R language and reproducible data analysis

Qiang Shen

Sept. 30, 2016

Reproducible workshop

- thinking
- discussion

加强心理学研究的可重复性

liuzhen 发布于 2016-09-07 15:28:48 阅读次数 1110

主持人：胡传鹏
主讲人：王非、过继成思、宋梦迪、胡传鹏
特邀主讲人：沈强、魏立佳
时间：**10月14日9:30-17:30**
地点：陕西师范大学（雁塔校区），崇鉴楼二层闻道堂
报名费用：免费
报名方式：请务必在此链接中填写报名信息：**https://www.wenjuan.com/s/yUBr6v/**
报名要求：携带笔记本，提前下载R、R-Studio、lyx (https://www.lyx.org/Download) 以及R包rmarkdown；了解osf.io，以及熟悉参考文献。

problem

- Run analysis and get the result
- copy paste it into a file and write up the report.

There is no single document to integrate data analysis with textual representations; i.e. data, code, and text are not linked.

problem

- error-prone due to manual work
- tedious jobs to copy and paste
- Graphical User Interface is not recordable
- Tiny change need to redo the whole procedure.
- Communication cost is high for collaboration

reproducible research

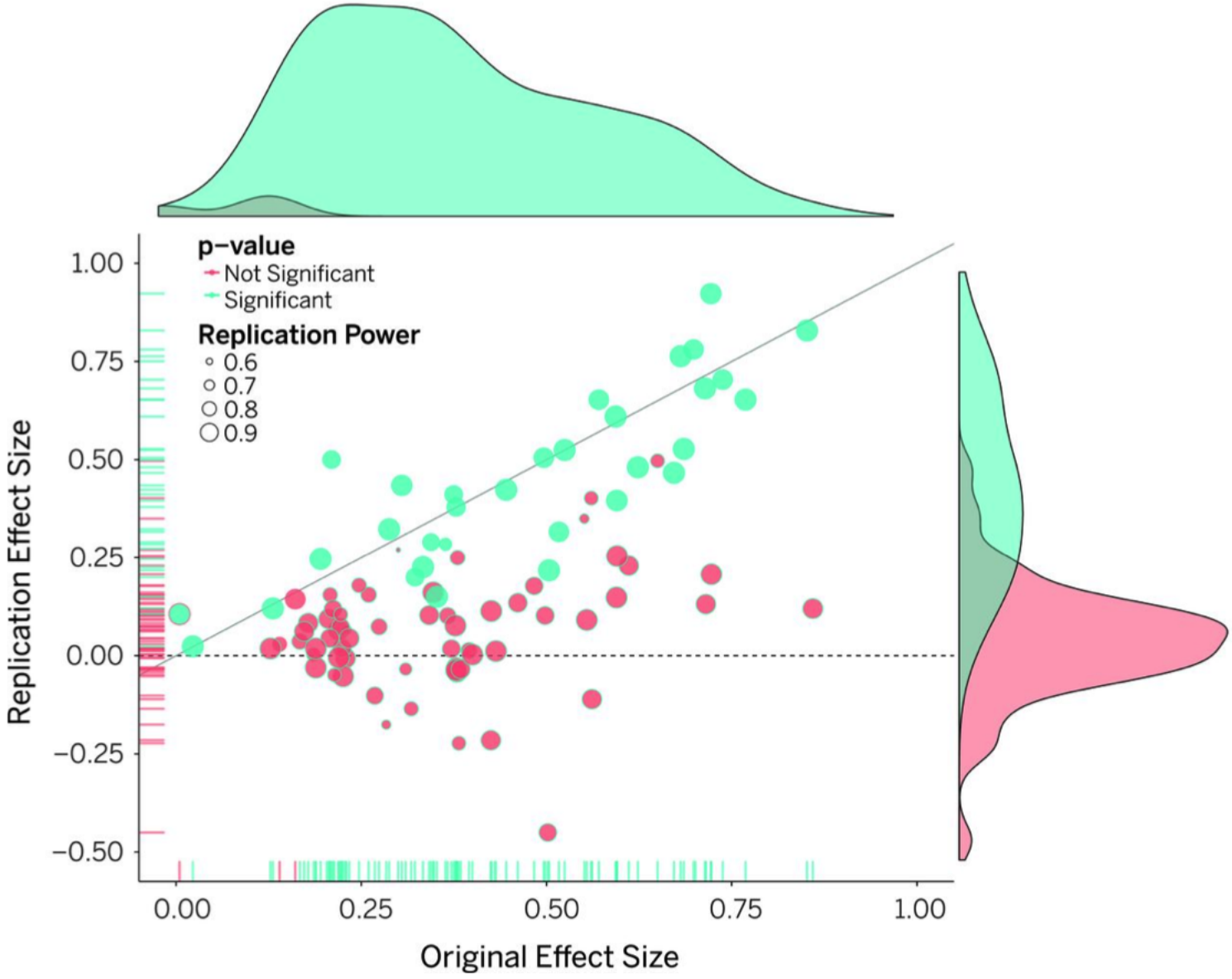
- Reproducibile property of research conclusion

reproducible research

- Reproducibile property of research conclusion
- reproducibility of your own work.

reproducible research

Estimating the reproducibility of psychological science



literate cons

- Text and code all can make documents difficult to read.
- Can substantially slow down processing of documents.

How Do I Make My Work Reproducible?

- version control
- literate programming

Version control

Start from a real scenario: daily tasks

- **Create** things
- **Save** things
- **Edit** things
- Save the thing **again**

