

R Markdown

Ivy Jansen, Pieter Vershelde



Wat is R Markdown?

Een *tekstverwerker* voor het maken van reproduceerbare en dynamische documenten met R

- Geschreven in markdown (een eenvoudig te schrijven platte tekstindeling)
- Bevat stukjes ingesloten R-code
- Er is een *compilatie* stap nodig om het finale document te krijgen

Waarom R Markdown gebruiken?

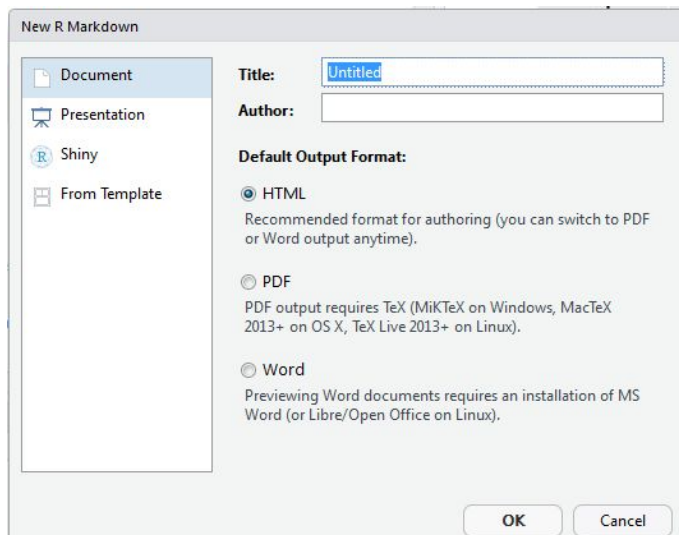
- Snel en reproduceerbaar rapport van je R analyses
- Code en output in eenzelfde document, zonder manuele copy-paste stappen
- Eenvoudig tekst toe te voegen en structuur aan te brengen met koppen, lijsten, ...
 - Snel te leren
 - Geen handmatige formattering nodig
 - Nog steeds goed leesbaar
- Verschillende output formaten mogelijk, zoals .html, .pdf, ...
- Bij wijzigingen in je data krijg je een volledig up-to-date document door 1 simpele “klik op de knop”

R Markdown basics

Een R Markdown file is een platte tekstfile met extensie `.Rmd` en bevat 3 belangrijke onderdelen:

1. Een **YAML header**
2. **Chunks** met R code
3. Markdown tekst

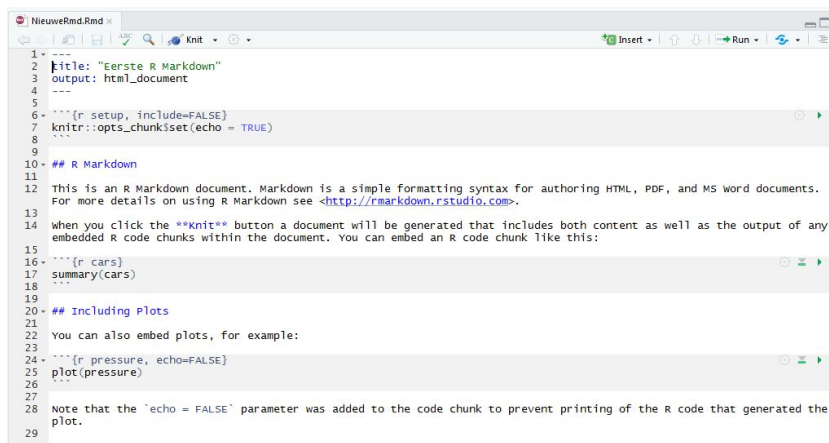
Een nieuwe R Markdown file aanmaken via **File -> New File -> R Markdown**



Hier kan je al kiezen uit allerlei opties:

- Document met output formats
 - HTML
 - PDF
 - Word
- Presentation met output formats
 - HTML (ioslides)
 - HTML (Slidy)
 - PDF (Beamer)
- Shiny
- From Template
 - INBO huisstijl

