

# A brief introduction to reproducible research with Emacs' Org mode and R

Bernd Weiß

<http://berndweiss.net>  
[bernd.weiss@uni-koeln.de](mailto:bernd.weiss@uni-koeln.de)

2012-07-06

# Outline

Disclaimer and Acknowledgment

Introduction

- Motivating problem

- Emacs

- Org mode

- Babel: Active code in Org mode

Babel and R

- Introduction

- Tangling and weaving

- A short example

- A brief Org example

R and other languages (Python)

References

# Topic

## Disclaimer and Acknowledgment

## Introduction

Motivating problem

Emacs

Org mode

Babel: Active code in Org mode

## Babel and R

Introduction

Tangling and weaving

A short example

A brief Org example

## R and other languages (Python)

## References

# Disclaimer and Acknowledgment

- ▶ I am by no means an expert user of Org mode or Org Babel. However, I know enough to acknowledge the incredible work of Carsten Dominik (created Org in 2003), Eric Schulte and Dan Davison (developed the Babel functionality within Org).
- ▶ All materials can be found on github:  
[https://github.com/berndweiss/ps2012-07-KRUG\\_org\\_r](https://github.com/berndweiss/ps2012-07-KRUG_org_r)

# Topic

Disclaimer and Acknowledgment

## Introduction

Motivating problem

Emacs

Org mode

Babel: Active code in Org mode

## Babel and R

Introduction

Tangling and weaving

A short example

A brief Org example

R and other languages (Python)

References

# Topic

Disclaimer and Acknowledgment

Introduction

Motivating problem

Emacs

Org mode

Babel: Active code in Org mode

Babel and R

Introduction

Tangling and weaving

A short example

A brief Org example

R and other languages (Python)

References

# Motivating problem

- ▶ Writing an empirical research paper
  - ▶ is almost never a linear process.
  - ▶ also means that your colleagues need to trust you.  
However, how can I make them trust me?
- ▶ So, we need a tool that provides
  - ▶ “dynamic elements”, i.e. updates tables and graphs every time the data gets updated.
  - ▶ full access to code to check for correctness (i.e. “reproducible research”).

# Literate programming and reproducible research

**Literate programming (LP)** Enhances traditional software development by embedding code in explanatory essays and encourages treating the act of development as one of communication with future maintainers.

**Reproducible research (RR)** Embeds executable code in research reports and publications, with the aim of allowing readers to re-run the analyses described.

(Source: Schulte et al. 2012: 3)



# Existing tools

Tool	LP	RR	$\text{\LaTeX}$ Export	HTML Export	Language
Javadoc	partial	no	no	yes	Java
POD	partial	no	no	yes	Perl
Haskell	partial	no	yes	yes	Haskell
noweb	yes	no	yes	yes	any
cweb	yes	no	yes	yes	C/C++
Sweave	partial	yes	yes	yes	R
SASweave	partial	yes	yes	yes	R/SAS
Statweave	partial	yes	yes	yes	any
Scribble	yes	yes	yes	yes	scheme
Knitr	partial	yes	yes	yes	R
dexy	yes	yes	yes	yes	any
Org-mode	yes	yes	yes	yes	any

(Source: Schulte et al. 2012: 6, modified and updated)

# Topic

Disclaimer and Acknowledgment

## Introduction

Motivating problem

**Emacs**

Org mode

Babel: Active code in Org mode

## Babel and R

Introduction

Tangling and weaving

A short example

A brief Org example

R and other languages (Python)