Start from a real scenario

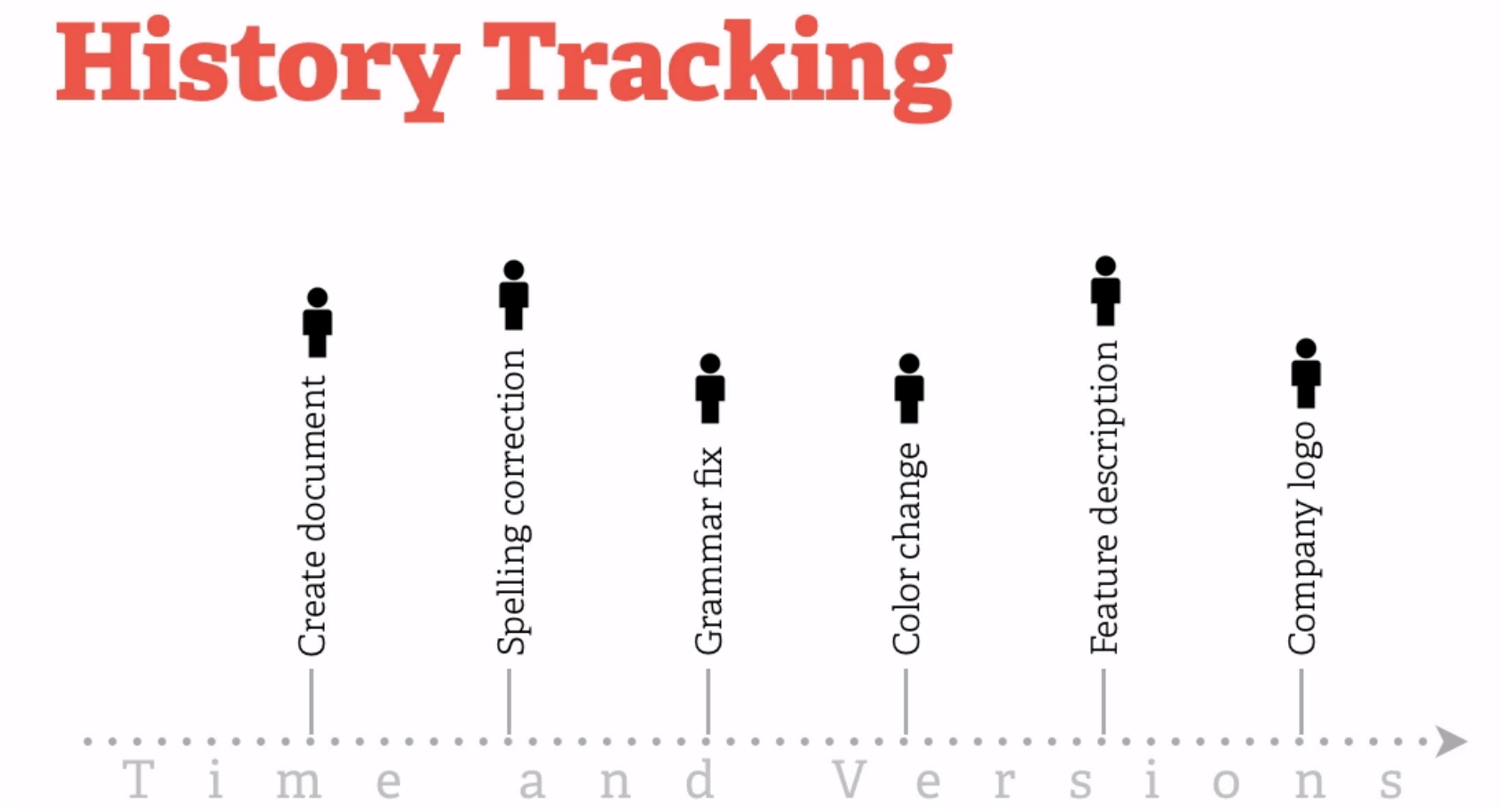


Figure 1: History tracking

Start from a real scenario

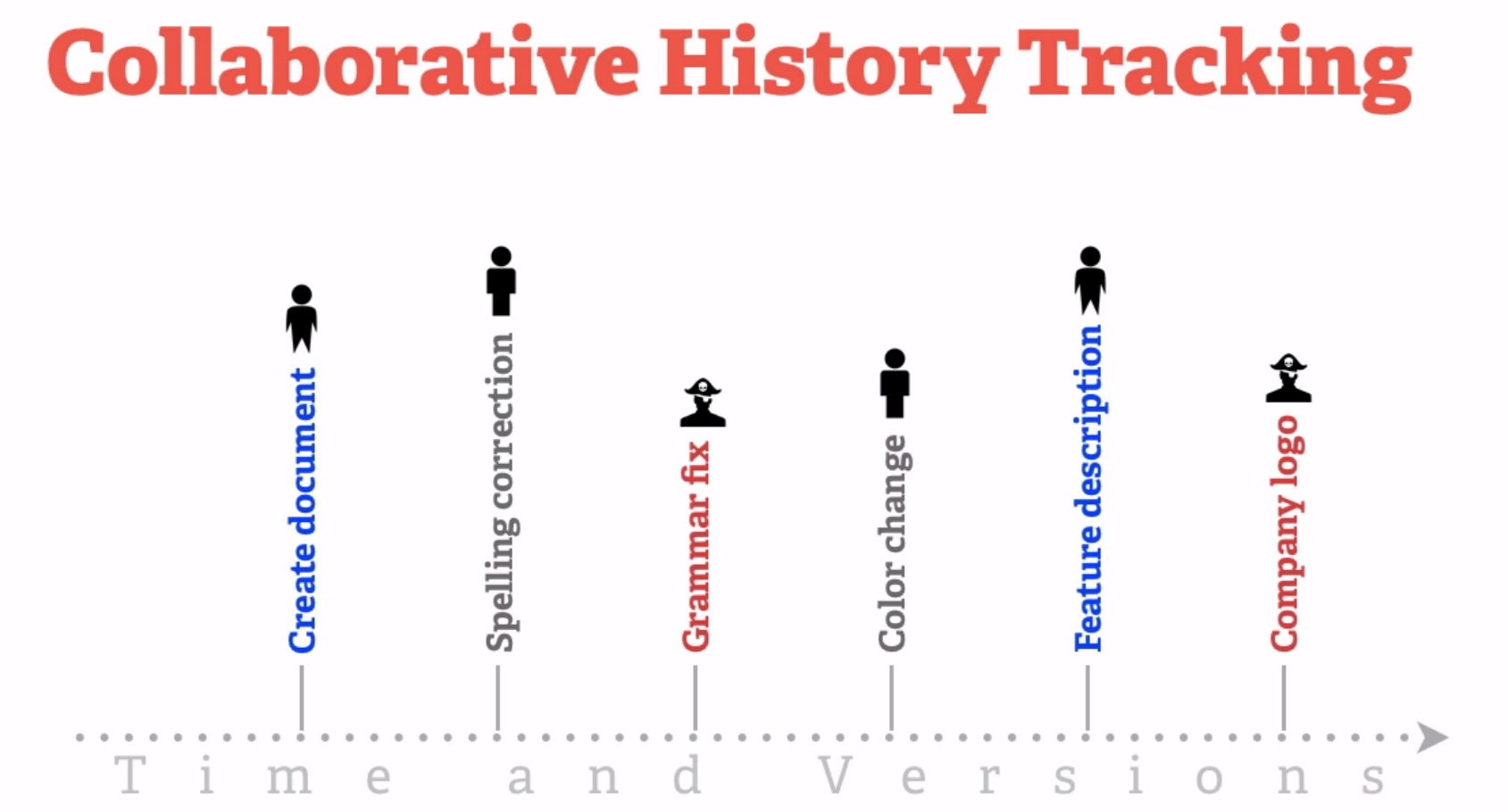


Figure 2: collaborative history tracking

Version control is important!

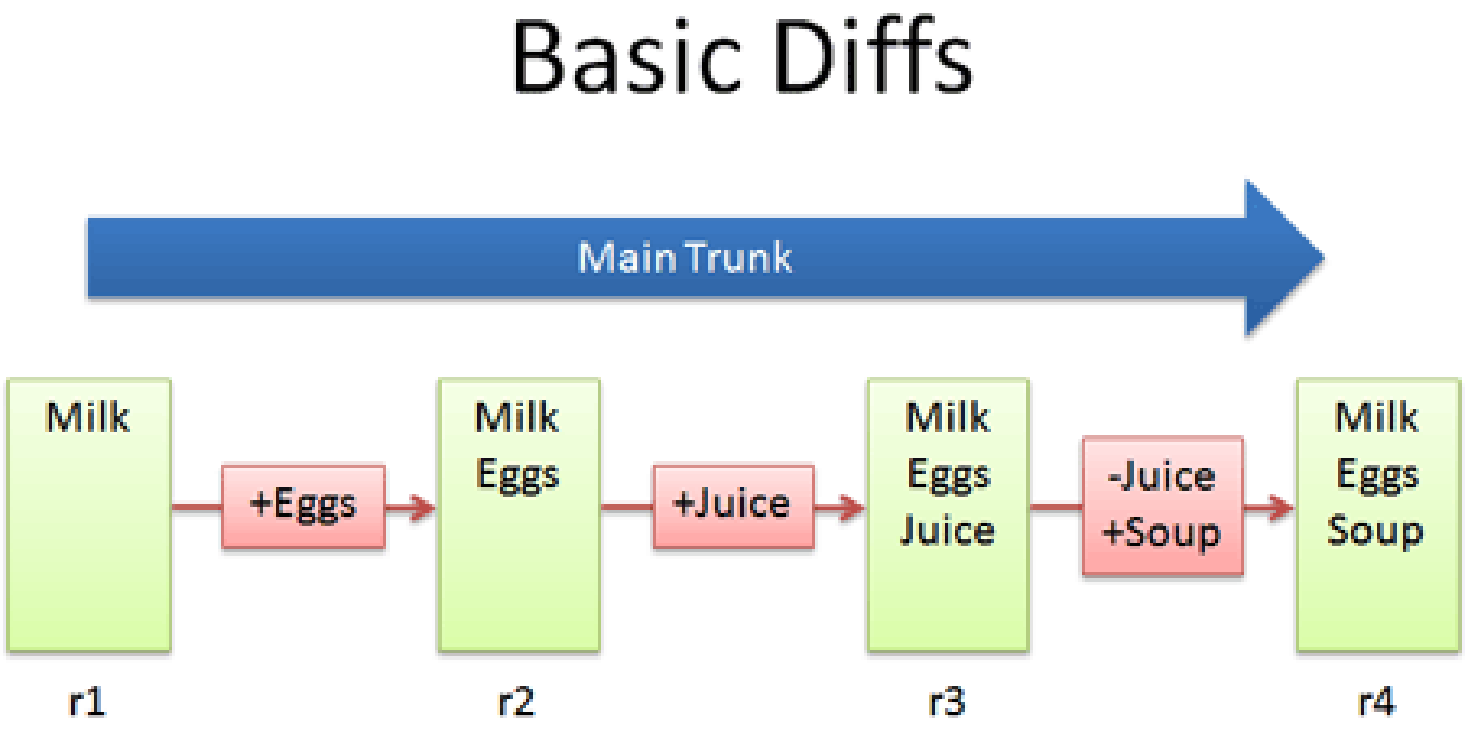


Figure 3: version control

Clound storage

- Dropbox
- Nutstore

Git

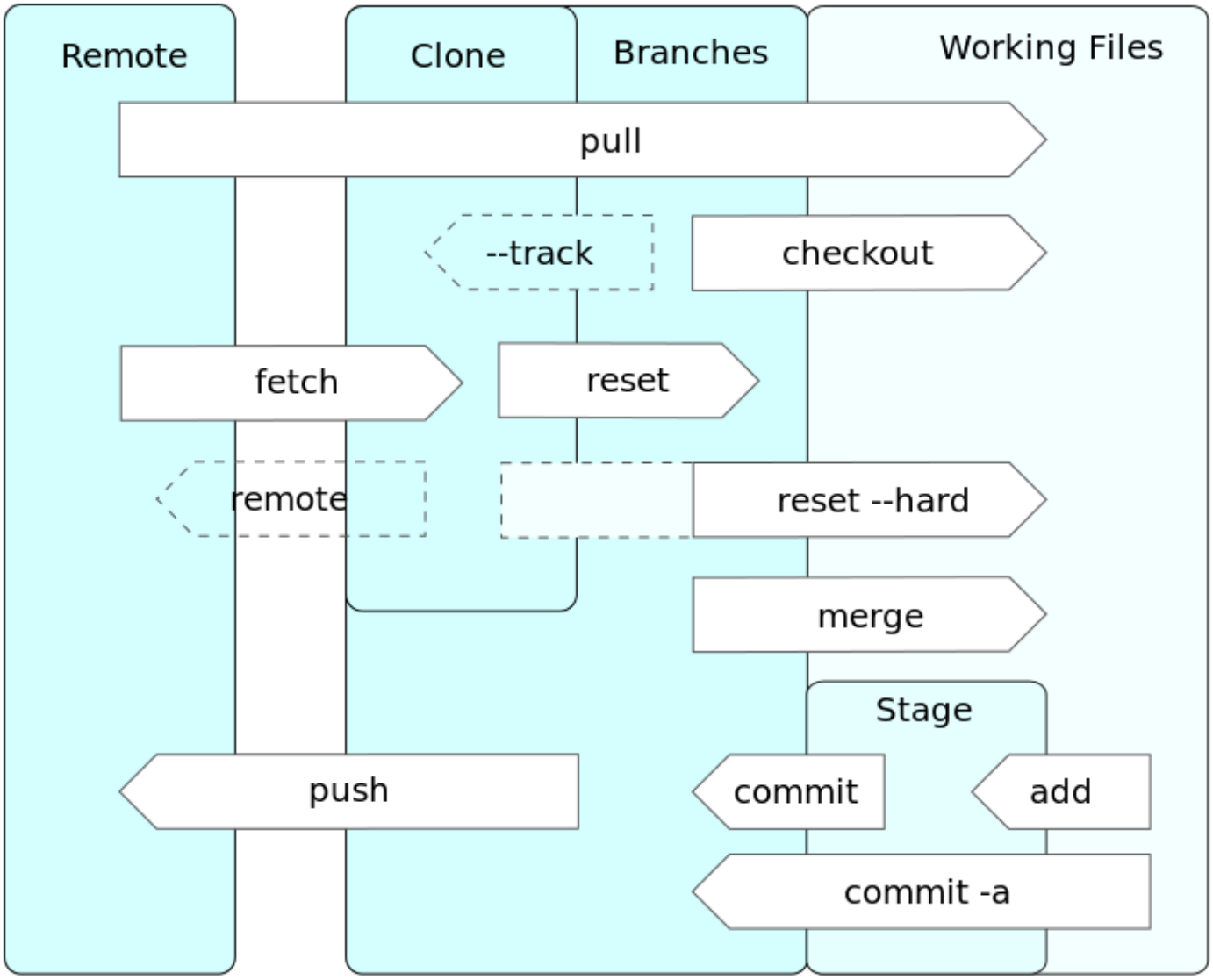


Figure 4: Git

Github

- a web-based Git repository hosting service

Prerequsites:

- Basic knowledge of R
- Working knowledge of RStudio
- Basic knowledge of command shell
- Curiosity to explore new stuff!

