Adding a corporate identity to reproducible research

R Belgium, Zaventem March 7 2017

Thierry Onkelinx
Research Institute for Nature and Forest (INBO)
Summary

1. Introduction
2. ggplot2 for graphics
3. Short Markdown intro
4. Creating a corporate identity for RMarkdown
5. Tips and tricks
6. Demo

Flanders
State of the Art
Introduction
Me

- senior statistician at Research Institute for Nature and Forest (INBO)
- team Biometric and Quality Assurance
- statistical consultancy for scientists
  - getting the questions clear
  - design of experiments, sample size calculation
  - (assisting with) data analysis
  - what to report
- helpdesk for R and statistics
- training for scientists on R and statistics
- projects requiring a statistician
  - designing longterm monitoring
  - automated analysis of longterm monitoring
Quick poll

- who is using a WYSIWYG editor (Word, Powerpoint, LibreOffice, ...)?
Quick poll

- who is using a WYSIWYG editor (Word, Powerpoint, LibreOffice, ...)?
- who is using LaTeX?
Quick poll

- who is using a WYSIWYG editor (Word, Powerpoint, LibreOffice, ...)?
- who is using LaTeX?
- who is using HTML?
Quick poll

- who is using a WYSIWYG editor (Word, Powerpoint, LibreOffice, ...)?
- who is using LaTeX?
- who is using HTML?
- who is using Markdown?
Quick poll

► who is using a WYSIWYG editor (Word, Powerpoint, LibreOffice, ...)?
► who is using LaTeX?
► who is using HTML?
► who is using Markdown?
► who is using base R graphics?
Quick poll

- who is using a WYSIWYG editor (Word, Powerpoint, LibreOffice, ...)?
- who is using LaTeX?
- who is using HTML?
- who is using Markdown?
- who is using base R graphics?
- who is using lattice graphics?
Quick poll

- who is using a WYSIWYG editor (Word, Powerpoint, LibreOffice, ...)?
- who is using LaTeX?
- who is using HTML?
- who is using Markdown?
- who is using base R graphics?
- who is using lattice graphics?
- who is using ggplot2 graphics?
Quick poll

► who is using a WYSIWYG editor (Word, Powerpoint, LibreOffice, ...)?
► who is using LaTeX?
► who is using HTML?
► who is using Markdown?
► who is using base R graphics?
► who is using lattice graphics?
► who is using ggplot2 graphics?
► who is using ggvis graphics?
Quick poll

- who is using a WYSIWYG editor (Word, Powerpoint, LibreOffice, ...)?
- who is using LaTeX?
- who is using HTML?
- who is using Markdown?
- who is using base R graphics?
- who is using lattice graphics?
- who is using ggplot2 graphics?
- who is using ggvis graphics?
- who is using RStudio IDE?
Once upon a time there were only base graphics and MS Word

- copy-paste results from R into a Word document
  - tedious
  - error-prone when data or analysis changed
- graphics exported from R and imported with link to file in Word
- base graphics are tedious to lay-out
- combining references and equations in large documents was a horror story
Enter the LaTeX era

Pro

▶ no more problems with references and equations in large documents
▶ R code and text combined using Sweave
  ▶ no more tedious and error-prone copy-paste
▶ lay-out can be defined in a style

Contra

▶ requires knowledge of LaTeX syntax
  ▶ turned out to be impossible read for our clients
▶ pdf output instead of word
ggplot2 for graphics
Very short intro

- based on the grammar of graphics (Wilkinson, 2005)
- plot specification at a high level of abstraction
- very flexible
- theme system for polishing plot appearance
- mature and complete graphics system
- many users, active mailing list
Basic idea