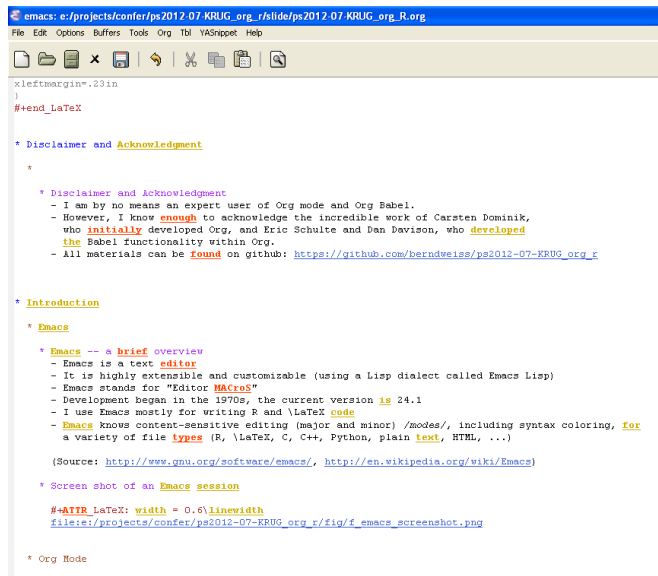
References

# Emacs – a brief overview

- ▶ Emacs is a text editor
- ▶ It is highly extensible and customizable (using a Lisp dialect called Emacs Lisp)
- ▶ Emacs stands for “Editor MACroS”
- ▶ Development began in the 1970s, the current version is 24.1
- ▶ I use Emacs mostly for writing R and  $\text{\LaTeX}$ -code
- ▶ Emacs knows content-sensitive editing (major and minor) *modes*, including syntax coloring, for a variety of file types (R,  $\text{\LaTeX}$ , C, C++, Python, plain text, HTML, ...)

(Source: <http://www.gnu.org/software/emacs/>,  
<http://en.wikipedia.org/wiki/Emacs>)

# Screen shot of an Emacs session



```
emacs: e:/projects/confer/ps2012-07-KRUG_org_r/slide/ps2012-07-KRUG_org_R.org
File Edit Options Buffers Tools Org Tbl YASnippet Help

xleftmargin=.23in
)
#+end_LaTeX

* Disclaimer and Acknowledgment

*
* Disclaimer and Acknowledgment
- I am by no means an expert user of Org mode and Org Babel.
- However, I know enough to acknowledge the incredible work of Carsten Dominik,
  who initially developed Org, and Eric Schulte and Dan Davison, who developed
  the Babel functionality within Org.
- All materials can be found on github: https://github.com/berndweiss/ps2012-07-KRUG\_org\_r

* Introduction

* Emacs

* Emacs -- a brief overview
- Emacs is a text editor
- It is highly extensible and customizable (using a Lisp dialect called Emacs Lisp)
- Emacs stands for "Editor MACroS"
- Development began in the 1970s, the current version is 24.1
- I use Emacs mostly for writing R and \LaTeX code
- Emacs knows content-sensitive editing (major and minor) /modes/, including syntax coloring, for
  a variety of file types (R, \LaTeX, C, C++, Python, plain text, HTML, ...)

(Source: http://www.gnu.org/software/emacs/, http://en.wikipedia.org/wiki/Emacs)

* Screen shot of an Emacs session

#+ATTR LaTeX: width = 0.6\linewidth
file:e:/projects/confer/ps2012-07-KRUG_org_r/fig/f_emacs_screenshot.png

* Org Mode
```

# Topic

Disclaimer and Acknowledgment

## Introduction

Motivating problem

Emacs

**Org mode**

Babel: Active code in Org mode

## Babel and R

Introduction

Tangling and weaving

A short example

A brief Org example

R and other languages (Python)

References

# Org mode

- ▶ Org is an Emacs (major) mode for notes, planning, and authoring
- ▶ Org files are plain-text files
- ▶ I use Org for
  - ▶ writing technical documents (can be exported to PDF via pdfLaTeX or HTML) or creating slides
  - ▶ organizing my professional and private life (appointments, todo lists, ...)
  - ▶ maintaining my personal “knowledge base”
  - ▶ ...
- ▶ More infos on <http://www.orgmode.org> (tutorials, videos, ...)

