Reproducible Research with R

Brooke Anderson

1/25/2021

Overview

Aims for lecture

1. What are "knitted" documents?
2. How to create these with R
3. What is data pre-processing?
4. Creating reproducible data pre-processing protocols for your research

Running example

As a running example, we will use data preprocessing with the `xcms` package, available on Bioconductor.

Package description:

> *"Framework for processing and visualization of chromatographically separated and single-spectra* ***mass spectral data****. Imports from AIA/ANDI NetCDF, mzXML, mzData and mzML files.* ***Preprocesses data*** *for high-throughput, untargeted* ***analyte profiling****."*

Knitted documents

You already use knitted documents!

You have likely already seen and used examples of **knitted documents**.

Many tutorials for R or Python packages are written as knitted documents. For example, here's part of the `xcms` vignette:

3 Initial data inspection

The `OnDiskMSnExp` organizes the MS data by spectrum and provides the methods `intensity`, `mz` and `rtime` to access the raw data from the files (the measured intensity values, the corresponding m/z and retention time values). In addition, the `spectra` method could be used to return all data encapsulated in `Spectrum` objects. Below we extract the retention time values from the object.

```
head(rtime(raw_data))
```

```
## F1.S0001 F1.S0002 F1.S0003 F1.S0004 F1.S0005 F1.S0006
## 2501.378 2502.943 2504.508 2506.073 2507.638 2509.203
```

Definition of knitted documents

The defining characteristic of a knitted document is that it interweaves two elements:

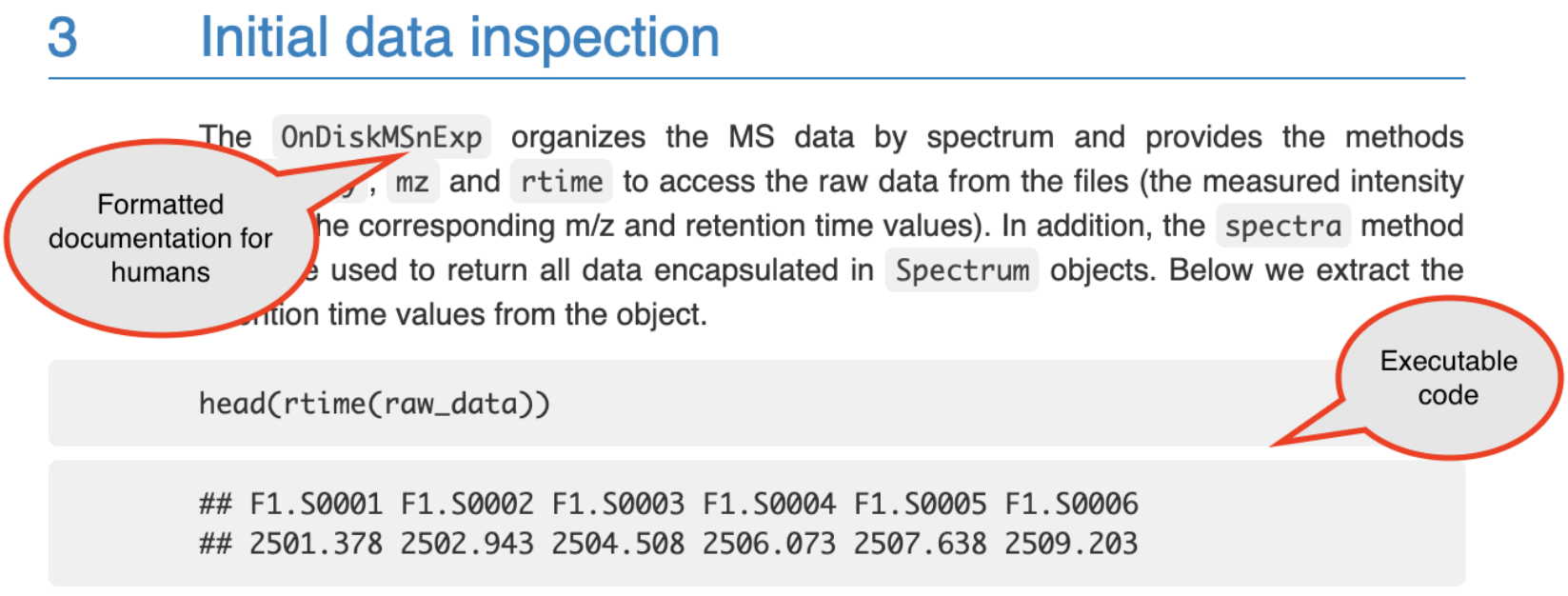
1. Executable code
2. Formatted documentation meant for humans