- `ggplot()` defines the defaults for data and aesthetics
- `geom_xyz()` defines how the aesthetics are represented on the plot
- `scale_type_xyz()` defines the scale of each aesthetic
- `facet_zzz()` defines subsets in the data
- `coord_yyy()` defines coordinate systems
- `theme()` defines the general look and feel
Example

```r
library(ggplot2)
ggplot(mtcars, aes(x = wt, y = mpg, shape = factor(cyl), colour = factor(cyl))) + geom_point() + facet_grid(vs ~ am)
```
Using build-in themes

ggplot(mtcars, aes(x = wt, y = mpg, shape = factor(cyl), colour = factor(cyl))) + geom_point() + facet_grid(vs ~ am) + theme_bw(base_size = 6)
Setting a default theme

```r
theme_set(theme_classic(base_size = 6))

ggplot(mtcars, aes(x = wt, y = mpg, shape = factor(cyl), colour = factor(cyl))) + geom_point() + facet_grid(vs ~ am)
```
Definition of a theme

theme_grey

## function(base_size = 11, base_family = "") {
##   half_line <- base_size / 2
##
##   theme(
##     # Elements in this first block aren't used directly, but are inherited
##     # by others
##     line = element_line(colour = "black", size = 0.5, linetype = 1,
##       lineend = "butt"),
##     rect = element_rect(fill = "white", colour = "black",
##       size = 0.5, linetype = 1),
##     text = element_text(
##       family = base_family, face = "plain",
##       colour = "black", size = base_size,
##       lineheight = 0.9, hjust = 0.5, vjust = 0.5, angle = 0
##       margin = margin(), debug = FALSE
##     ),
##
##     axis.line = element_blank(),
##     axis.line.x = NULL,
##     axis.line.y = NULL,
##     axis.text = element_text(size = rel(0.8), colour = "grey30"),
##     axis.text.x = element_text(margin = margin(t = 0.8 * half_line / 2), vjust = 1),
##     axis.text.x.top = element_text(margin = margin(b = 0.8 * half_line / 2), vjust = 0),
##     axis.text.y = element_text(margin = margin(l = 0.8 * half_line / 2), hjust = 1),
##     axis.text.y.right = element_text(margin = margin(r = 0.8 * half_line / 2), hjust = 0),
##
##     axis.ticks = element_line(colour = "grey20"),
##     axis.ticks.length = unit(half_line / 2, "pt"),
##
##     axis.title.x = element_text(
##       margin = margin(t = half_line),
##       vjust = 1
##     ),
##     axis.title.x.top = element_text(
##       margin = margin(b = half_line),
##       vjust = 0
##     ),
##
##     axis.title.y = element_text(
##       angle = 90,
##       margin = margin(r = half_line),
##       vjust = 1
##     ),
##     axis.title.y.right = element_text(
##       angle = -90,
##       margin = margin(l = half_line),
##       vjust = 0
##     ),
##
##     legend.background = element_rect(colour = NA),
##     legend.spacing = unit(0.4, "cm"),
##     legend.spacing.x = NULL,
##     legend.spacing.y = NULL,
##     legend.margin = margin(0.2, 0.2, 0.2, 0.2, "cm"),
##     legend.key = element_rect(fill = "grey95", colour = "white"),
##     legend.key.size = unit(1.2, "lines"),
##     legend.key.height = NULL,
##     legend.key.width = NULL,
##     legend.text = element_text(size = rel(0.8)),
##     legend.text.align = NULL,
##     legend.title = element_text(hjust = 0),
##     legend.title.align = NULL,
##     legend.position = "right",
##     legend.direction = NULL,
##     legend.justification = "center",
##     legend.box = NULL,
##     legend.box.margin = margin(0, 0, 0, 0, "cm"),
##     legend.box.background = element_blank(),
##     legend.box.spacing = unit(0.4, "cm"),
##
##     panel.background = element_rect(fill = "grey92", colour = NA),
##     panel.border = element_blank(),
##     panel.grid.major = element_line(colour = "white"),
##     panel.grid.minor = element_line(colour = "white", size = 0.25),
##     panel.spacing = unit(half_line, "pt"),
##     panel.spacing.x = NULL,
##     panel.spacing.y = NULL,
##     panel.ontop = FALSE,
##
##     strip.background = element_rect(fill = "grey85", colour = NA),
##     strip.text = element_text(colour = "grey10", size = rel(0.8)),
##     strip.text.x = element_text(margin = margin(t = half_line, b = half_line)),
##     strip.text.y = element_text(angle = -90, margin = margin(l = half_line, r = half_line)),
##     strip.placement = "inside",
##     strip.placement.x = NULL,
##     strip.placement.y = NULL,
##     strip.switch.pad.grid = unit(0.1, "cm"),
##     strip.switch.pad.wrap = unit(0.1, "cm"),
##
##     plot.background = element_rect(colour = "white"),
##     plot.title = element_text(
##       size = rel(1.2),
##       hjust = 0, vjust = 1,
##       margin = margin(b = half_line * 1.2)
##     ),
##     plot.subtitle = element_text(
##       size = rel(0.9),
##       hjust = 0, vjust = 1,
##       margin = margin(b = half_line * 0.9)
##     ),
##     plot.caption = element_text(
##       size = rel(0.9),
##       hjust = 1, vjust = 1,
##       margin = margin(t = half_line * 0.9)
##     ),
##     plot.margin = margin(half_line, half_line, half_line, half_line),
##
##     complete = TRUE
##   )
## }
Custom themes: our approach

