La famille *down

Nathalie Vialaneix, INRA/MIAT

RUG Toulouse, November 18th 2019

What is Markdown? (*.md files)

According to **Wikipedia**,

> *Markdown is a lightweight markup language with plain text formatting syntax. Its design allows it to be converted to many output formats.*

Why Markdown?

- formatting text and documents is easy
- probably the most used lightweight markup language for numeric documents (especially technical documentations)

But: is sometimes a bit limited if you want to have custom outputs

What is **R**Markdown? (*.Rmd files)

- developed by RStudio (and especially by Yihui Xie)
- combines Markdown documents with **R** scripts (or python, C++, bash, ...) to produce dynamic (and even interactive) documents

What is **R**Markdown? (*.Rmd files)

- developed by RStudio (and especially by Yihui Xie)
- combines Markdown documents with **R** scripts (or python, C++, bash, ...) to produce dynamic (and even interactive) documents
- also available as **R** notebooks (similar to Jupyter notebooks)

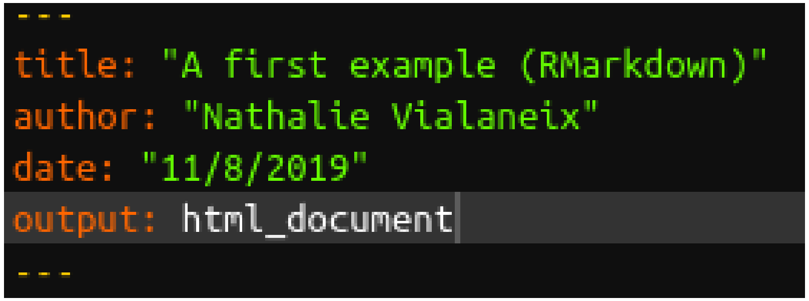
Basic RMarkdown documents

What do you need? (certified exact on linux)

- **R**
- RStudio (not mandatory but highly recommended) and if you don't know how to install it on linux, please, ask your sysadmin
- **packages**: at least `knitr` but other packages (`rmarkdown`, `markdown`, ...)
- pandoc to obtain different output types from `*.md` files: Rmd $\rightarrow$ md $\rightarrow$ (with pandoc) HTML
- it is best to have $\LaTeX$ installed to render maths or obtain PDF documents

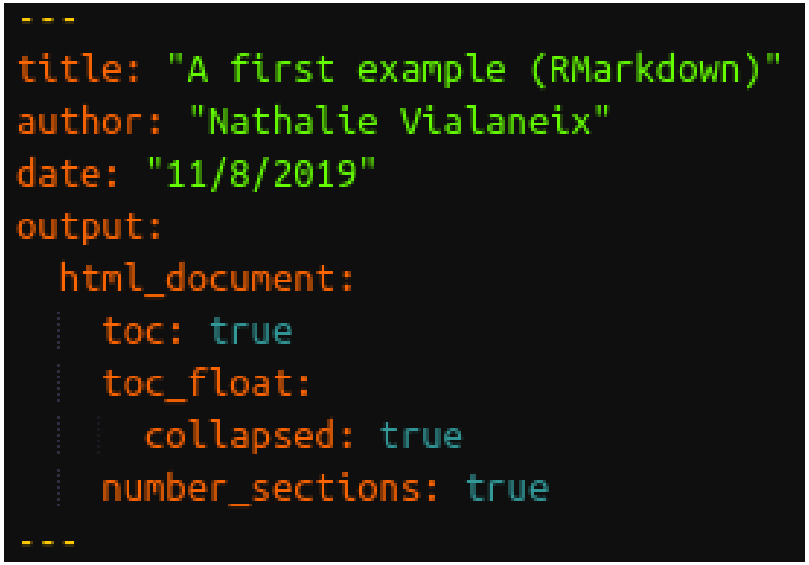
Basic demo...

Yaml headers

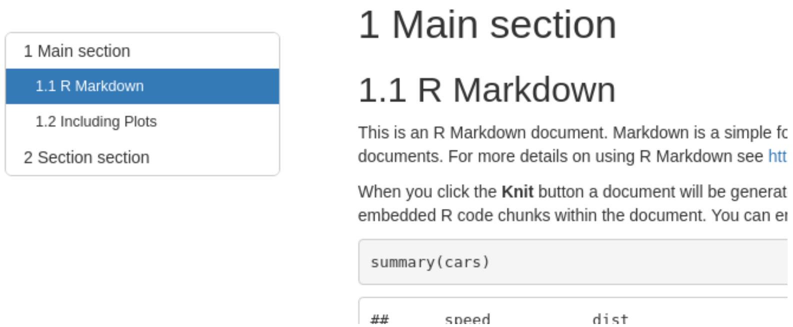


```
---
title: "A first example (RMarkdown)"
author: "Nathalie Vialaneix"
date: "11/8/2019"
output: html_document
---
```