# An example Org file (source file)

```
1  #+TITLE: Hello World!
2  #+AUTHOR: Martin Mosquito
3  #+DATE: <2012-07-06 Fr>
4
5  \thispagestyle{empty}
6
7  * Headline
8
9  Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed
10 diam nonumy eirmod tempor invidunt ut labore et dolore magna
11 aliquyam erat, sed diam voluptua. At vero eos et accusam et
12 justo duo dolores et ea rebum. Stet clita kasd gubergren, no
13 sea takimata sanctus est Lorem ipsum dolor sit.
14
15 A list:
16 - Item 1
17 - Item 2
18
19 ** Subsection
20
21 Lorem ipsum dolor sit amet, consetetur sadipscing elitr,
22 sed diam nonumy eirmod tempor invidunt ut labore et dolore
23 magna aliquyam erat, sed diam
24
25 \begin{equation}
26 y = \alpha + \beta_1 x_1 + \epsilon
27 \end{equation}
```

# An example Org file (exported to PDF)

Hello World!

Martin Mosquito

2012-07-06

## Contents

<b>1</b>	<b>Headline</b>	<b>1</b>
1.1	Subsection . . . . .	1

## 1 Headline

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit.

A list:

- Item 1
- Item 2

### 1.1 Subsection

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam

$$y = \alpha + \beta_1 x_1 + \epsilon \tag{1}$$



# Topic

Disclaimer and Acknowledgment

## Introduction

Motivating problem

Emacs

Org mode

**Babel: Active code in Org mode**

## Babel and R

Introduction

Tangling and weaving

A short example

A brief Org example

R and other languages (Python)

References

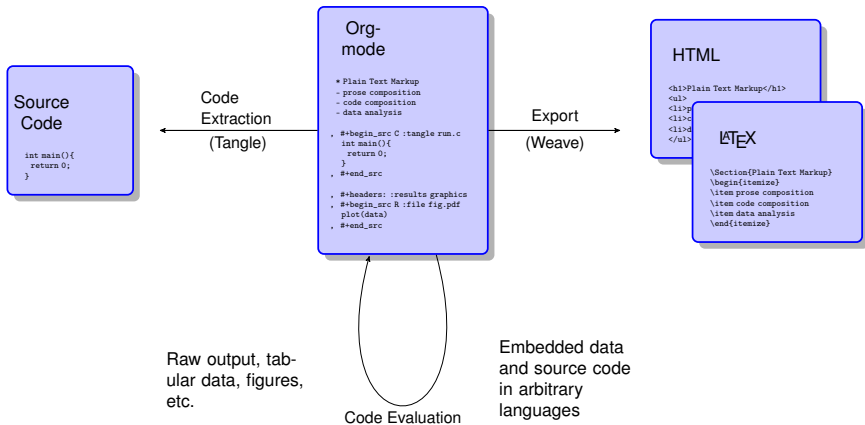
# Babel: Active code in Org mode

*Babel is about letting many different languages work together. Programming languages live in blocks inside natural language Org-mode documents. A piece of data may pass from a table to a Python code block, then maybe move on to an R code block, and finally end up embedded as a value in the middle of a paragraph or possibly pass through a gnuplot code block and end up as a plot embedded in the document.*

(Source:

<http://orgmode.org/worg/org-contrib/babel/intro.html>)

# Using code with(in) Org mode



Source: Schulte et al. (2012)

# Topic

Disclaimer and Acknowledgment

Introduction

Motivating problem

Emacs

Org mode

Babel: Active code in Org mode

**Babel and R**

Introduction

Tangling and weaving

A short example

A brief Org example

R and other languages (Python)

References

# Topic

Disclaimer and Acknowledgment

Introduction

Motivating problem

Emacs

Org mode

Babel: Active code in Org mode

**Babel and R**

Introduction

Tangling and weaving

A short example

A brief Org example

R and other languages (Python)