Kies Document, en geef het een titel. De nieuwe file ziet er dan als volgt uit:



```
1 ---
2 title: "Eerste R Markdown"
3 output: html_document
4 ---
5
6 ```{r setup, include=FALSE}
7 knitr::opts_chunk$set(echo = TRUE)
8 ```
9
10 ## R Markdown
11
12 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS word documents.
13 For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
14
15 When you click the "Knit" button a document will be generated that includes both content as well as the output of any
16 embedded R code chunks within the document. You can embed an R code chunk like this:
17
18 ```{r cars}
19 summary(cars)
20 ```
21
22 ## Including Plots
23
24 You can also embed plots, for example:
25
26 ```{r pressure, echo=FALSE}
27 plot(pressure)
28 ```
29
30 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the
31 plot.
```

Om een volledig rapport te maken dat alle tekst, code én resultaten bevat

- Klik op de Knit knop



- Of gebruik de *shortcut* CTRL + SHIFT + K

Wanneer je een document **Knit** (= breien), dan *breit* R Markdown alle onderdelen aan mekaar als volgt

- De .Rmd file wordt door **knitr** omgezet naar een markdown (.md) document
 - Alle code chunks worden uitgevoerd
 - Dit bevat alle tekst, code en de output
- Het .md document wordt verwerkt door **pandoc** tot een finaal document
 - Hiervoor zijn veel output formaten mogelijk



Markdown syntax

Een beperkte set van conventies om platte tekst te formatteren, die eenvoudig te leren en te schrijven zijn, en de tekst zelf nog gemakkelijk leesbaar houdt.

Tekst formatting

- *Italic*: 1 sterretje of underscore rond de tekst
`*italic*` or `_italic_`
- **Bold**: 2 sterretjes of underscores rond de tekst
`**bold**` or `__bold__`
- Code (tekst met vaste breedte): accent grave (`) rond de tekst
``code``
- Superscript²: accent circonflex (^) rond de tekst
`superscript^2^`
- Subscript₂: tilde (~) rond de tekst
`subscript~2~`

Koppen

- Kop 1: 1 hashtag
`# 1st Level Header`
- Kop 2: 2 hashtags
`## 2nd Level Header`
- Kop 3: 3 hashtags
`### 3rd Level Header`
- ...

Lijsten

- Lijst met opsommingstekens
 - * Bulleted list item 1
 - * Item 2
 - Item 2a
 - Item 2b
 - Met sterretjes of streepjes
 - Gebruik eenzelfde symbool voor de verschillende niveaus
 - In het gerenderde document krijgt elk niveau een ander opsommingsteken
- Genummerde lijst
 - 1. Numbered list item 1
 - 1. Item 2.
 - Een (willekeurig) cijfer of letter gevolgd door een .
 - In het gerenderde document verhogen de cijfers (letters) automatisch
 - Bij diepere niveaus start de nummering opnieuw
- Er mogen zelfs blanco regels tussen gelaten worden
- Voor het eerste item **moet** een blanco regel
- Diepere levels worden voorafgegaan door 4 spaties (of 2 tabs, als RStudio zo ingesteld is)

Links en figuren

- Link onder de url <http://www.inbo.be> zelf
`<http://www.inbo.be>`
- Link onder [een stuk tekst](#)
`[een stuk tekst](http://www.inbo.be)`
- Figuur met optionele titel, afmetingen kunnen gespecificeerd worden (ZONDER spaties rond =)
`![Optionele titel](../Figuren/logo_inbo_0.jpg){width=4cm, height=2cm}`

INSTITUUT
NATUUR- EN
BOSONDERZOEK

Code chunks

Om de R code in een R Markdown document te laten uitvoeren, moet die in een **chunk** staan. Er zijn 3 manieren om een chunk aan te maken:



1. *Shortcut* CTRL + ALT + I
2. De **Insert** knop in de balk bovenaan → R
3. Door manueel de chunk delimiters ````${r}```` en ````` in te tikken

Het resultaat is telkens

```
```${r}```
```

```
```
```

Tussen deze chunk delimiters schrijf je dan (kleine) stukken R code, die je nog steeds kan uitvoeren zoals in een R script. In het **Run** menu bovenaan kan je verschillende mogelijkheden kiezen, of m.b.v. onderstaande shortcuts:

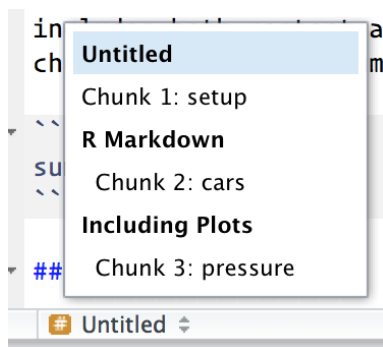
- Regel per regel: CTRL + ENTER
- De huidige chunk: CTRL + SHIFT + ENTER, of met de groene *Play*-pijl 
- Alle voorgaande chunks: CTRL + ALT + P, of met de *pijl tot het groene streepje* 
- De volgende chunk: CTRL + ALT + N
- Alle chunks: CTRL + ALT + R

Binnen de accolades (op de eerste regel van de chunk, na de `r`) kunnen nog een hele reeks opties meegegeven worden.

Chunk naam

Er kan een **unieke** naam gegeven worden aan elke chunk (**vermijd spaties, punten en underscores**). Deze volgt na de `r` en een spatie: ````${r} chunknaam``. Dit heeft 3 voordelen:


1. Je kan gemakkelijker navigeren tussen de verschillende chunks, door gebruik te maken van de drop-down navigator links onder in de script editor.



2. Figuren die in een chunk aangemaakt worden, krijgen een logische naam (gebaseerd op de chunk naam), zodat deze gemakkelijker te hergebruiken zijn.
3. Je kan verwijzen naar chunks.

Chunk opties

In de chunk header kunnen nog een hele reeks extra opties meegegeven worden:

- Na de chunk naam
- Gescheiden door komma's
- Allemaal achter mekaar, geen regeleinde toegelaten
- Kunnen ook gekozen worden via het *Instellingen wiel*

Getoonde onderdelen

De belangrijkste opties controleren of de code uitgevoerd moet worden en welke resultaten er getoond worden in het finale document.

- **eval** = FALSE: code wordt getoond maar niet uitgevoerd (en dus ook geen output). Dit is handig om voorbeeldcode te laten zien, of om een groot blok code uit te schakelen zonder de regels uit te commentariëren.
- **include** = FALSE: code wordt uitgevoerd maar niet getoond. Ook geen output getoond. Gebruik dit voor setup code (laden van packages e.d.) die je niet wil tonen in je rapport.
- **echo** = FALSE: Code wordt niet getoond maar wel uitgevoerd, en de output wordt wel getoond. Zinvol voor rapporten waar de lezer de onderliggende code niet moet zien.
- **results** = 'hide': verbergt de output.
results = 'hold': output wordt pas getoond nadat alle code in die chunk uitgevoerd is.
- **fig.show** = 'hide' verbergt de figuren.
- **message** = FALSE: er worden geen messages getoond.
- **warning** = FALSE: er worden geen warnings getoond.
- **error** = TRUE: het compileren gaat gewoon door, ondanks een fout in de code. De foutmelding wordt getoond in het rapport. Kan handig zijn om te debuggen. Default, **error** = FALSE, laat het knitten mislukken als er een fout in het document zit.

