importe également que l'on ait précisé à *knitr* de produire les tableaux au format HTML (`options(knitr.table.format = "html")`, voir section 4.2 page 4).

Le paramètre `caption` (optionnel) permet d'indiquer un titre de tableau. Pour ajouter un identifiant, on utilisera le paramètre `table.attr`. Enfin, il est possible de spécifier l'alignement de chaque colonne avec `align`.

```
R> kable(tab, table.attr = 'id="exemple_kable"',
caption = "Tableau généré avec kable",
align = c('l','r','r'))
```

Table 2. Tableau généré avec kable

| | No | Yes |
|------|-----|-----|
| 1st | 122 | 203 |
| 2nd | 167 | 118 |
| 3rd | 528 | 178 |
| Crew | 673 | 212 |

8.2 Package xtable

Le package `xtable` offre de multiples possibilités pour générer un tableau au format HTML. Il est recommandé de personnaliser au préalable les options `xtable.type` et `xtable.caption.placement` (voir section 4.2 page 4).

Comme pour `kable`, `xtable` sera appelée dans un chunk ayant pour option `results='asis'`. La syntaxe est la suivante :

```
R> require(xtable)
xtable(tab, caption="Tableau généré avec xtable",
       label="exemple_xtable",
       align=c("lrr"))
```

Table 3. Tableau généré avec xtable

| | No | Yes |
|------|--------|--------|
| 1st | 122.00 | 203.00 |
| 2nd | 167.00 | 118.00 |
| 3rd | 528.00 | 178.00 |
| Crew | 673.00 | 212.00 |

8.3 Package tables

Pour une gestion aisée du titre du tableau, une version récentes (0.7.67 ou plus) du package `tables` est requise. La dernière version peut être installée avec la commande :

```
R> install.packages("tables", repos="http://R-Forge.R-project.org")
```

Dans un premier temps, le tableau sera calculé avec la fonction `tabular` (voir l'aide de cette dernière pour plus de détails) :

```
R> require(tables)
tab2 <- tabular( (Species + 1) ~ (n=1) + Format(digits=2)*
                (Sepal.Length + Sepal.Width)*(mean + sd), data=iris )
```

Le tableau sera ensuite converti au format HTML avec la fonction `html`. Cette dernière sera appelée dans un chunk ayant pour option `results='asis'`. Petite particularité, le titre devra être indiqué avant d'appeler `html` avec la fonction `table_options`. Il est recommandé de remettre cette valeur à `NULL` juste après pour éviter d'impacter les tableaux suivants.


```
R> table_options(HTMLcaption = 'Tableau généré avec tabular et htm
html(tab2, id='exemple_tabular')
```

Table 4. Tableau généré avec tabular et html

| Species | n | Sepal.Length | | Sepal.Width | |
|-------------------|-----|--------------|------|-------------|------|
| | | mean | sd | mean | sd |
| setosa | 50 | 5.01 | 0.35 | 3.43 | 0.38 |
| versicolor | 50 | 5.94 | 0.52 | 2.77 | 0.31 |
| virginica | 50 | 6.59 | 0.64 | 2.97 | 0.32 |
| All | 150 | 5.84 | 0.83 | 3.06 | 0.44 |

```
R> table_options(HTMLcaption = NULL)
```

9 Graphiques

Selon la norme HTML 5, on aura recours à la balise `<figure>` pour inclure des graphiques ou d'autres médias. Une même figure peut contenir plusieurs graphiques. Le titre d'une figure sera indiqué avec une balise `<figcaption>`.

Les figures sont automatiquement numérotées. De même, une table des figures est générée automatiquement. Le préfixe des numéros de figure et le titre de la table des figures peuvent être personnalisés (voir chapitre 4.1 page 3).

Les balises `<figure>` et `<figcaption>` seront directement entrées dans le fichier markdown. (NB : cette approche fonctionnera sous réserve que l'option `knit_hooks$set(plot = hook_plot_html)` ait été appliquée au début du document, voir section 4.2 page 4.)

```
<figure id="exemple_fig">
````{r, echo=FALSE}
plot(cars)
````
<figcaption>Titre de la figure</figcaption>
</figure>
```

produira

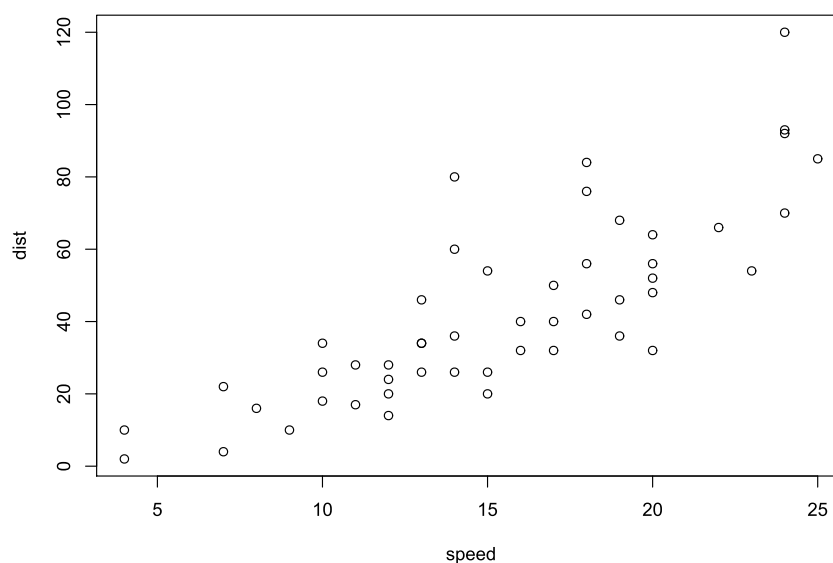


Figure 1. Titre de la figure

L'ajout d'un identifiant permet de réaliser un lien interne. Par exemple :

Voir `[figure](#exemple_fig)`.