Yaml headers



```
---
title: "A first example (RMarkdown)"
author: "Nathalie Vialaneix"
date: "11/8/2019"
output:
  html_document:
    toc: true
    toc_float:
      collapsed: true
    number_sections: true
---
```



1 Main section
1.1 R Markdown
1.2 Including Plots
2 Section section

1 Main section

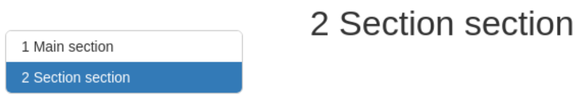
1.1 R Markdown

This is an R Markdown document. Markdown is a simple fc
documents. For more details on using R Markdown see htt

When you click the **Knit** button a document will be generat
embedded R code chunks within the document. You can e

```
summary(cars)
```

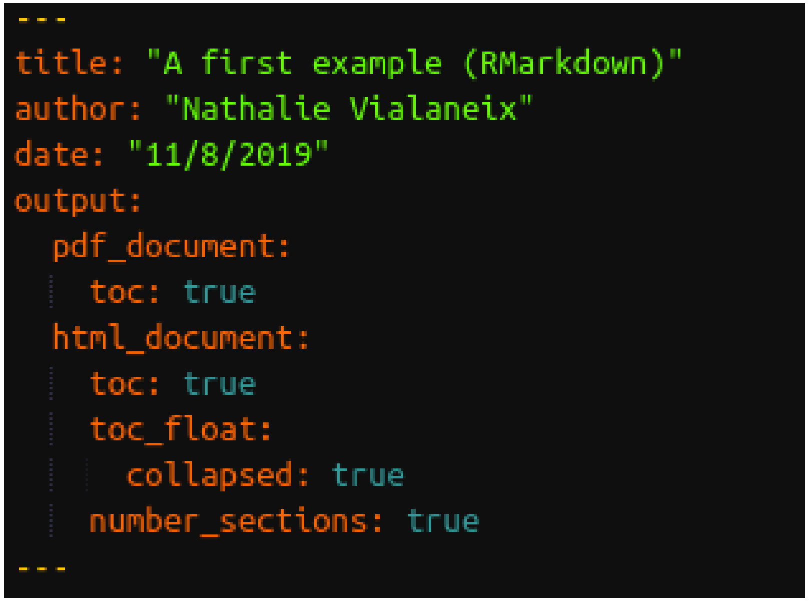
```
##       speed             dist
```



1 Main section
2 Section section

2 Section section

Yaml headers



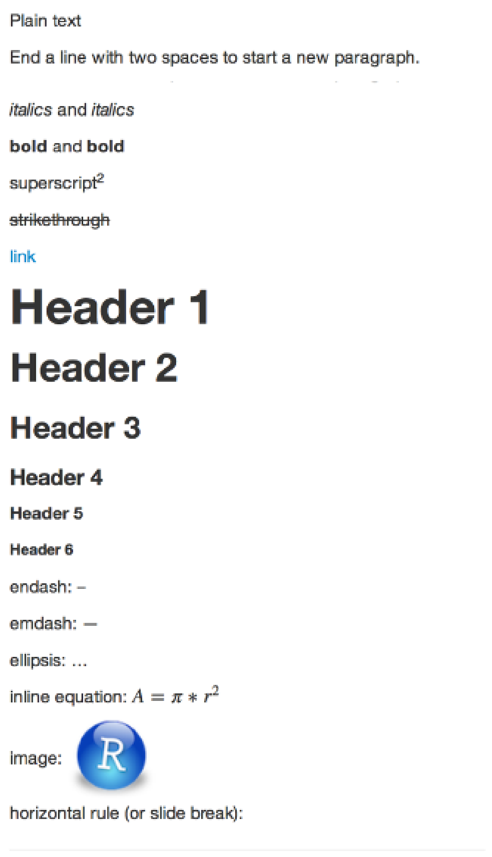
```
---
title: "A first example (RMarkdown)"
author: "Nathalie Vialaneix"
date: "11/8/2019"
output:
  pdf_document:
    toc: true
  html_document:
    toc: true
    toc_float:
      collapsed: true
    number_sections: true
---
```

Text formatting

Syntax

```
Plain text

End a line with two spaces
to start a new paragraph.

*italics* and _italics_

**bold** and __bold__

superscript^2^

~~strikethrough~~

[link](www.rstudio.com)

# Header 1

## Header 2

### Header 3

#### Header 4

##### Header 5

###### Header 6

endash: --

emdash: ---

ellipsis: ...

inline equation: $A = \pi*r^{2}$

image: ![](path/to/smallorb.png)

horizontal rule (or slide break):

***
```