References

# The structure of Org code blocks

`#+NAME: <name>`

`#+BEGIN_SRC <language> <switches> <header arguments>`  
`<body>`

`#+END_SRC`

# R code blocks I (exports code and results)

## R code block

This is a short sentence. This is another sentence.  
And, finally, here comes some R code:

```
#+BEGIN_SRC R
set.seed(1)
x1 <- rnorm(100)
mean(x1)
x1.sd <- sd(x1)
#+END_SRC
```

```
#+RESULTS:
: [1] 0.1088874
```

## Evaluated and exported R code block

This is a short sentence. This is another sentence. And, finally, here comes some R code:

```
1 set.seed(1)
2 x1 <- rnorm(100)
3 mean(x1)
4 x1.sd <- sd(x1)
```

```
[1] 0.1088874
```

# R code blocks II (exports only results)

## R code block

This is a short sentence. Hey, there is another sentence. And, finally, here comes some R code:

```
#+BEGIN_SRC R :results output :exports results
set.seed(1)
x1 <- rnorm(100)
mean(x1)
x1.sd <- sd(x1)
#+END_SRC

#+RESULTS:
: [1] 0.1088874
```

## Evaluated and exported R code block

This is a short sentence. Hey, there is another sentence. And, finally, here comes some R code:

```
[1] 0.1088874
```



# R inline code blocks (source)

The mean of `x` is

```
src_R[:exports results :results value raw]{round(mean(x1), 3)}
```

and the standard error is

```
src_R[:exports results :results value raw]{round(x1.sd, 3)}.
```

Remember that the vectors `=x1=` and `=x1.sd=` have been defined on the previous slide in a separate code block.

# R inline code blocks (evaluated)

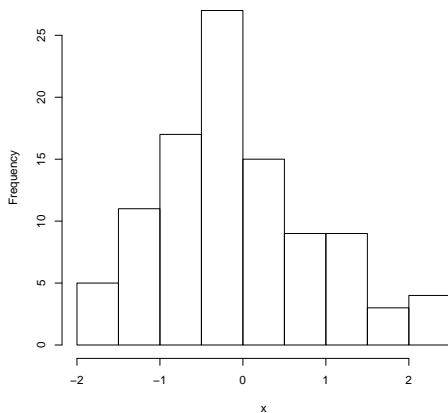
The mean of `x` is 0.109 and the standard error is 0.898.  
Remember that the vector `x1` has been defined on the previous slide in a separate code block.

# Creating a base graphics plot (source)

```
1:  #+headers: :exports both
2:  #+headers: :results graphics
3:  #+headers: :file img.pdf
4:  #+begin_src R
5:  x <- rnorm(100)
6:  hist(x)
7:  #+end_src
8:
9:  #+RESULTS:
10:  [[file:img.pdf]]
```

# Creating a base graphics plot (evaluated)

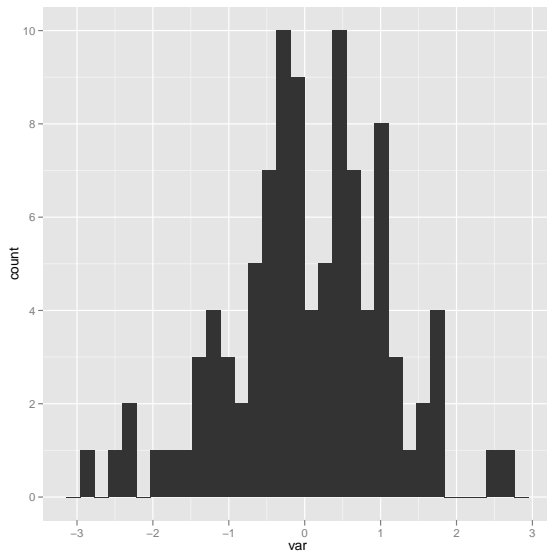
```
1 x <- rnorm(100)
2 hist(x, main = "")
```



