Reproducible thesis

The current folder ('thesis') contains the reproducible thesis organised around a document 'thesis-core.Rmd' that is built using the R package 'papaja', Version 0.1.0.9997 (Aust & Barth, 2020), within the R Markdown framework (Xie et al., 2020). The APA format is enabled by the 'apa7' document class (Weiss, 2021).[1] The thesis document is organised around the document named 'thesis-core.Rmd', following the procedure laid out by Tobias Heycke for the rendering of theses (see instructions at https://rpubs.com/theycke/380678 and materials at https://osf.io/g7dbt). The chapter and the appendices are automatically knitted (i.e., rendered) and added into the thesis when the document 'thesis-core.Rmd' is knitted. The output file is named 'thesis-core.pdf'. To remove three blank pages at the beginning (by-products of the rendering process), the 'thesis-core.pdf' file was copied into another file, which was renamed as 'Pablo_Bernabeu_PhD_thesis_2022.pdf', and the three blank pages were removed.

The knitting—or compilation—was originally performed in RStudio 2021.09.1, Build 372.[2].

A LaTeX compiler is necessary to knit the document. A lightweight compiler called TinyTeX can be installed by running `install.packages('tinytex')` in the R console.

The thesis can be knitted through either of the following commands. Command (a) is based on a custom function created in the folder 'R_functions', and it is designed to avoid common knitting errors. Command (b) is a standard function of R Markdown.

(a)
```
source('R_functions/knit_deleting_service_files.R')
knit_deleting_service_files('thesis/thesis-core.Rmd')
```

(b)
```
rmarkdown::render('thesis/thesis-core.Rmd')
```

A third option is using the 'Knit' button in RStudio.

[1]For further information, the websites of these frameworks are as follows. **papaja**: https://github.com/crsh/papaja. **R Markdown**: https://rmarkdown.rstudio.com/manuscript.html. **apa7**: https://mirror.ox.ac.uk/sites/ctan.org/macros/latex/contrib/apa7/apa7.pdf.

[2]RStudio 2021.09.2+382 "Ghost Orchid" Release (fc9e217980ee9320126e33cdf334d4f4e105dc4f, 2022-01-04) for macOS. Mozilla/5.0 (Macintosh; Intel Mac OS X 12_2_1) AppleWebKit/537.36 (KHTML, like Gecko) QtWebEngine/5.12.10 Chrome/69.0.3497.128 Safari/537.36

Methods applied to run the code

The results from code are entered in the thesis in one of three ways, depending on the length of the code and the amount of running time required:

1. Code present in the `.Rmd` file, and run as the thesis is rendered: used for concise, fast code. Example from Chapter-2.Rmd:

```
maxVIF_semanticpriming = car::vif(semanticpriming_lmerTest) %>% max %>% ceiling
```

2. Code sourced from separate scripts, and run as the thesis is rendered: used for very long code. Example from Appendix C:

```
source('semanticpriming/power_analysis/semanticpriming_all_powercurves.R')
```

3. Code run separately, with only the result being presented in the thesis: used for very slow code. Example from Appendix B:

```
semanticpriming =
  read.csv('semanticpriming/data/final_dataset/semanticpriming.csv')
```

Although the reproducibility of the code can be more immediately tested when the complete code is present in the `.Rmd` file, the code from the second and third methods can also be tested by accessing the appropriate code scripts (for further information, see README.pdf in the root directory).