Onderstaande tabel laat zien welke onderdelen niet getoond worden door de verschillende opties:

| Option | Run code | Show code | Output | Plots | Messages | Warnings |
|--------------------------|----------|-----------|--------|-------|----------|----------|
| eval = FALSE | - | | - | - | - | - |
| include = FALSE | | - | - | - | - | - |
| echo = FALSE | | - | | | | |
| results = "hide" | | | - | | | |
| fig.show = "hide" | | | | - | | |
| message = FALSE | | | | | - | |
| warning = FALSE | | | | | | - |

Figuren

Opties hoe figuren die met R gemaakt zijn (bvb ggplot) getoond moeten worden

- `fig.width = 3`: breedte van de figuur in inch
 - Wil je een breedte van 10cm, specificeer dan `fig.width = 10/2.54`, want 1 inch = 2.54 cm
- `fig.height = 2`: hoogte van de figuur in inch
- `fig.align = "left", "right", "center"`: alignering van de figuur
- `fig.cap = ""`: titel van de figuur (character string)

Tip: Als je gebruik wil maken van de figure caption, moet je elke figuur in een aparte chunk zetten. Op deze manier kan je dan ook cross-references toevoegen naar de figuur.

Caching

Normaal begint elke compilatie van een R Markdown document met een schone lei. Wanneer er echter zware berekeningen gedaan worden, kan het interessant zijn deze resultaten te bewaren, zodat deze chunks niet telkens opnieuw uitgevoerd moeten worden. Daarvoor dienen de volgende opties:

- `cache = TRUE`: output van de chunk wordt bewaard onder een speciale naam. In volgende runs zal knitr controleren of de **code** veranderd is, en enkel dan de chunk opnieuw uitvoeren, anders de bewaarde resultaten gebruiken.
- `cache.path = ""`: locatie waar de cache-files bewaard moeten worden, indien verschillend van de default.

Er moet wel zorgvuldig omgesprongen worden met het cachen, want er wordt enkel naar de code van de desbetreffende chunk gekeken, niet naar de dependencies (bvb eerdere chunks waar data ingelezen en/of bewerkt worden). Dit kan vermeden worden door deze afhankelijkheden mee te nemen als chunk optie:

- `dependson = ""`: character vector met **alle** chunks waarvan deze chunk afhankelijk is

of de volgende globale optie te specificeren:

- `autodep = TRUE`: achterhaal de afhankelijkheden tussen chunks automatisch door de globale variabelen in de code te analyseren

Toch is het goed om regelmatig de cache volledig te verwijderen met het commando `knitr::clean_cache()` (in de console).

Globale opties

Sommige van de eerder gedefinieerde opties wil je graag hanteren voor alle chunks. Dit kan door in het begin van het document een chunk aan te maken met de code `knitr::opts_chunk$set()` en deze opties tussen de haakjes vast te leggen, gescheiden door komma's.

```
knitr::opts_chunk$set(  
  echo = FALSE,  
  message = FALSE,  
  warning = FALSE,  
  cache = TRUE,  
  cache.path = "cache/",  
  fig.height = 6,  
  fig.width = 9  
)
```

Nu worden voor het hele document de code, messages en warnings verborgen, de resultaten gecached in de subfolder `cache` (die moet bestaan), en zijn alle figuren 6 inch hoog en 9 inch breed.

Inline code

De mogelijkheid bestaat om output van R code ook rechtstreeks in de tekst te integreren. Dit kan met de code ``r ``. Dit kan handig zijn als je bepaalde eigenschappen van je gegevens (bvb aantal records, gemiddelde, *p*-waarde, ...) wil vermelden in de tekst.

De dataset `iris` bevat gegevens over ``r nrow(iris)`` bloemen. Voor het bloemblad is de gemiddelde lengte ``r mean(iris$Petal.Length)`` cm en de gemiddelde breedte ``r mean(iris$Petal.Width)`` cm.

Bij het compileren van het document worden de resultaten van de berekeningen in de tekst geïntegreerd:

De dataset `iris` bevat gegevens over 150 bloemen. Voor het bloemblad is de gemiddelde lengte 3.758cm en de gemiddelde breedte 1.199cm.

YAML header

Optionele sectie die bovenaan het document komt, omringd door `---`.