Literate programming

- conceived by Donald Knuth (Knuth,1984)
- mix the source code and documentation together
- code is divided into text and code “chunks”.
- **weaved** to produce documents and **tangled** to get source code

Literate programming

1. itself is only a concept or idea.

- A documentation language
- A programming language

2. S**weave** system (Friedrich Leisch) used LaTeX and R
3. **knit**r supports a variety of documentation languages

reproducible programming in Rstudio

- Sweave (rstudio->preference->Sweave)
- knitr

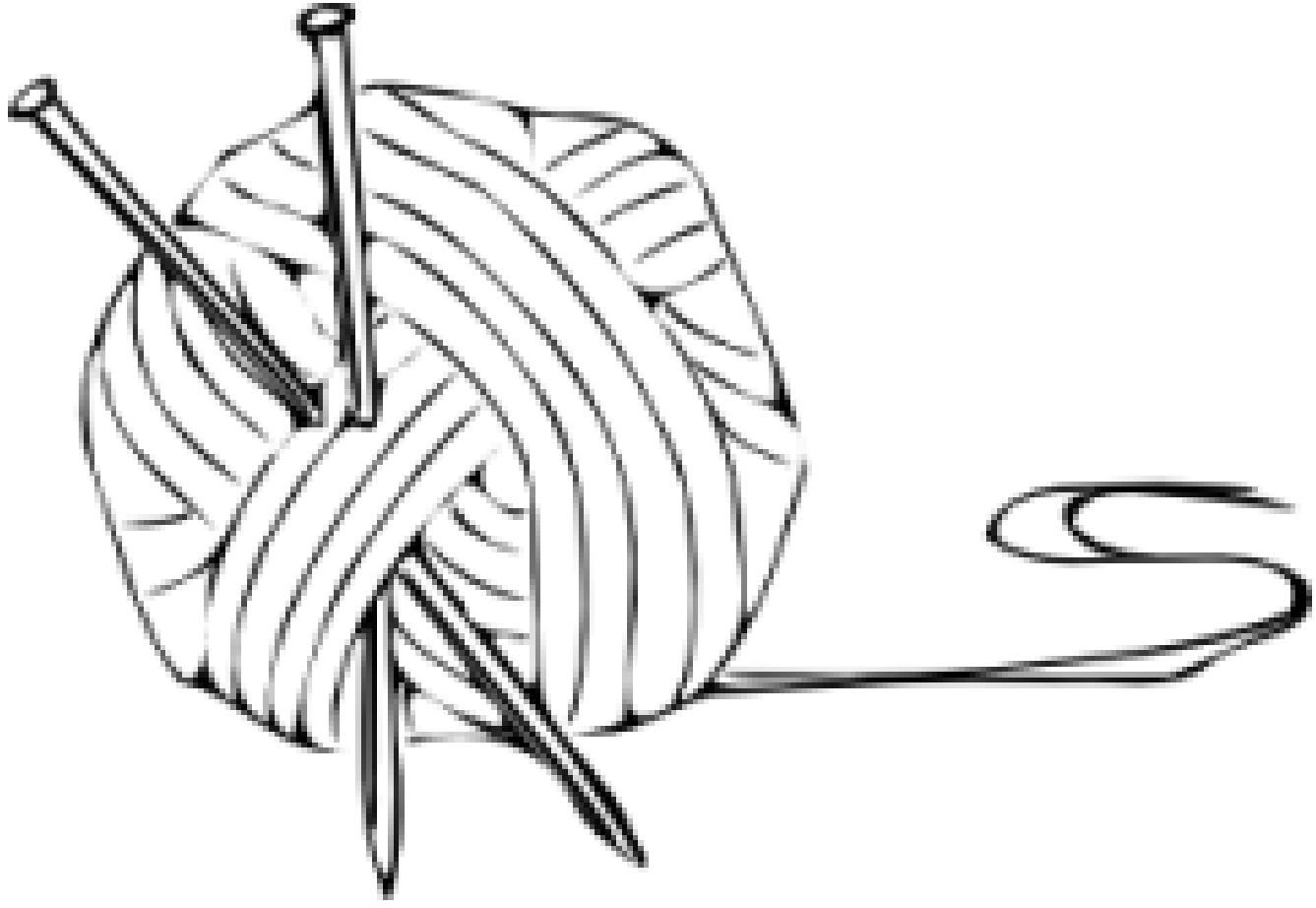
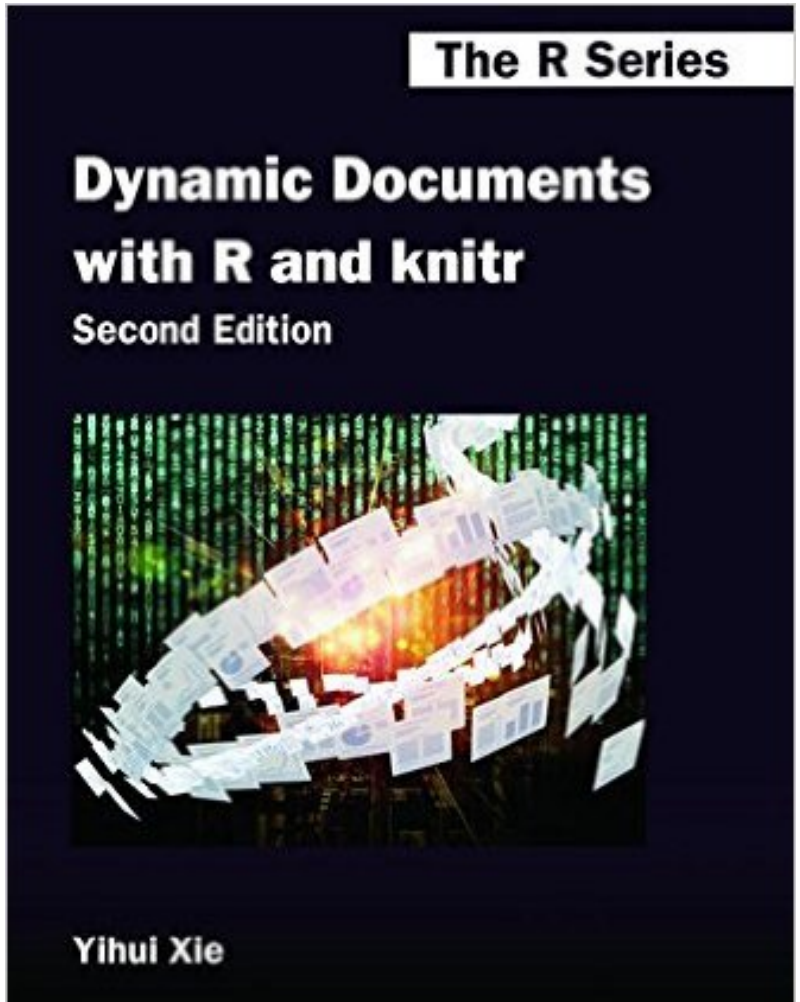


Figure 5

Knitr

- An R package written by Yihui Xie
- Supports **LaTeX**, **RMarkdown**,and HTML as documentation languages Can export to, do PDF, HTML
- Built right into RStudio for your convenience.