Becomes



Plain text

End a line with two spaces to start a new paragraph.

italics and *italics*

bold and **bold**

superscript^{2}

~~strikethrough~~

link

Header 1

Header 2

Header 3

Header 4

Header 5

Header 6

endash: –

emdash: —

ellipsis: …

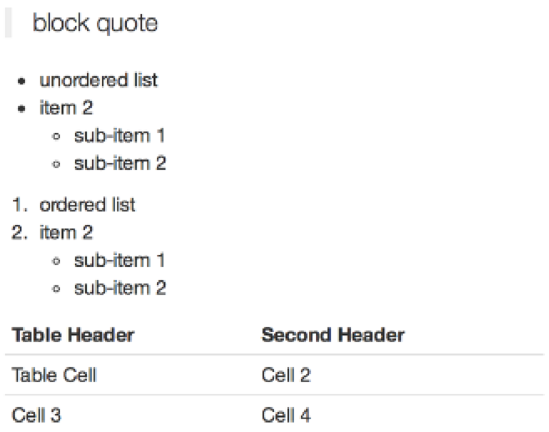
inline equation: $A = \pi * r^{2}$

image:

horizontal rule (or slide break):

Text formatting

```
> block quote

* unordered list
* item 2
    + sub-item 1
    + sub-item 2

1. ordered list
2. item 2
    + sub-item 1
    + sub-item 2

Table Header  | Second Header
------------- | -------------
Table Cell    | Cell 2
Cell 3        | Cell 4
```