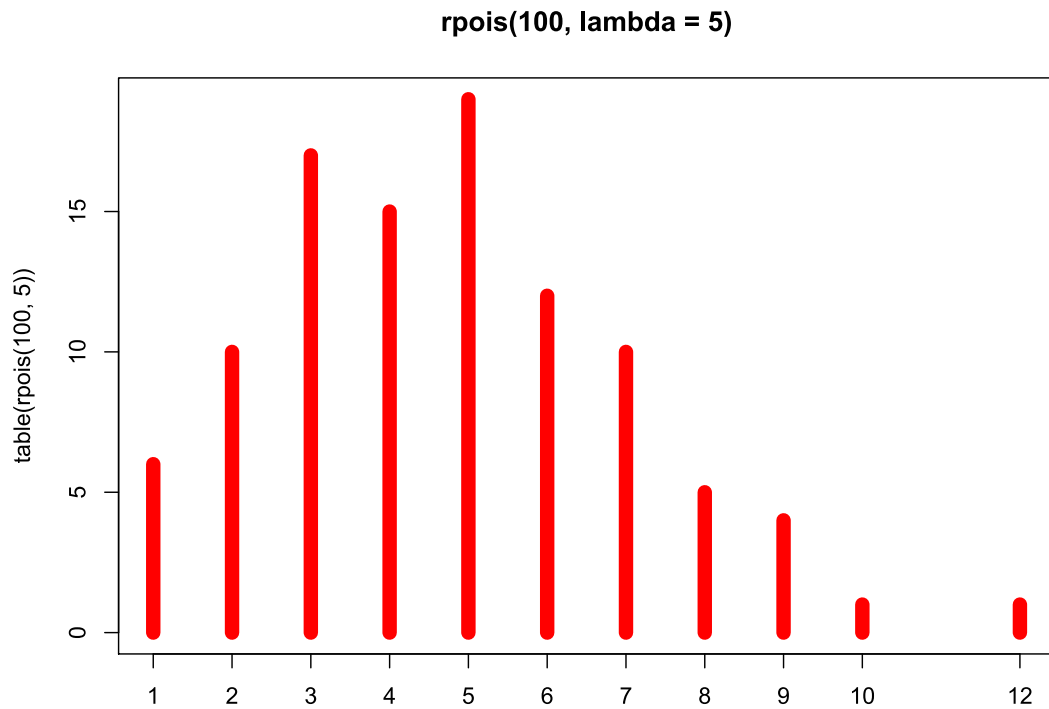
produira

Voir table 1 page 10.

Un graphique créé sans être encadré par une balise `<figure>` ne sera pas ajouté à la table des figures.

```
```{r, echo=FALSE}
plot(table(rpois(100, 5)), type = "h", col = "red", lwd = 10,
 main = "rpois(100, lambda = 5)")
```
```

produira



10 Notes de bas de page

Ceci est un texte^[Avec une note de bas de page].

produira :

Ceci est un texte⁶.

pandoc reconnaît plusieurs syntaxes pour rédiger des notes de bas de page (voir <http://johnmacfarlane.net/pandoc/README.html#footnotes> pour plus de détails).

11 Équations

Il est possible d'écrire des équations⁷ au format LaTeX.

6. Avec une note de bas de page

7. Les équations seront transformées au format MathML qui n'est pas correctement interprété par tous les navigateurs. La bibliothèque MathJax sera ajoutée dans une prochaine version.

```
\alpha+\beta=\gamma$
\int_0^\infty e^{-x^2} dx=\frac{\sqrt{\pi}}{2}
```

produira :

$$\alpha + \beta = \gamma$$

$$\int_0^\infty e^{-x^2} dx = \frac{\sqrt{\pi}}{2}$$

Il est également possible d'inclure directement du code HTML, qui sera conservé tel quel lors de l'export.

12 Ponctuation intelligente

L'option ponctuation intelligente de *pandoc*⁸ est active. Cette dernière gère les points de suspensions (...), les tirets semi-cadratin (–), les tirets cadratin (—) et les guillemets anglais (“”).

Par exemple :

```
--   ---   ...   "test"
```

produira

```
– — ... “test”
```

La ponctuation française n'est pas encore gérée mais pourra être ajoutée dans une future version.

Liste des figures

Figure 1. Titre de la figure 10

Liste des tables

Table 1. Paramètres du bloc YAML initial 4

8. <http://johnmacfarlane.net/pandoc/README.html#smart-punctuation>

| | |
|--|---|
| Table 2. Tableau généré avec kable | 8 |
| Table 3. Tableau généré avec xtable | 8 |
| Table 4. Tableau généré avec tabular et html | 9 |