Standaard

```
---
title: "Les 3 - R Markdown"
output: html_document
---
```

Extra opties

- `subtitle`: "Een propere manier om R code en tekst te combineren"
- `author`: "Ivy Jansen, Pieter Verschelde"
- `date`: 2019-03-19
- `papersize`: a4: default is letter
- `urlcolor`: blue: zorgt ervoor dat de links blauw kleuren
- `bibliography`: `rmarkdown.bib`: bibliography file, BiBTeX, Endnote, medline,...
- `csl`: `apa.csl`: citation style language

Specifieke opties voor de output

- `toc`: `true`: inhoudstafel toevoegen
- `toc_depth`: 2: tem kop 2 in de inhoudstafel
- `number_sections`: `true`: secties nummeren

Deze opties kunnen gebruikt worden bij alle output formaten. Andere opties zijn specifiek voor de verschillende output formaten, en hiervoor verwijzen we naar <https://bookdown.org/yihui/rmarkdown/documents.html>. Let erop dat het output formaat nu gevolgd wordt door een dubbelpunt, en dat de opties voorafgegaan worden door 2 extra spaties.

```
---
title: "Les 3 - R Markdown"
output:
  html_document:
    toc: true
    toc_depth: 2
    number_sections: true
---
```

Tabellen

Standaard print R Markdown een tabel zoals je die in de console ziet (met een dubbele hashtag ervoor).

```
iris[1:10,]
```

| ## | Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
|------|--------------|-------------|--------------|-------------|---------|
| ## 1 | 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| ## 2 | 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| ## 3 | 4.7 | 3.2 | 1.3 | 0.2 | setosa |

```
## 4      4.6      3.1      1.5      0.2 setosa
## 5      5.0      3.6      1.4      0.2 setosa
## 6      5.4      3.9      1.7      0.4 setosa
## 7      4.6      3.4      1.4      0.3 setosa
## 8      5.0      3.4      1.5      0.2 setosa
## 9      4.4      2.9      1.4      0.2 setosa
## 10     4.9      3.1      1.5      0.1 setosa
```

Er bestaan een hele reeks packages en functies om dit mooier te tonen. De eenvoudigste manier is met de functie `kable` uit het package `knitr`.

```
knitr::kable(iris[1:10,])
```

| Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
|--------------|-------------|--------------|-------------|---------|
| 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| 5.0 | 3.6 | 1.4 | 0.2 | setosa |
| 5.4 | 3.9 | 1.7 | 0.4 | setosa |
| 4.6 | 3.4 | 1.4 | 0.3 | setosa |
| 5.0 | 3.4 | 1.5 | 0.2 | setosa |
| 4.4 | 2.9 | 1.4 | 0.2 | setosa |
| 4.9 | 3.1 | 1.5 | 0.1 | setosa |

Kijk zeker eens naar de help welke mogelijkheden er allemaal zijn om de tabel verder te formatteren (rij- en kolomnamen, titel, alignering, digits, ...).

Tip: Als je een `caption` toevoegt, kan je ook hier gebruik maken van cross-references voor tabellen.

Shortcuts

- Accent grave (`): ALT + 96
- Document knitten: CTRL + SHIFT + K
- Nieuwe chunk: CTRL + ALT + I
- Code regel per regel uitvoeren: CTRL + ENTER
- Volledige code chunk uitvoeren: CTRL + SHIFT + ENTER
- Alle voorgaande chunks uitvoeren: CTRL + ALT + P
- De volgende chunk uitvoeren: CTRL + ALT + N
- Alle chunks uitvoeren: CTRL + ALT + R

More to read

- [Introduction to R Markdown](#)
- R for data science
 - Boek van Hadley Wickham en Garrett Grolemond
 - Hardcopy beschikbaar op INBO
 - [Digitale versie](#)
- Cheat Sheets
 - In RStudio onder Help menu
 - [Online](#)

References

- [R for data science](#)
- [Coding Club Edinburgh](#)