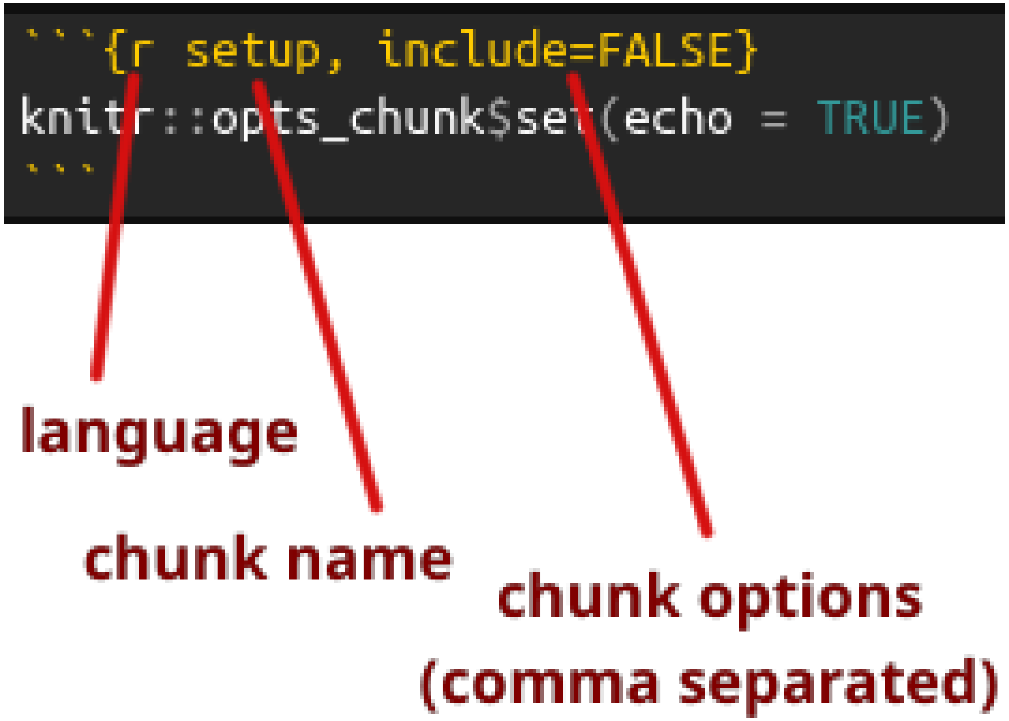
> block quote

- unordered list
- item 2
  - sub-item 1
  - sub-item 2

1. ordered list
2. item 2
   - sub-item 1
   - sub-item 2

Table Header	**Second Header**
Table Cell	Cell 2
Cell 3	Cell 4

Code chunk and options



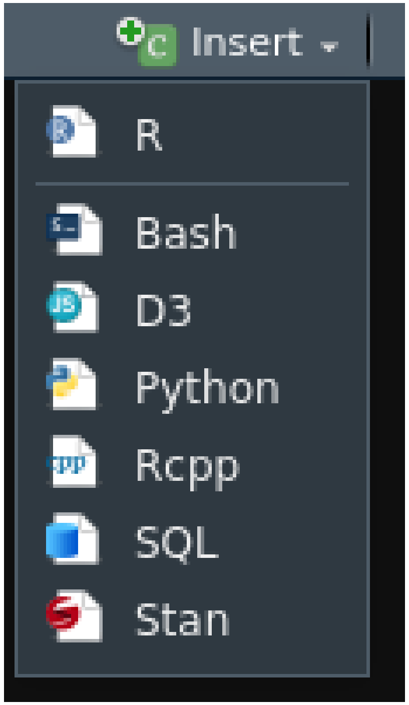
~~~
```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```
~~~

Inline code can be included with

```
`r params$date`
```

Code chunk and options

Types of chunks:



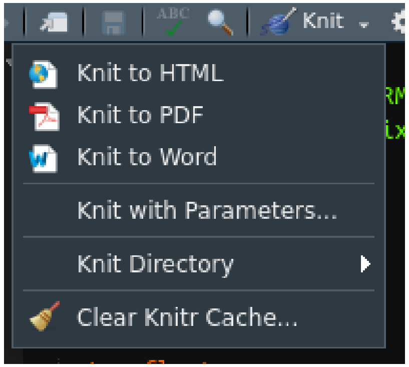
Code chunk and options

A few useful options:

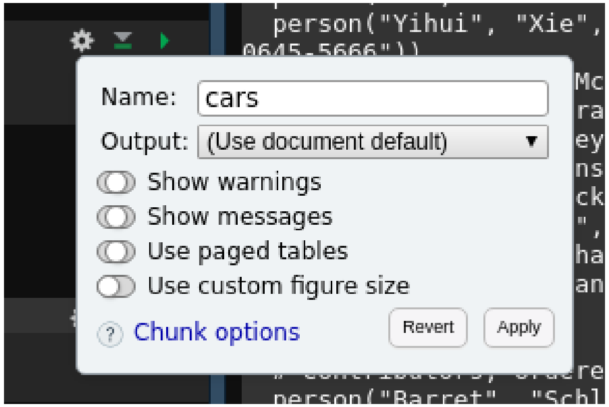
- `eval=TRUE/FALSE`: run (or not) the code in the chunk
- `include=TRUE/FALSE`: include (or not) the chunk in the final document
- `echo=TRUE/FALSE`: include (or not) the output of the code in the final document
- `results='markup'/'hide'/'asis'/'hold'`: display the output of the code normally/not/as it is (useful when combined with `xtable` for HTML outputs of tables for instance)/all in a row after the code chunk is displayed (and not one by one)
- `error/message/warning=TRUE/FALSE`: display (or not) the different types of messages obtained from the code
- `cache=TRUE/FALSE`: cache the result of the chunk
- `fig.width/height/align/cap/...`: different options on rendering of figures

Run code and output types

Run the whole document:

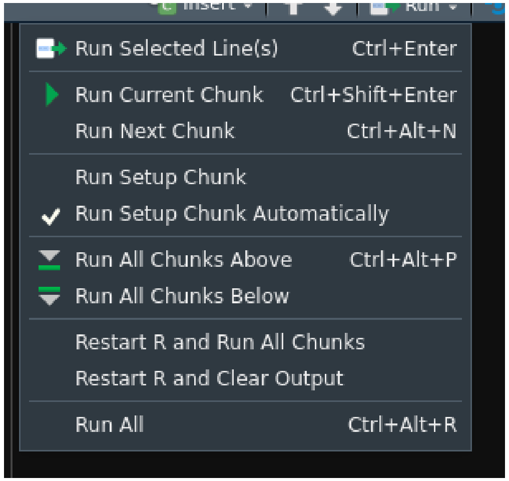


Run interactively:



Name: cars

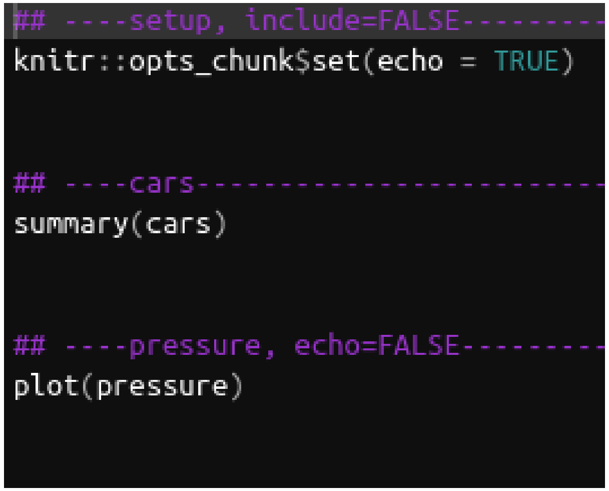
Output: (Use document default)


Run Selected Line(s) Ctrl+Enter
Run Current Chunk Ctrl+Shift+Enter
Run Next Chunk Ctrl+Alt+N
Run Setup Chunk
✓ Run Setup Chunk Automatically
Run All Chunks Above Ctrl+Alt+P
Run All Chunks Below
Restart R and Run All Chunks
Restart R and Clear Output
Run All Ctrl+Alt+R

Run code and output types

Run the whole document externally:

```
# render document
knitr::render("mybeautifuldoc.Rmd", output_format = "HTML",
              encoding = "UTF-8")
# extract R code from chunk
knitr::purl("mybeautifuldoc.Rmd")
```



```
## ----setup, include=FALSE--------
knitr::opts_chunk$set(echo = TRUE)


## ----cars-------------------------
summary(cars)


## ----pressure, echo=FALSE--------
plot(pressure)
```

Useful to run a report with computationally intensive code on a server! See: http://www.nathalievialaneix.eu/doc/pdf/tutoR_cluster.pdf

Run code and output types

- HTML: often light and can include interactive graphics (with `ggplotly` for instance)
    - HTML and LaTeX equations can be included directly in the document
    - **But**: problems rendering equations without internet connexion, you need to copy several folders in addition to the file to make it work
    - problems partially solved with the option: `self_contained: true` (in HTML headers)
- PDF: self-contained but often heavier, needs LaTeX installed
    - usually longer to knit (pandoc produces a TeX document and then LaTeX is run)
    - can account for LaTeX options in the headers (including bibliography)
    - can use the power of LaTeX to obtain multiple version documents (student and teacher versions for instance)

(and, of course, **forget Word**!)

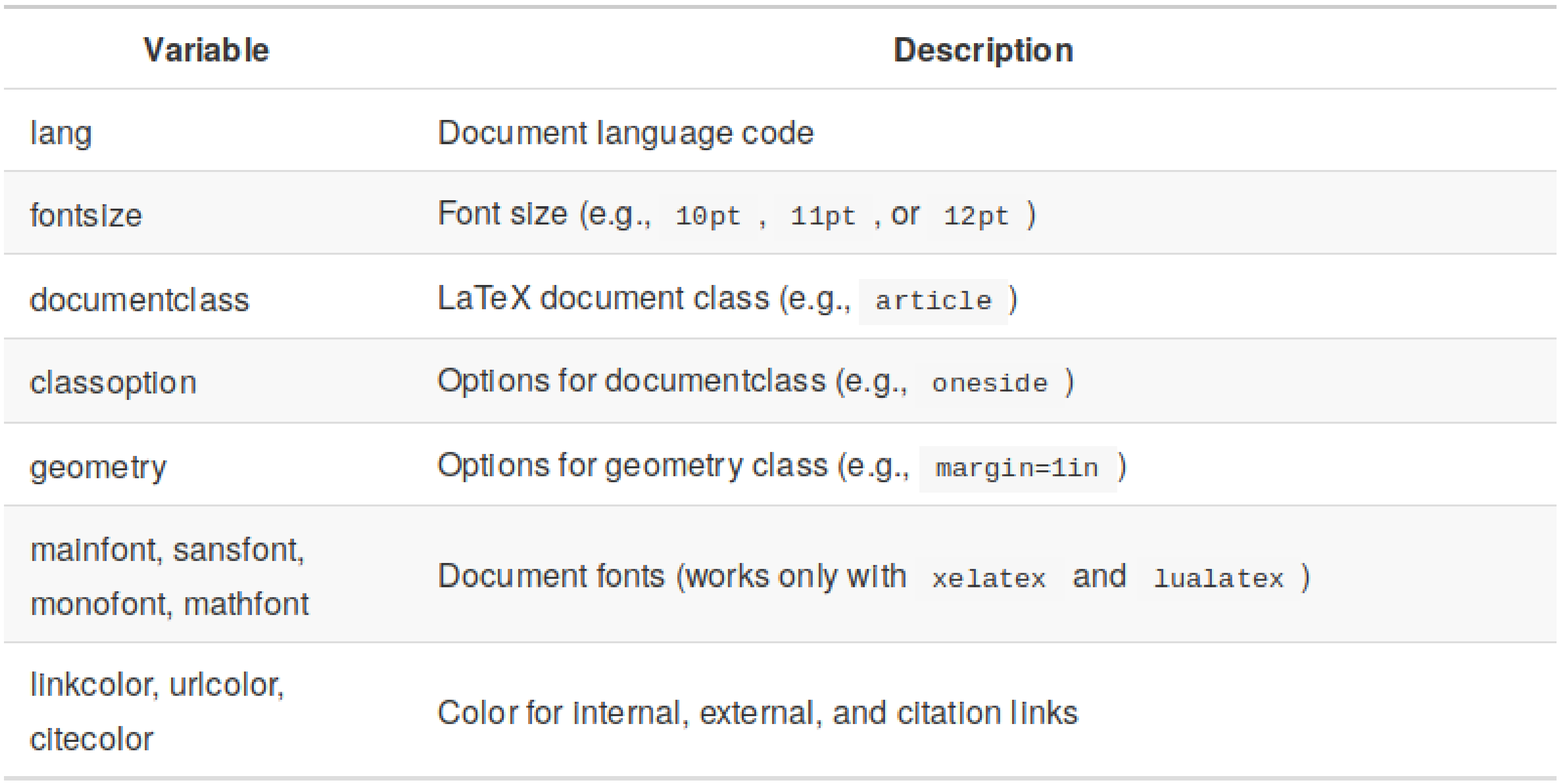
Advanced options for RMarkdown documents

Appearance and style in HTML documents

- `theme` (in header) specifies the Bootswatch theme to use
- `highlight` (in header) specifies the highlighting style to use (supports `tango`, `pygments`, `kate`, `monochrome`, `espresso`, `zenburn`, `haddock`, `breezedark` and `textmate`)

Or: you can use your own CSS with `css` (in header)

$\LaTeX$ options in PDF documents



Variable	Description
lang	Document language code
fontsize	Font size (e.g., `10pt` , `11pt` , or `12pt`)
documentclass	LaTeX document class (e.g., `article`)
classoption	Options for documentclass (e.g., `oneside`)
geometry	Options for geometry class (e.g., `margin=1in`)
mainfont, sansfont, monofont, mathfont	Document fonts (works only with `xelatex` and `lualatex`)
linkcolor, urlcolor, citecolor	Color for internal, external, and citation links

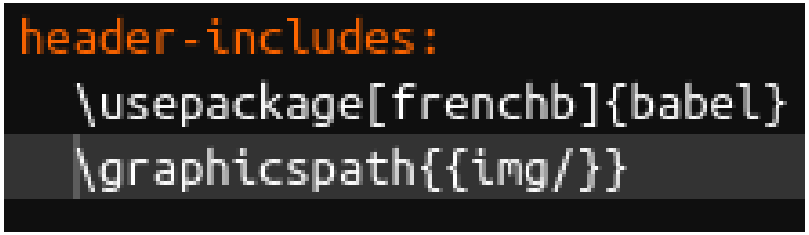
LaTeX options in PDF documents

Bibliography is managed using:

- the citation engine (default is `pandoc-citeproc`):
    - `citation_package: natbib` (header, in `pdf_document`)
    - `citation_package: biblatex` (header, in `pdf_document`)
- the bibtex file: `bibliograph: mybib.bib` (header, in `pdf_document`)

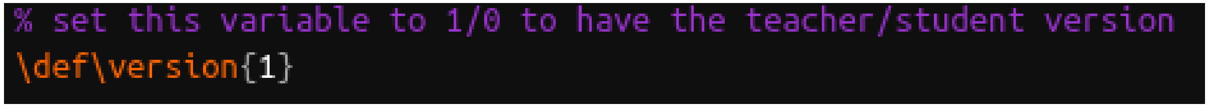
LaTeX options in PDF documents

More custom options in headers:



```
header-includes:
    \usepackage[frenchb]{babel}
    \graphicspath{{img/}}
```

LaTeX multiple version document

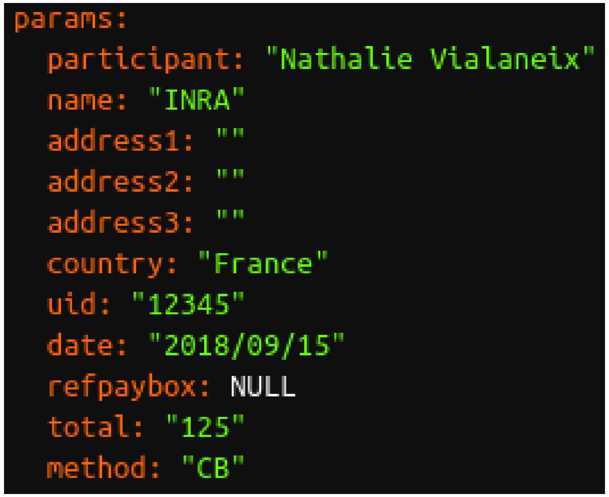


```
% set this variable to 1/0 to have the teacher/student version
\def\version{1}
```

```
\if \version1 \\
... in teacher version only
\fi
```

A **very** cool stuff: parameters!

How were useR! invoices generated (and sent)?