Knitr

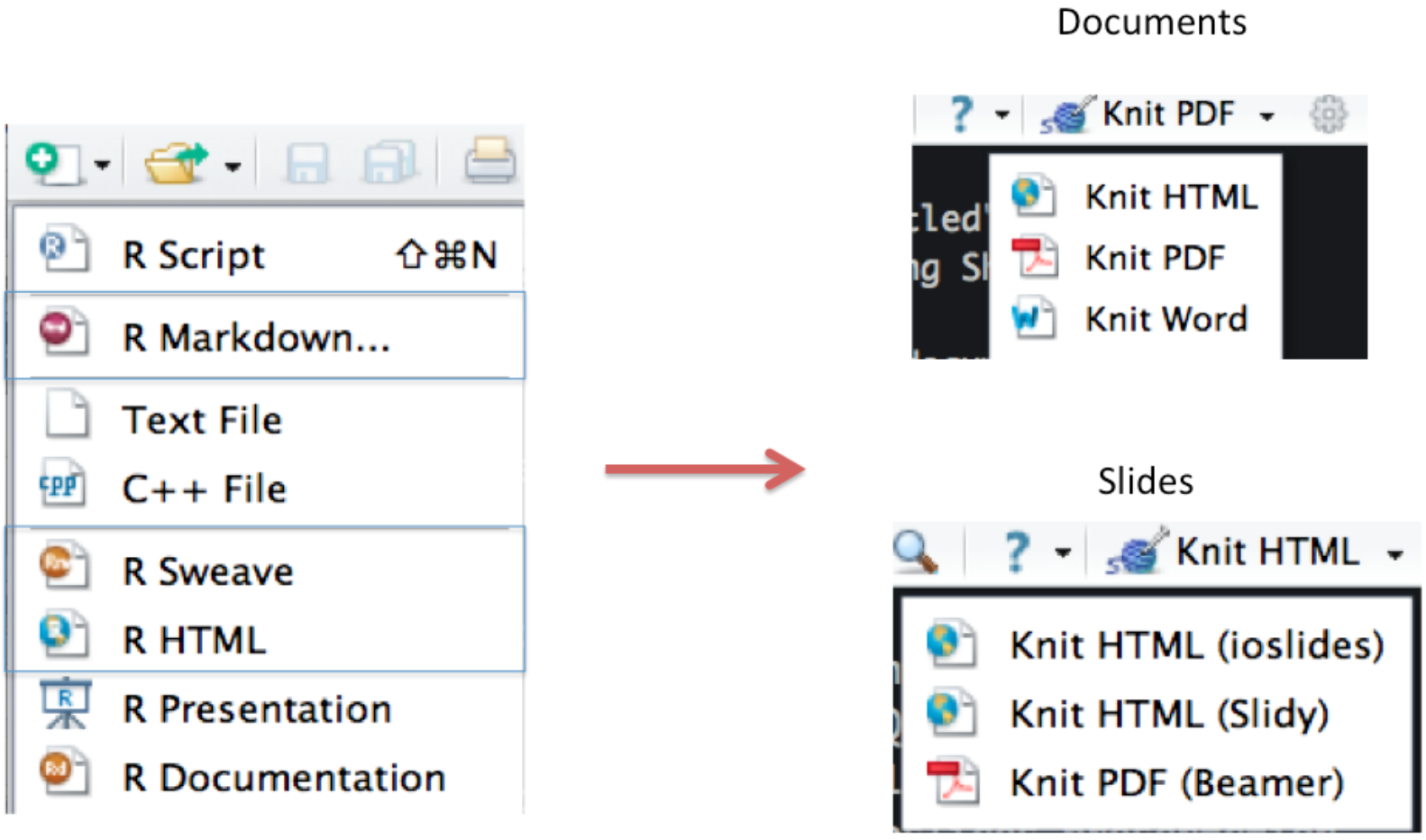


Figure 7

weave/knit in Rstudio

- Latex
- markdown

framework

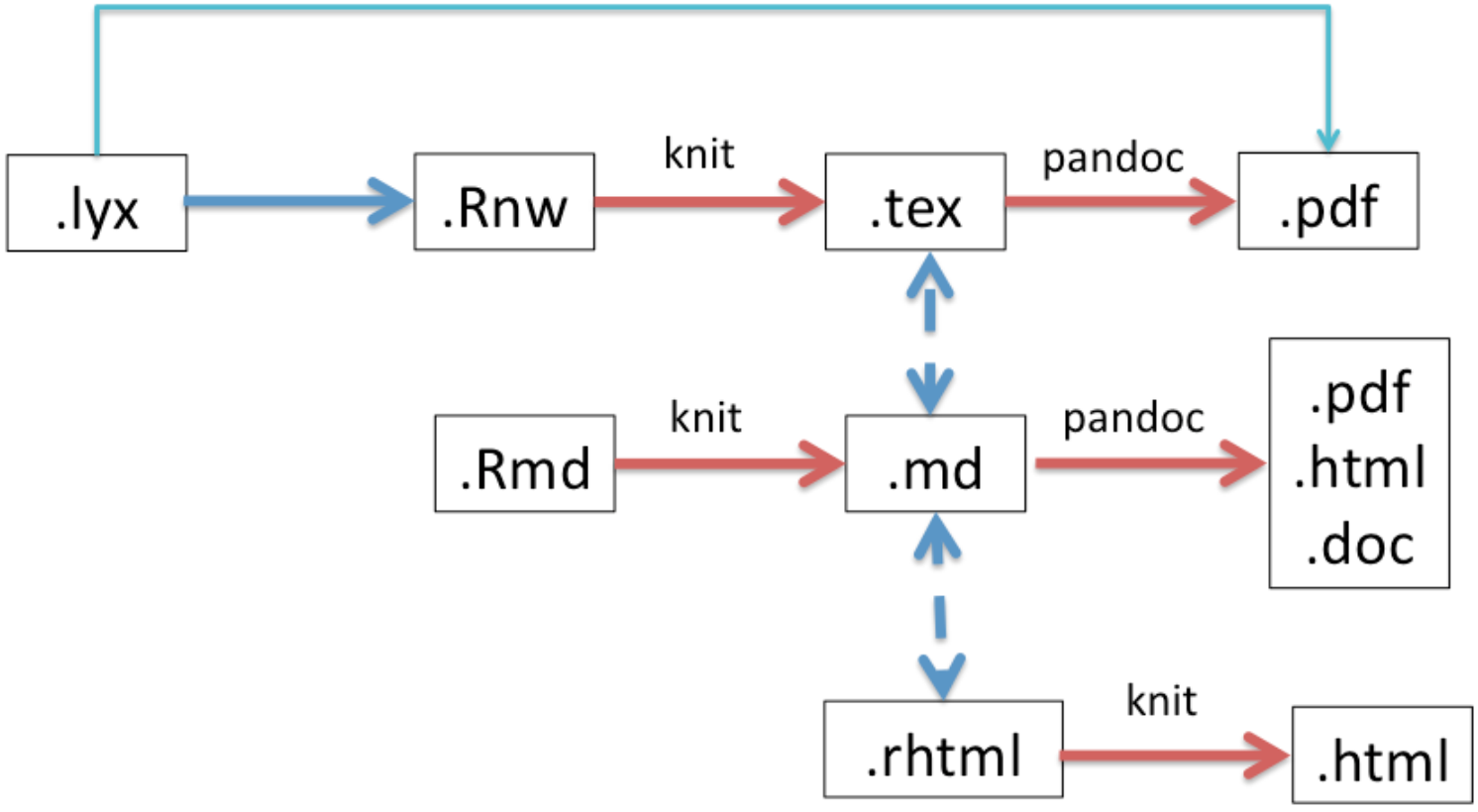
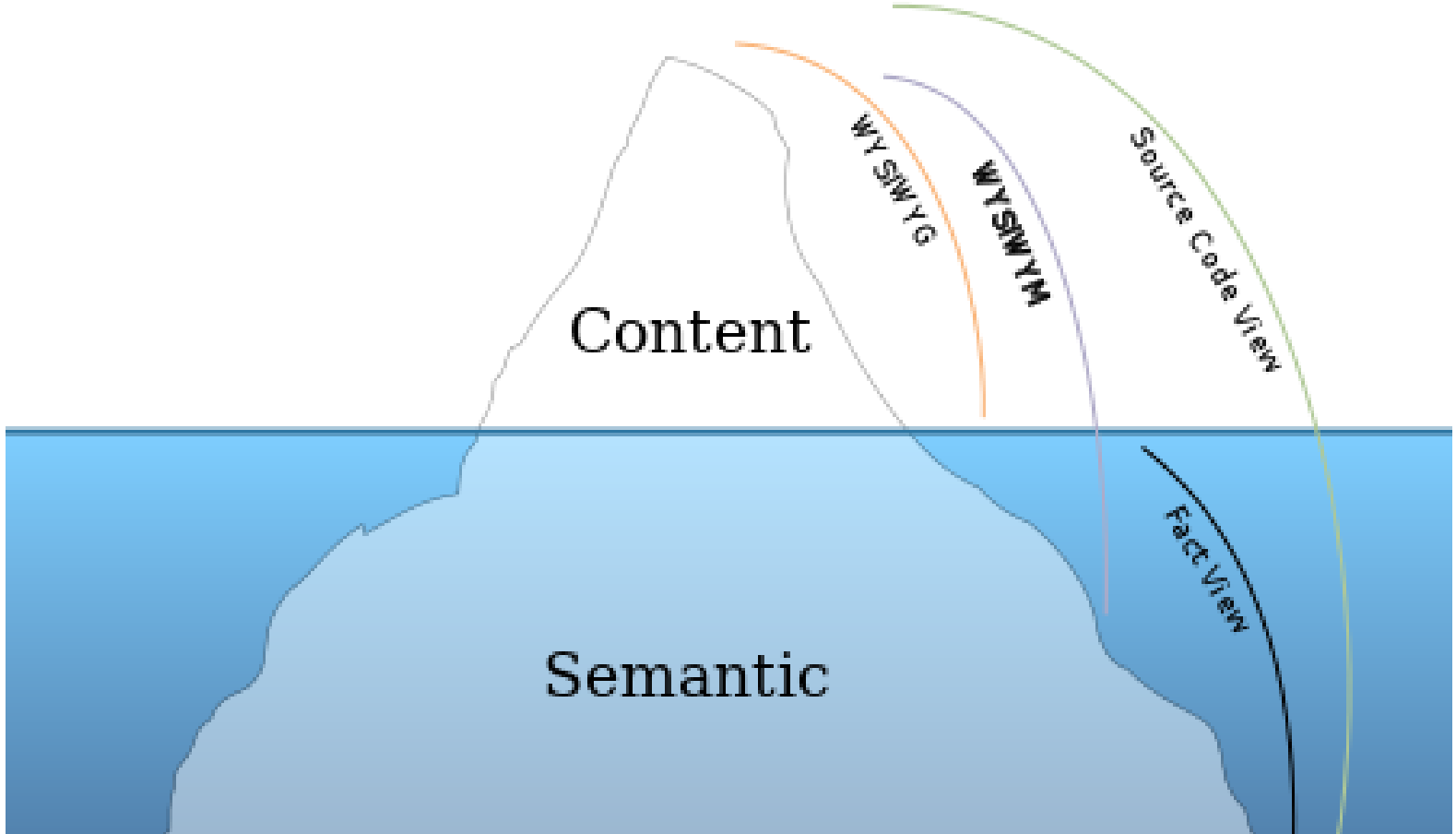


Figure 8

Latex

1. MiKTEX (Windows: `http://miktex.org/`),
2. MacTEX (BasicTeX) (Mac OS: `http://tug.org/mactex/`),
3. TEXLive (Linux: `http://tug.org/texlive/`).