Definition of knitted documents

The defining characteristic of a knitted document is that it interweaves two elements:

1. Executable code
2. Formatted documentation meant for humans

Example:



How knitted documents work

1. Knitted documents start as plain text
2. A special section at the start of the document (**preamble**) gives some overall directions about the document
3. Special combinations of characters indicate where the executable code starts
4. Other special combinations show where the regular text starts (and the executable code section ends)
5. Formatting for the rest of the document is specified with a **markup language**
6. You create the final document by **rendering** the plain text document. This process runs through two software programs.
7. The final document is attractive and **read-only**—you should never make edits to this output, only to your initial plain text document.

How knitted documents work

1. Knitted documents start as plain text

[Example of what the plain text looks like]

How knitted documents work

Writing plain text:

- Use a text editor (*not* Word or similar word processing programs)
- Only use ASCII [?]
- What file extension to use when you save the file?

How knitted documents work

2. A special section at the start of the document (**preamble**) gives some overall directions about the document

In RMarkdown documents, this preamble is specified using **YAML**.

[Example of what YAML looks like]

How knitted documents work

[Where to find more about what you can put in the YAML or other "details" part of the document—preamble]

[There are other types of knitted documents, too—they might use other languages for the preamble and the markup.]

How knitted documents work

3. Special combinations of characters indicate where the executable code starts
4. Other special combinations show where the regular text starts (and the executable code section ends)

[Example of a code chunk]

How knitted documents work

This combination indicates the start of executable code:

```{r}
```

How knitted documents work

This combination indicates the start of executable code:

```{r}
```

This combination indicates the start of regular documentation (that is, the end of executable code):

```
```

How knitted documents work

This combination indicates the start of executable code:

```{r}
```

This combination indicates the start of regular documentation (that is, the end of executable code):

```
```

In the starting combination, you can also add some specifications for how you want the code run and showed:

```{r echo = FALSE, fig.align = "center"}
```

How knitted documents work

5. Formatting for the rest of the document is specified with a **markup language**

You do not have buttons to click for formatting like bold, italics, font size, and so on. Instead, you use **special characters or character combinations** to specify formatting in the final document.

For example, you'll surround a word or phrase in ** to make it bold.

To write "**this**" in the final document, you'll write `"**this**"` in the plain text initial document.

How knitted documents work

The start of this document:

3 Initial data inspection

The `OnDiskMSnExp` organizes the MS data by spectrum and provides the methods `intensity`, `mz` and `rtime` to access the raw data from the files (the measured intensity values, the corresponding m/z and retention time values). In addition, the `spectra` method could be used to return all data encapsulated in `Spectrum` objects. Below we extract the retention time values from the object.

```
head(rtime(raw_data))
```

```
## F1.S0001 F1.S0002 F1.S0003 F1.S0004 F1.S0005 F1.S0006
## 2501.378 2502.943 2504.508 2506.073 2507.638 2509.203
```

Is written like this:

```
# Initial data inspection

The `OnDiskMSExp` organizes the MS data ...
```

How knitted documents work

6. You create the final document by **rendering** the plain text document. This process runs through two software programs.
7. The final document is attractive and **read-only**—you should never make edits to this output, only to your initial plain text document.

Why use knitted documents?

1. Code is checked every time you render the document
2. Code is formatted without special symbols
3. Code can be re-run with updated or new datasets
4. Document is in plain text, so it can be tracked well with version control

Why use knitted documents?

1. Code is checked every time you render the document

Why use knitted documents?

2. Code is formatted without special symbols

[Example of code symbols in Word that can mess up code]

Why use knitted documents?

3. Code can be re-run with updated or new datasets

Why use knitted documents?

4. Document is in plain text, so it can be tracked well with version control

[Picture of diff from git tracking]

Creating knitted documents in R

Pre-processing for research data

Preprocessing choices: GUI or script

Reproducible data pre-processing protocols