```
params:
  participant: "Nathalie Vialaneix"
  name: "INRA"
  address1: ""
  address2: ""
  address3: ""
  country: "France"
  uid: "12345"
  date: "2018/09/15"
  refpaybox: NULL
  total: "125"
  method: "CB"
```

and then, used in the Rmd file through `params$participant`, `params$name`, ...

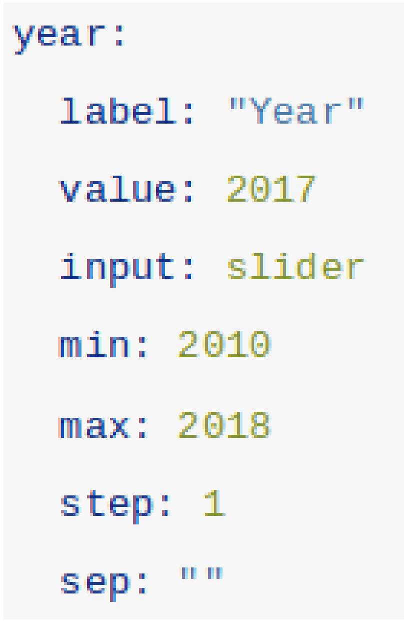
A **very** cool stuff: parameters!

... or passed externally using `render`:

```
## loop over users ##############################################
render("invoice-base.Rmd",
       params = list(participant = paste(users$FIRSTNAME[cur], usersS
                     name = users$"Billing.name"[cur],
                     address1 = users$"Billing.address..street."[cur]
                     address2 = users$"Billing.address..other."[cur],
                     address3 = users$"Billing.address..state..town.a
                     country = users$"Billing.address..country."[cur]
                     uid = users$UID[cur],
                     date = gsub("-", "/", as.character(as.Date(savec
                     total = users$Amount[cur],
                     method = users$Payment.type[cur],
                     refpaybox = users$transaction[cur]
       ),
       output_file = paste0("invoices/", file_names[cur]))

# send email
send.mail(from = "contact@user2019.fr",
          to = ...,
          attach.files = paste0("invoices/", file_names[cur]))
```

A **very** cool stuff: parameters!

- used to produce automatic report on standard analyses (*e.g.*, differential analysis of RNAseq data with only two conditions)
- used to run standard analyses on multiple datasets at once
- parameters can be set interactively using the knit buttom
- constrains can be declared for parameters