- WYSIWYM: Document Processing
- WYSIWYG: Word Processing



Latex

.Rnw in Rstudio

- example-1.Rnw
- example-1-knitr.Rnw
- knitr-minimal.Rnw

`http://tobi.oetik-er.ch/lshort/lshort.pdf`

lyx

- lyx:`https://www.lyx.org/`
- compatible with knitr after LyX 2.0.3.

combines the power and flexibility of TeX/LaTeX with the ease of use of a graphical interface.

lyx

- knitr-minimal.lyx
- knitr.lyx

Latex/lyx

R code in .Rnw - chunks
- inline

```
##chunk
<<>>=
set.seed(1121)
(x=rnorm(20))
mean(x);var(x)
@
##inline
\Sexpr{pi}
```

lyx: table output

```
<<xtable, results="asis">>=
n <- 100
x <- rnorm(n)
y <- 2*x + rnorm(n)
out <- lm(y ~ x)
library(xtable)
xtable(summary(out)$coef, digits=c(0, 2, 2, 1, 2))
@
```

Latex/lyx

```
result<-summary(with(mtcars,lm(mpg~hp+wt)))
library(knitr)
kable(result$coe)
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	37.2272701	1.5987875	23.284689	0.0000000
hp	-0.0317729	0.0090297	-3.518712	0.0014512
wt	-3.8778307	0.6327335	-6.128695	0.0000011

What is markdown

- A simplified version of “markup” languages
- No special editor required
- Simple, intuitive formatting elements

markdown in R: rmarkdown

1. markdown

markdown_example.md 2. R code - chunks
- inline
demo.Rmd figure.Rmd

markdown in Rstudio

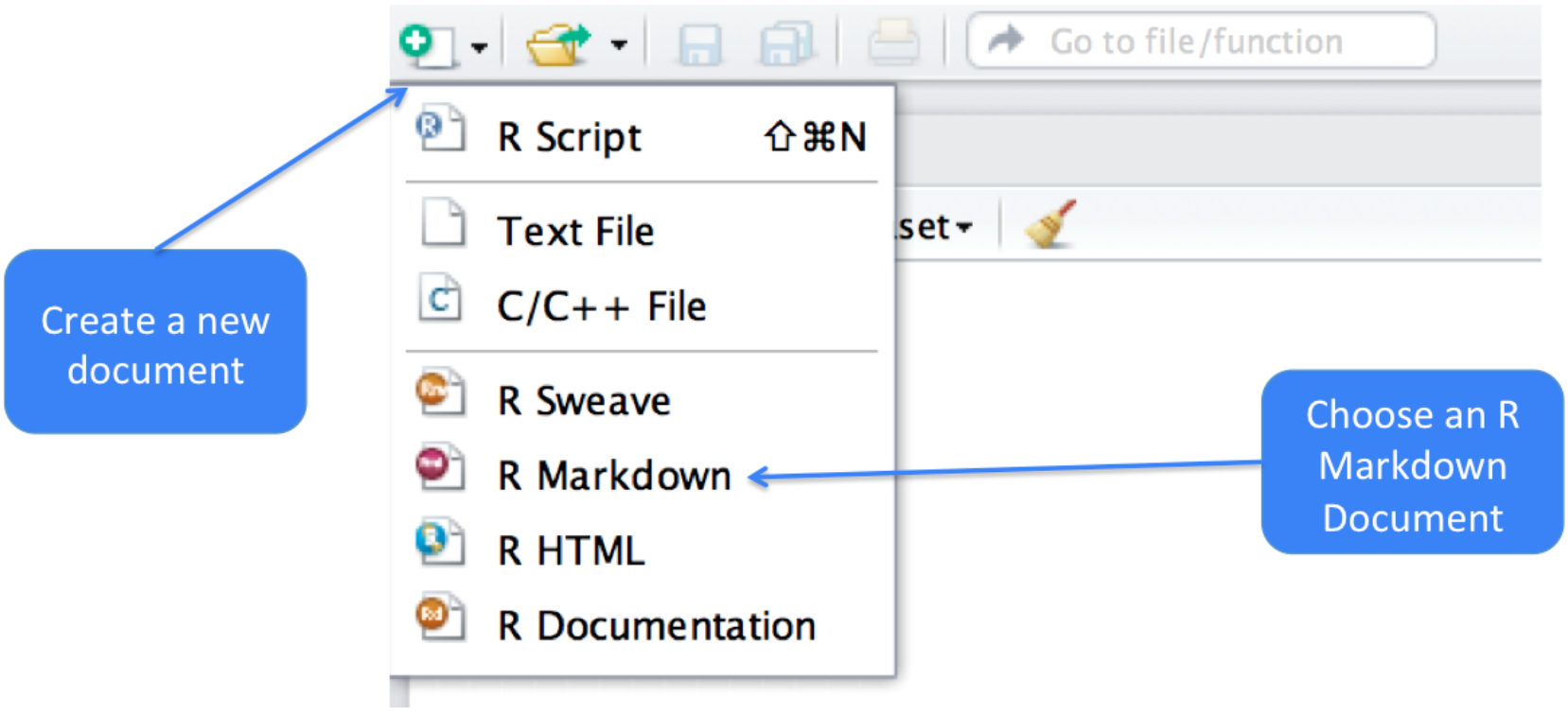


Figure 9

markdown in Rstudio

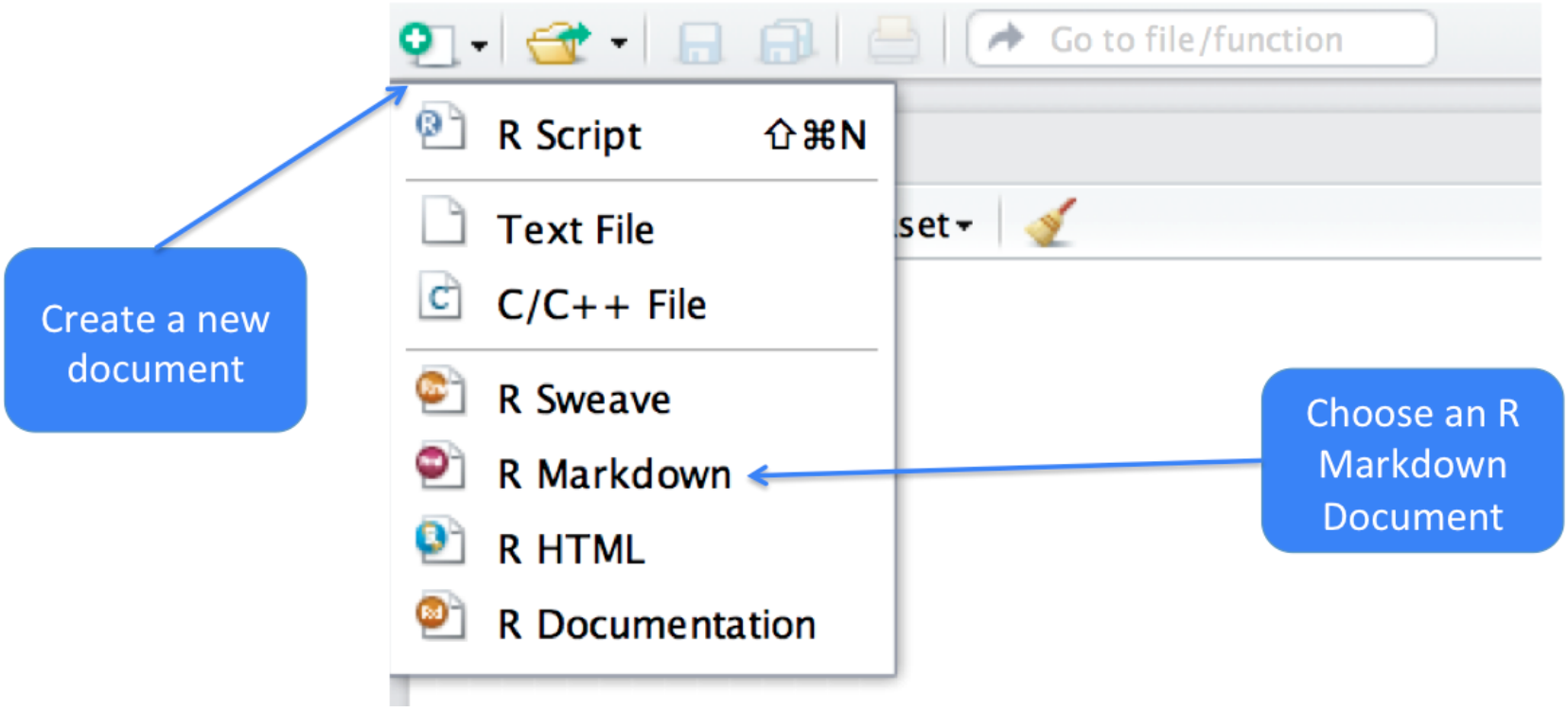


Figure 10

markdown in Rstudio