- create a custom theme starting from \texttt{theme\_grey}
- use \texttt{update\_geom\_defaults()} to update default geom colour and fill
- currently no \texttt{update\_scale\_default()}
  - solution: define a custom version of \texttt{scale\_type\_xyz()}
  - \textbf{warning}: ggplot2 must be loaded first in order to mask custom functions with same name in ggplot2
- store the functions in an R package
  - \textbf{tip}: add a vignette with lots of different figures to visualise the theme
- the package sets the relevant theme as default upon loading
- distribute the package to co-workers
  - \textbf{tip}: don’t tweak the plot until the concept is finalised
Using the default corporate identity

```r
library(INBOtheme)

ggplot(mtcars, aes(x = wt, y = mpg, shape = factor(cyl), colour = factor(cyl))) +
geom_point() + facet_grid(vs ~ am)
```

![Graph with scatter plots showing weight (wt) against miles per gallon (mpg) for different numbers of cylinders (factor(cyl)). The graph is divided into two panels based on the transmission type (vs).](image-url)
Switching to other theme in corporate identity

```r
demand_set(theme_vlaanderen2015(6, transparent = "plot"))
switchColour(vl.darkyellow) # a bunch of update_geom_default() calls
ggplot(mtcars, aes(x = wt, y = mpg, shape = factor(cyl), colour = factor(cyl))) +
  geom_point() + facet_grid(vs ~ am)
```
Short Markdown intro
Markup language

- plain text file
- markup is added as extra code in between text
- similar principle as HTML or LaTeX but simpler syntax
  - syntax is more human readable than LaTeX or HTML
  - can be mixed with LaTeX or HTML
- use pandoc to convert markdown to the required format
  - pdf (through LaTeX), HTML, Word, Epub, odt, ...
  - syntax reflects greatest common denominator
Basic syntax

- one or more blank lines separate two paragraphs
- **bold**, _italics_
- headers: line starts with one or more `#` followed by the title
  - chapter: `#` chapter title
  - section: `##` section title
  - subsection: `###` subsection title
- unnumbered list: start each item with –
- numbered lists: start each item with 1.

See the RStudio cheat sheet and reference guide on RMarkdown
YAML

- stands for YAML Ain’t Markup Language
- contains all settings for the corporate identity
- listed at the top of the file

---

```yaml
title: "Adding a corporate identity to reproducible research"
author: "Thierry Onkelinx"
bibliography: RBelgium.bib
output:
  INBOmd::inbo_slides:
    location: "R Belgium, Zaventem March 7 2017"
    institute: "Research Institute for Nature and Forest (INBO)"
    theme: "vlaanderen"
    flandersfont: TRUE
toc_name: "Summary"
cover: "1.jpg"

---
```
RMarkdown principle

- plain text formatted with Markdown + R chunks
- use `rmarkdown::render()` to render the file
  1. runs all R code of the Rmd file in a **new** session
  2. create an md file which has all chunks replaced by their output
  3. converts the resulting md file using pandoc
- `rmarkdown::render()` can render multiple output formats in one go
- RStudio: click the **knit** button
Creating a corporate identity for RMarkdown
Select output formats

- each output format requires
  - a dedicated function
  - a template
  - auxiliary files

- support for Word is limited to styles, for odt probably too
  - only useful for co-workers/clients how insist on having a word version