```
year:
  label: "Year"
  value: 2017
  input: slider
  min: 2010
  max: 2018
  step: 1
  sep: ""
```

Want to know more?

- **R** Markdown cheatsheet
- **R** Markdown reference
- **R** Markdown book

Other types of documents

Slides

Using the same approach, you can make:

- **R** presentation (included in **R** studio) but with rather limited features

(my old class on **R** was made using it)

- `xarigan` slides (HTML), also developed by Yihui Xie (see Chapter 7 of the **R** Markdown book): easy to use and includes many options but if you want to obtain a custom result, you better be an HTML/CSS ninja, can be exported in PDF with `pagedown::chrome_print` (but I do not recommend it)

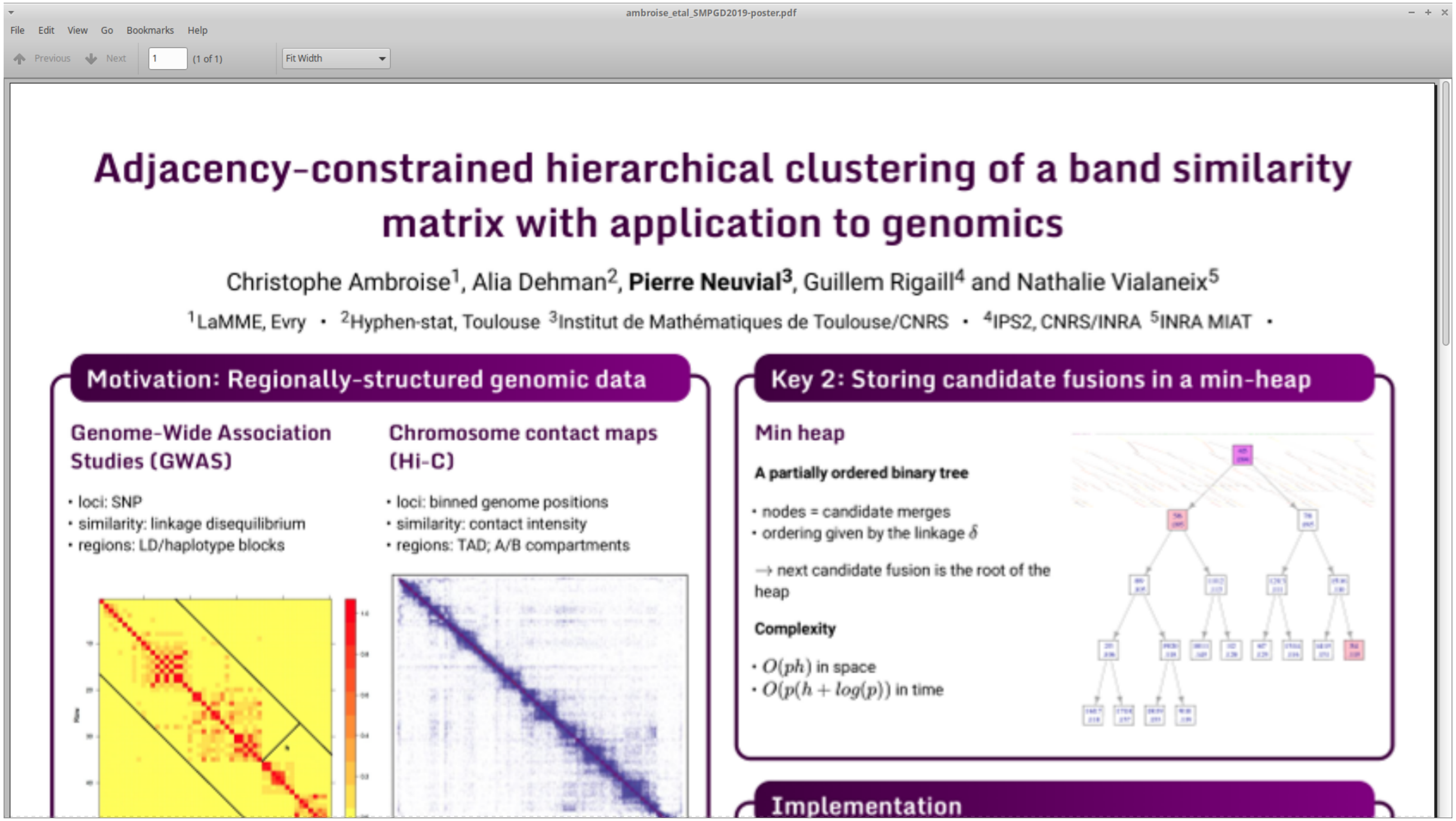
this presentation was made with xaringan using the `rladies` css

- `binb` (binb is not beamer) provides functionality to use themes for beamer directly in RMarkdown: if you want to obtain a custom result, you better be a $\LaTeX$ ninja, output PDF, long to compile (as beamer is)

(this seminar has been made using binb)

Posters

drposter (example is courtesy of Pierre Neuvial)


ambroise_etal_SMPGD2019-poster.pdf
File Edit View Go Bookmarks Help
Previous Next 1 (1 of 1) Fit Width

Adjacency-constrained hierarchical clustering of a band similarity matrix with application to genomics

Christophe Ambroise[1], Alia Dehman[2], **Pierre Neuvial[3]**, Guillem Rigaill[4] and Nathalie Vialaneix[5]

[1]LaMME, Evry • [2]Hyphen-stat, Toulouse [3]Institut de Mathématiques de Toulouse/CNRS • [4]IPS2, CNRS/INRA [5]INRA MIAT •

Motivation: Regionally-structured genomic data

Genome-Wide Association Studies (GWAS)

- loci: SNP
- similarity: linkage disequilibrium
- regions: LD/haplotype blocks

Chromosome contact maps (Hi-C)

- loci: binned genome positions
- similarity: contact intensity
- regions: TAD; A/B compartments

Key 2: Storing candidate fusions in a min-heap

Min heap

A partially ordered binary tree

- nodes = candidate merges
- ordering given by the linkage δ

$\rightarrow$ next candidate fusion is the root of the heap

Complexity

- $O(ph)$ in space
- $O(p(h + log(p)))$ in time

Implementation

Documentation for packages

`pkgdown`: Generate cool package documentation from your package

Example from adjclust

Books

`bookdown`: generate printer ready books and ebooks from **R** Markdown

I never used it so no feedback...

Website

`blogdown`: generate website from RMarkdown pages using hugo (python static website generator).

The missing data reference website is generated through `blogdown` and thanks to its hugo support is automatically published on netlify.

Next to come...?

`coffeedown`... ?