# Creating a ggplot2 plot (Org source)

```
1:  #+headers: :exports results
2:  #+headers: :results graphics
3:  #+headers: :file f_ggplot2_ex.pdf
4:  #+begin_src R
5:  library(ggplot2)
6:  x <- data.frame(var = rnorm(100))
7:  ggplot(aes(x = var), data = x)
8:      + geom_histogram()
9:  #+end_src
10:
11:  #+RESULTS:
12:  [[file:f_ggplot2_ex.pdf]]
```

# Creating a ggplot2 plot (evaluated)



# Topic

Disclaimer and Acknowledgment

Introduction

Motivating problem

Emacs

Org mode

Babel: Active code in Org mode

**Babel and R**

Introduction

**Tangling and weaving**

A short example

A brief Org example

R and other languages (Python)

References

# Tangling and weaving code

- ▶ *Weaving* refers to the exportation of a mixed code/prose document to a format that can be read by a human (HTML,  $\text{\LaTeX}$ , Ascii, ...).
- ▶ *Tangling* means to extract only the code to a separate file

(Source: Schulte et al. 2012: 12)



# Example of code tangling

```
#+BEGIN_SRC R :tangle ../src/example_tangled.R
## Comment: Assign value 1 to variable x
x <- 1
#+END_SRC
```

# Example (cont'd): Tangled R code

Here is the content of `../src/example_tangled.R`

```
../src/example_tangled.R
```

```
## [[file:e:/projects/confer/ps2012-07-KRUG_org_r/slide/ps2012-0
```

```
## Comment: Assign value 1 to variable x
```

```
x <- 1
```

```
## Example-of-code-tangling:2 ends here
```

# Topic

Disclaimer and Acknowledgment

Introduction

Motivating problem

Emacs

Org mode

Babel: Active code in Org mode

**Babel and R**

Introduction

Tangling and weaving

**A short example**

A brief Org example

R and other languages (Python)

References

# A short example

See `pub/d_example.org` and `pub/d_example.pdf`.

# Topic

Disclaimer and Acknowledgment

Introduction

Motivating problem

Emacs

Org mode

Babel: Active code in Org mode

**Babel and R**

Introduction

Tangling and weaving

A short example

**A brief Org example**

R and other languages (Python)

References

# A brief Org example

See `pub/d_example.org` and `pub/d_example.pdf`.

# Topic

Disclaimer and Acknowledgment

Introduction

Motivating problem

Emacs

Org mode

Babel: Active code in Org mode

Babel and R

Introduction

Tangling and weaving

A short example

A brief Org example

**R and other languages (Python)**

References

# Chaining (source)

```
1 Define variable =y= in a Python code block:
2 #+name: chainexamp
3 #+begin_src python :results value :session *python*
4 y = 10**4
5 y
6 #+end_src
7
8 #+RESULTS: chainexamp
9 : 10000
10
11 Then pass variable =y= to an R code block:
12 #+BEGIN_SRC R :var x=chainexamp :session *R2*
13 print(x)
14 print(x*2)
15 #+END_SRC
16
17 #+RESULTS:
18 : [1] 10000
19 : [1] 20000
```



# Chaining (evaluated)

Define variable `y` in a Python code block:

```
1 y = 10**4  
2 y
```

10000

Then pass variable `y` to an R code block:

```
1 print(x)  
2 print(x*2)
```

[1] 10000

[1] 20000

# Topic

Disclaimer and Acknowledgment

Introduction

- Motivating problem

- Emacs

- Org mode

- Babel: Active code in Org mode

Babel and R

- Introduction

- Tangling and weaving

- A short example

- A brief Org example

R and other languages (Python)

References

# References

- ▶ Schulte, Eric, Dan Davison, Thomas Dye, und Carsten Dominik. 2012. A Multi-Language Computing Environment for Literate Programming and Reproducible Research. *Journal of Statistical Software* 46: 1–24.
- ▶ <http://orgmode.org/worg/org-contrib/babel/how-to-use-Org-Babel-for-R.html>