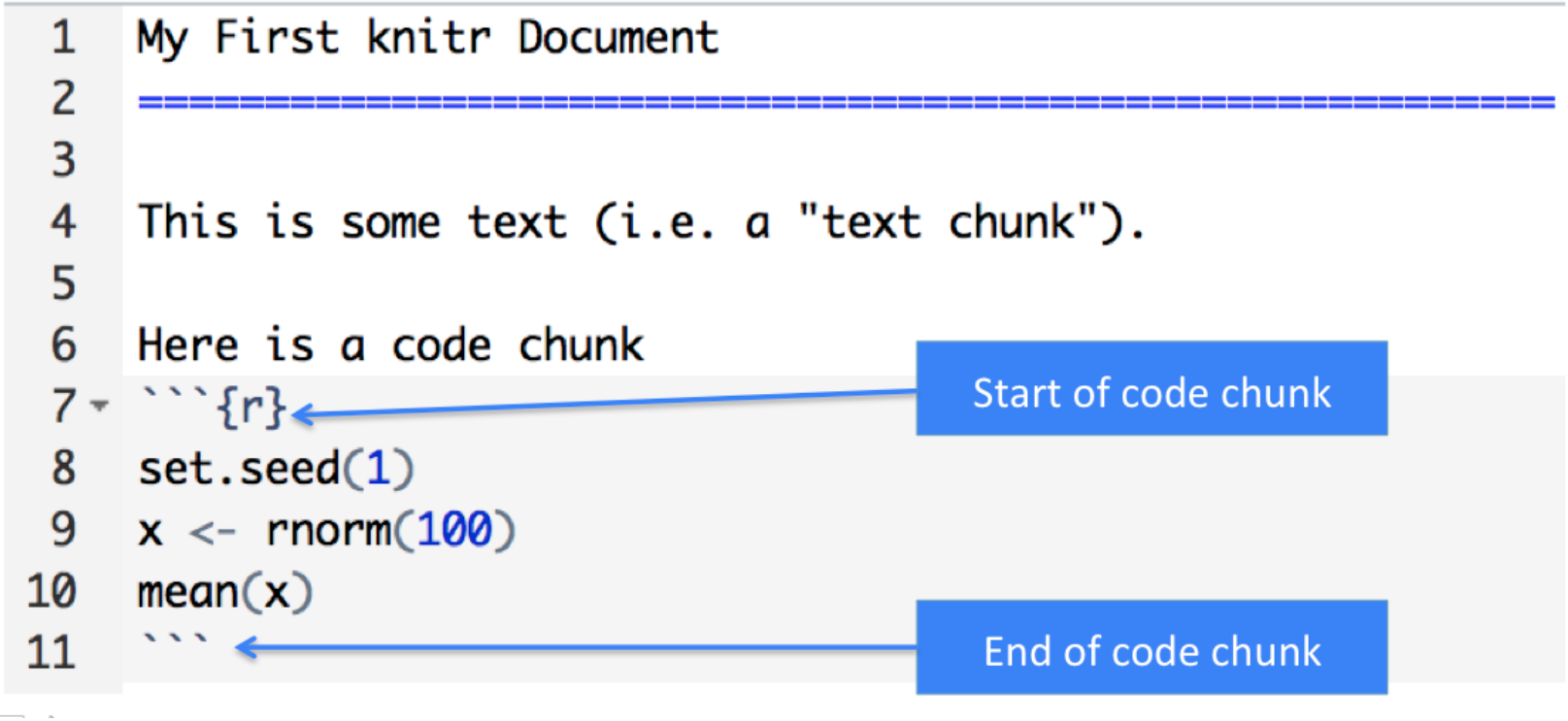


Figure 11

markdown in Rstudio

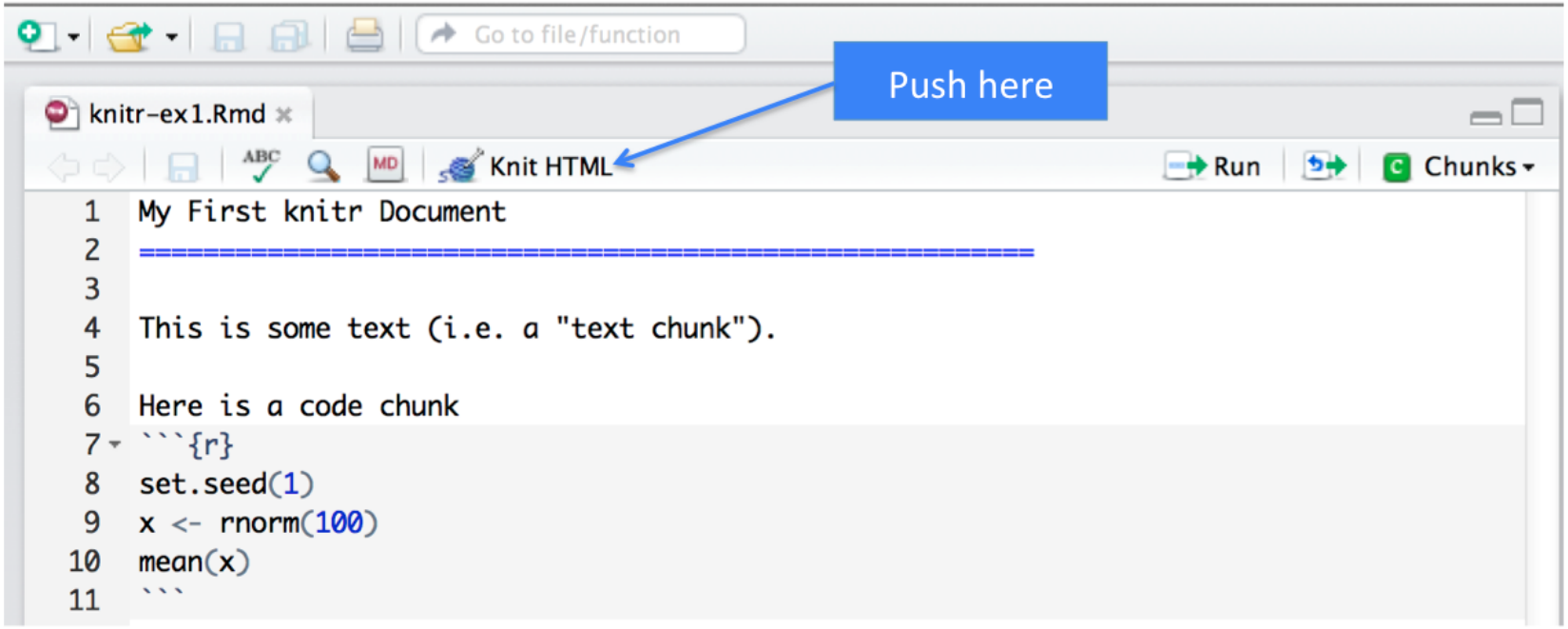


Figure 12

markdown in Rstudio

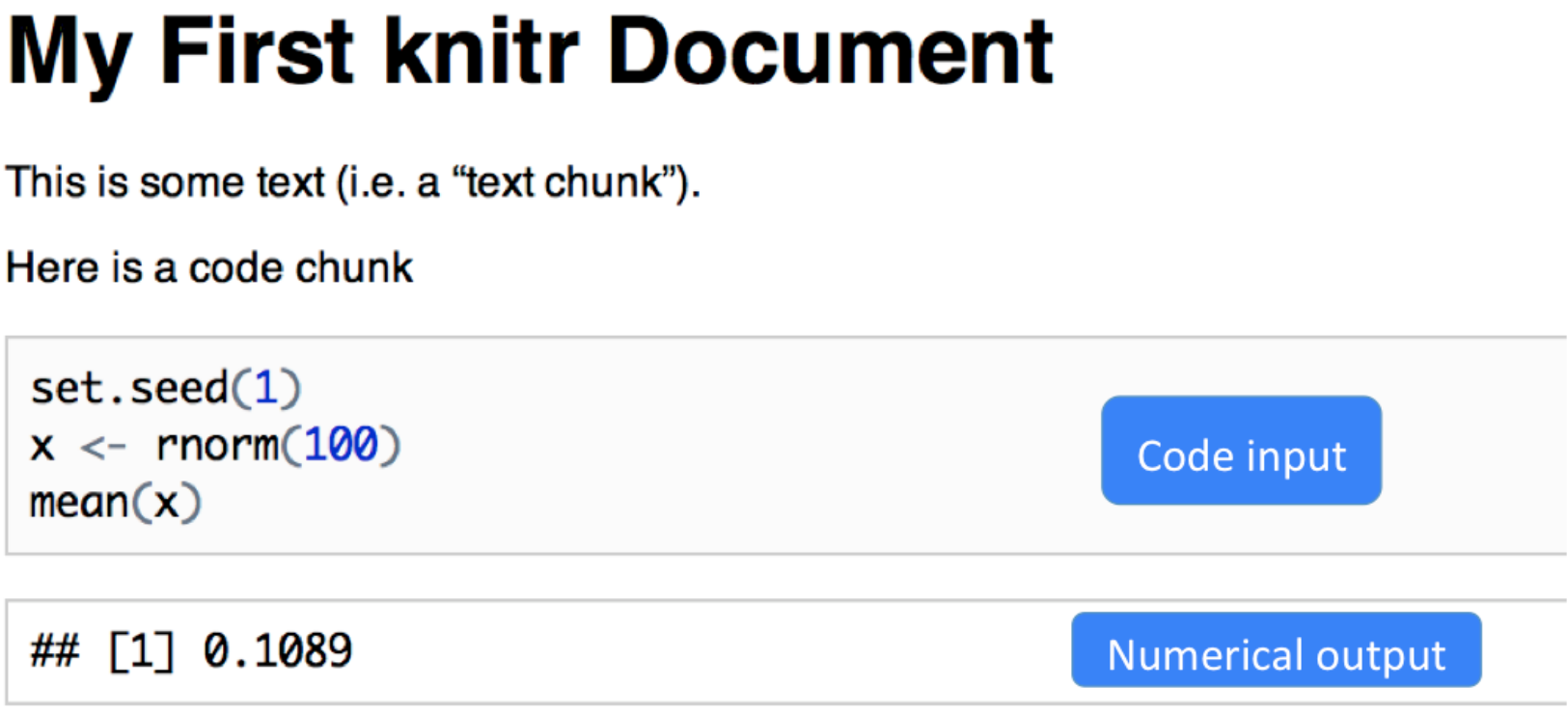


Figure 13

markdown in Rstudio

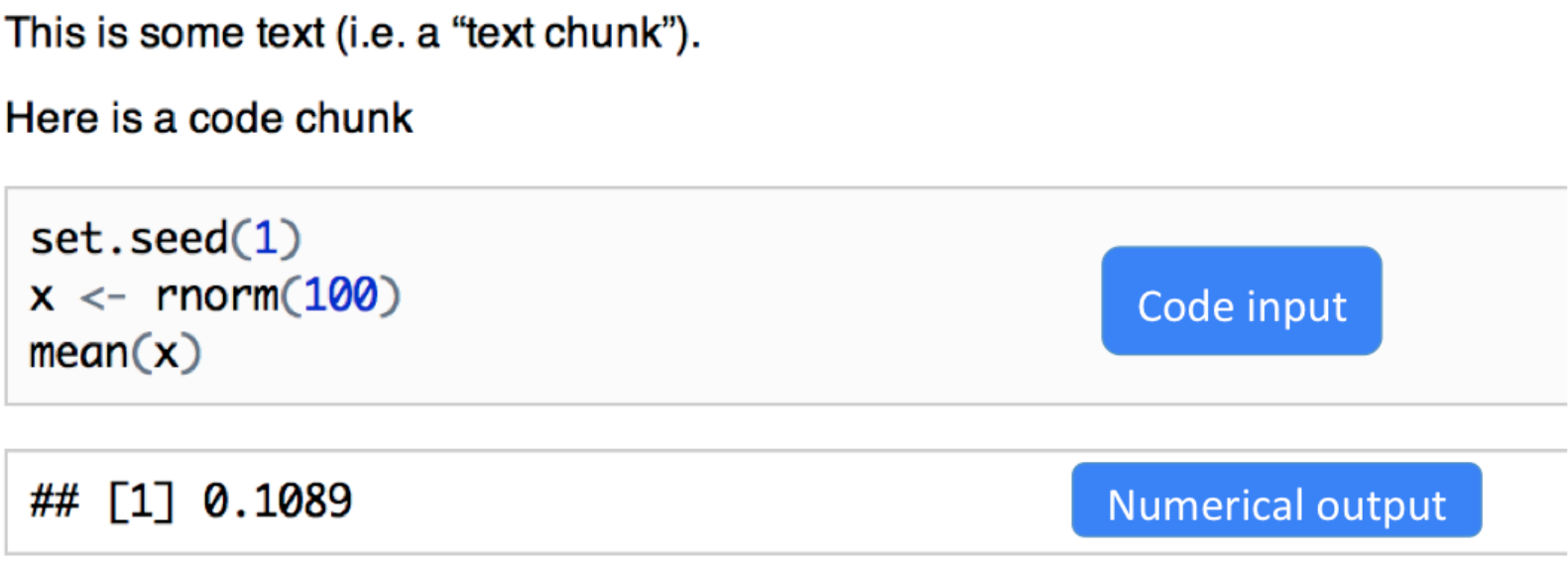


Figure 14

markdown in Rstudio

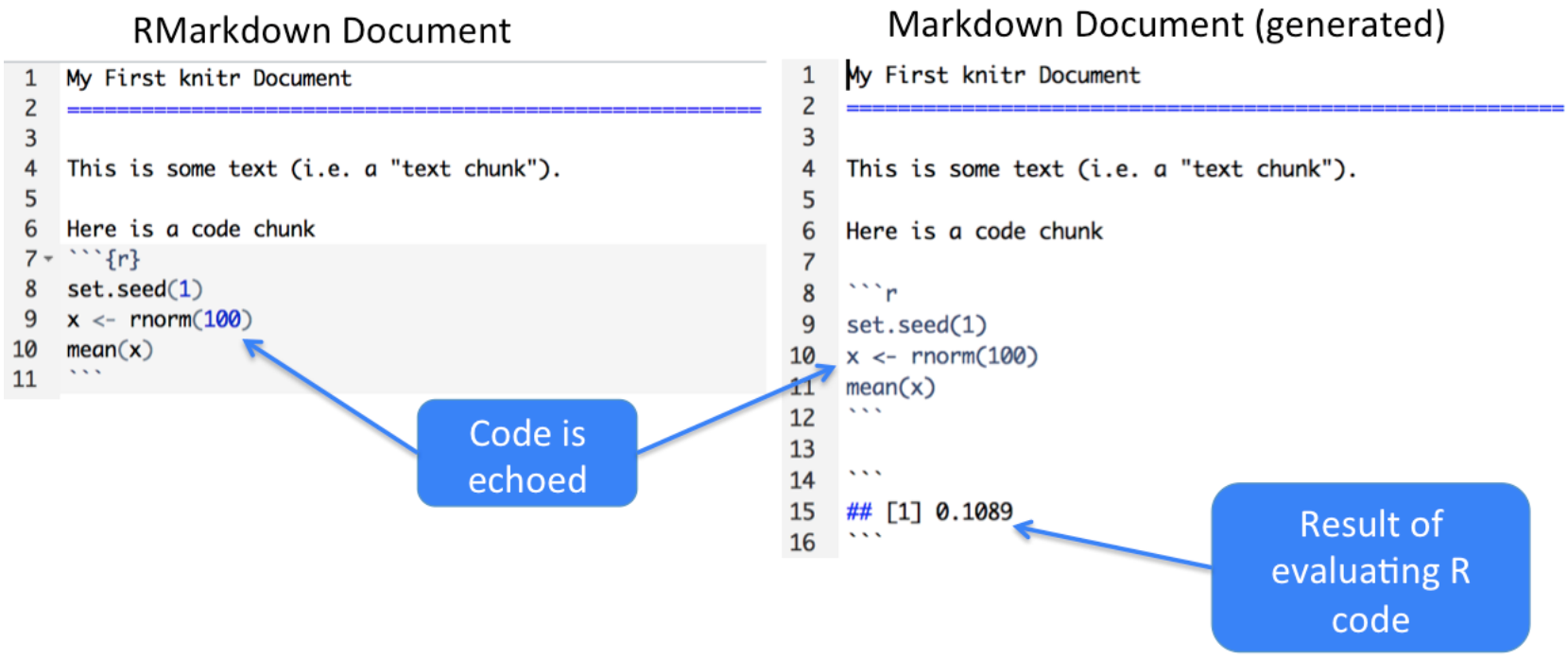


Figure 15

options

- options
- global options

Option	Effect
`eval`	Results printed when `TRUE`
`echo`	Code printed when `TRUE`
`include`	When `FALSE`, code is evaluated but neither the code nor results are printed.
`cache`	If the code has not changed, the results will be available but not evaluated again in order to save compilation time.
`fig.cap`	Caption text for images. Images will automatically be put into a special figure environment and be given a label based on the chunk label.
`fig.scap`	The short version of the image caption to be used in the list of captions
`out.width`	Width of displayed image
`fig.show`	Controls when images are shown. `'as.is'` prints them when they appear in code and `'hold'` prints them all at the end.
`dev`	Type of image to be printed, such as .png, .jpg, etc.
`engine`	`knitr` can handle code in other languages like Python, BASH, Perl, C++ and SAS.
`prompt`	Specifies the prompt character put before lines of code. If `FALSE`, there will be no prompt.
`comment`	For easier reproducibility, result lines can be commented out.

figures in rmarkdown