- we use pdf for reports and slides
  - requires a working installation of Tex

- HTML for website, epub and slides is on the to do list
Dedicated function

- defines which template to use
- handles all the arguments set in the YAML
- translates R function arguments to pandoc arguments
- sets default knitr chunk options
  - dimensions and resolution of figures
  - output width
  - ...
- combines everything into rmarkdown::output_format() which is used by rmarkdown::render()
The \LaTeX\ template

- basically a minimal source document for the required output format
- contains all elements which are generic for all documents of this template
  - replace exchangeable element with pandoc variables ($xxx$)
  - simple if-else construct are available
  - useful to convert pandoc variables into \LaTeX\ variables
- requires basic knowledge of \LaTeX

\documentclass[11pt, twoside]{extreport}
\usepackage[babel = $lang$]{inborapport_2015}
\title{$title$}
\author{$author$}
$if(reportnr)\reportnumber{$reportnr$}$endif$
\begin{document}
\maketitle
$body$
\end{document}
(LaTeX) style

- loads all required LaTeX packages
- defines the style of headings, plain text, verbatim text, ...
- defines how the title page(s) should look like
- defines the different styles of pages for beamer presentation
- requires someone willing to dive deep into the bowels of LaTeX

\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{inborapport_2015}
\def\reportnumber#1{\def\@reportnumber{#1}}
\RequirePackage[yyyymmdd, hhmmss]{datetime}
\reportnumber{\today\space currenttime}
\RequirePackage[a4paper, top = 2cm, bottom = 2.5cm, left = 3cm, right = 3cm]{geometry}
\RequirePackage[no-math]{fontspec}
\setmainfont[Ligatures=TeX]{Calibri}
\setmonofont{Courier New}[Scale = 0.67]
\def\@makechapterhead#1{\par}
   \begin{center}
   \textbf{\@reportnumber \ the chapter \ \\
\vspace{16pt}}
\end{center}
}
CSL file (optional)

- Citation Style Language
- XML file
- required in case you want a specific style for the references
- use http://editor.citationstyles.org/ to create it
RStudio template (optional)

- create a new Rmd file in RStudio from File > New File > R Markdown
- the dialog that pops up has the option From template
- this scans all installed R packages for templates
- works only if the package contains the required files in inst/rmarkdown
- the skeleton.Rmd can be the bare minimum or predefined text
  - useful for adding instructions or examples
- see http://rmarkdown.rstudio.com/developer_document_templates.html
Dedicated R package

The best way to bundle a corporate identity for Markdown is an R package

- R contains the functions
- inst contains the CSL files
- inst/pandoc contains the templates
- inst/local_tex contains a LaTeX directory structure with all the styles
- inst/rmarkdown contains RStudio menu templates

**Tip:** add a vignette which is a working example of the template.
Tips and tricks
User requirements

▶ end user
  ▶ knowledge of R and Markdown. LaTeX or HTML is a plus.
  ▶ a good editor is a must. We like RStudio.
  ▶ must have pandoc installed (included with RStudio)
  ▶ must have a working installation of Tex for pdf output
  ▶ will need good instructions on how to install and update the package in case of LaTeX styles. LaTeX styles require some additional steps after installing the R package

▶ package maintainer
  ▶ Good R skills
  ▶ Good LaTeX skills in case of pdf output
  ▶ Good HTML and CSS skills in case of HTML output
  ▶ Needs to provide support for end users
How to create a nice lay-out

- get a graphic designer to create a concept
- make sure that you have all the required details
  - font and fontsize
  - size and absolute positions of graphical elements
- convert the concept into an image and display it in the background
- don’t be afraid of trial and error
Need to write an (e)-book or a large report?

▶ consider bookdown (www.bookdown.org)
▶ output formats limited to HTML, pdf and E-book
▶ adds an extra layer over RMarkdown
  ▶ combine several Rmd files into one document
  ▶ adds support for internal references without the need for LaTeX or HTML hacks
    ▶ figures, tables, headers, equations, …
  ▶ support for custom blocks
  ▶ automatic screenshot of HTML widgets in case output format is pdf
▶ render using bookdown::render_book() or Build book in build pane of RStudio
▶ output_format argument can handle rmarkdown functions for corporate identity
Demo
References