```
n <- 100
x <- rnorm(n)
par(mfrow=c(1,2), las=1)
for(i in 1:8) {
  y <- i*x + rnorm(n)
  plot(x, y, main=i)
}
```

figures in rmarkdown

```
![](figure.png)
```

figures in rmarkdown

```
library(png)
library(grid)
img <- readPNG("figure/format.png")
grid.raster(img)
```

alternative with command

- .Rmd -> .md -> .pdf/.doc/.html

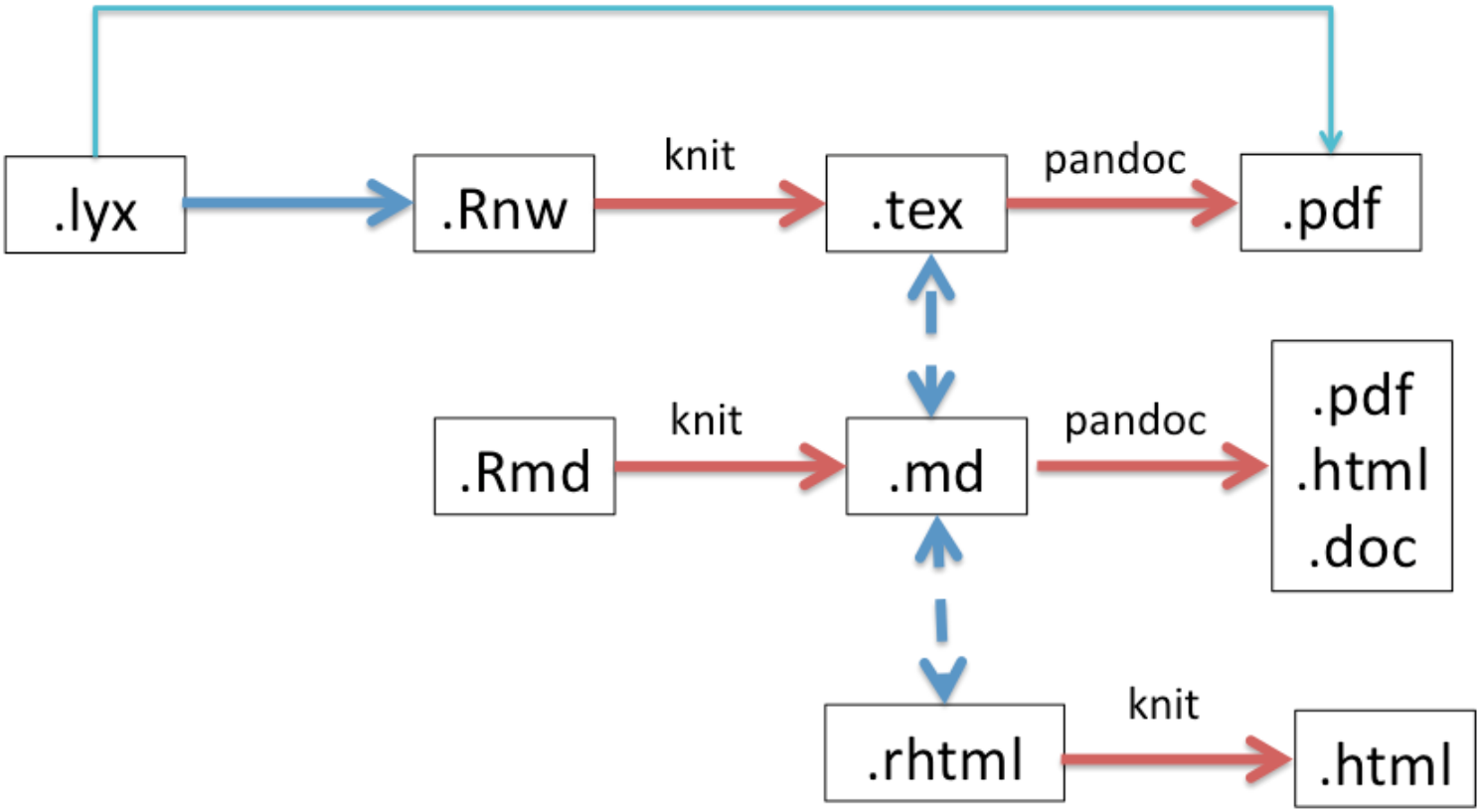


Figure 16

alternative with command

- .Rmd -> .md -> .pdf/.doc/.html
- .Rmd -> .md

```
library(knitr)
library(markdown)
##generate .md file
knit("test.Rmd")
## generate .html file
knit("003-minimal.Rhtml")
```

alternative with command

- .Rmd -> .md -> .pdf/.doc/.html
- .md -> .pdf/.doc/.html

```
##generate different format from .md file.
pandoc('test.md', format='html')  # HTML
system("pandoc test.md --latex-engine=xelatex -o test.pdf")
pandoc('test.md', format='docx')  # MS Word

## latex
pandoc('data_analysis.md', format='latex') # LaTeX/PDF
##or
system("pandoc -s test.md -t latex -o test.tex")

## slides
# system("pandoc -s -t slidy test.md -o My_Analysis.html")
system("pandoc -s -t beamer test.md -o My_Analysis.pdf")
```

slidify

http://slidify.org/start.html

```
library(slidify)
author('Qiang')
```

R code in rhtml

```
<!--begin.rcode
set.seed(1121)
(x=rnorm(20))
mean(x);var(x)
    end.rcode-->
```

Menu

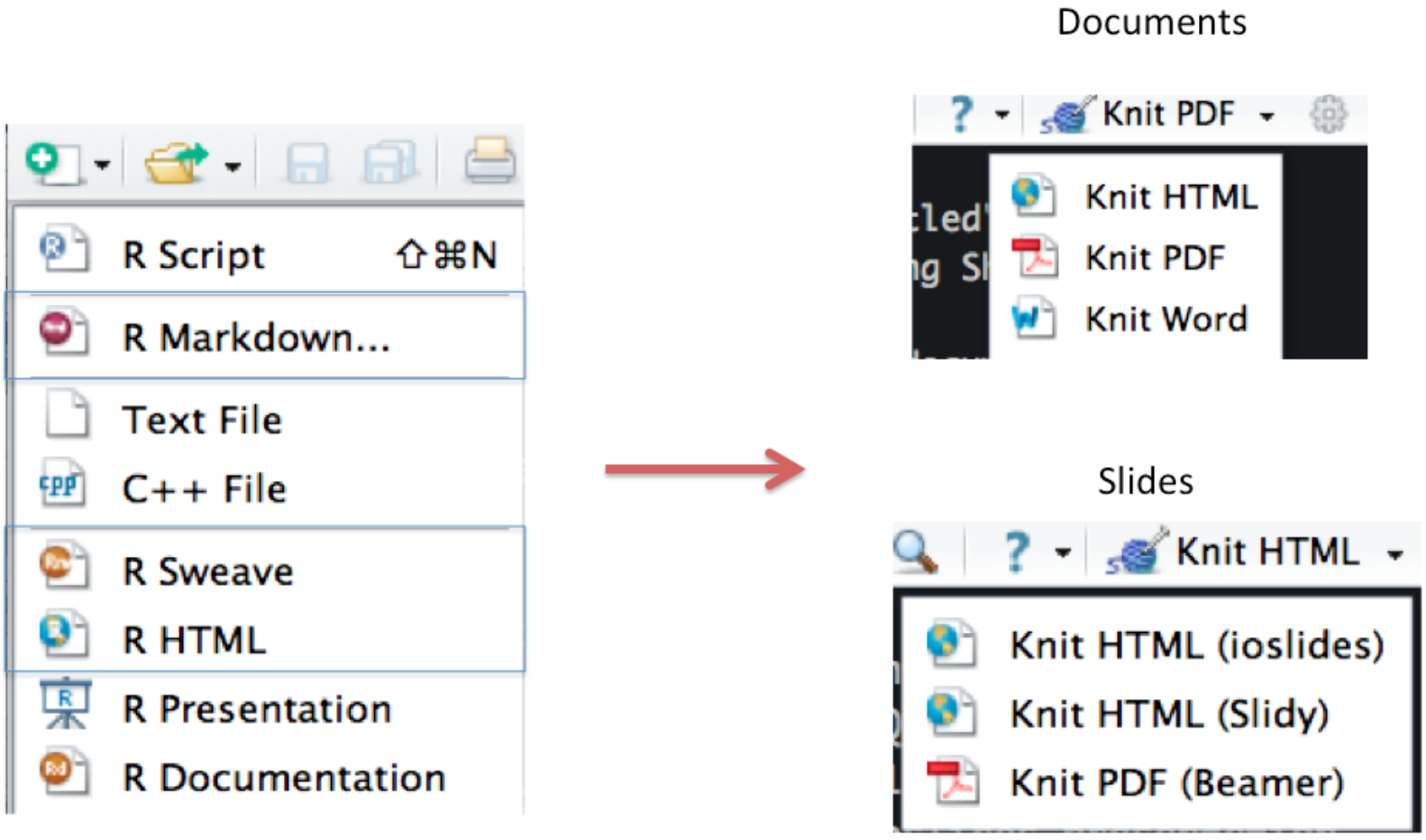


Figure 17

Framework

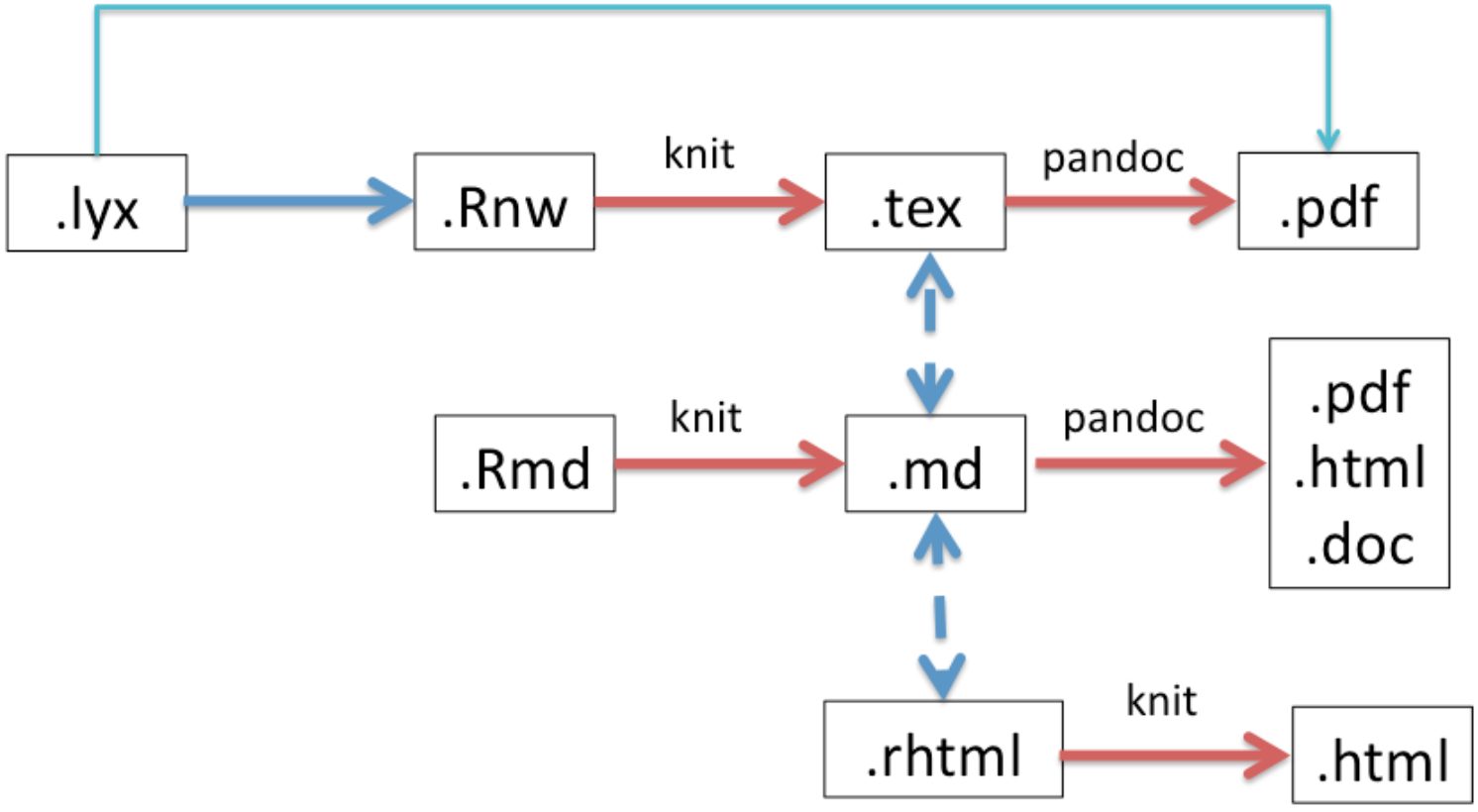


Figure 18

Summary

- Literate programming can be powerful to put text, code, data, output all in one document.
- knitr is a powerful tool for integrating code and text in a simple document format.