Generating static linguistic motion charts

Gede Primahadi Wijaya Rajeg & I Made Rajeg

5 February 2018

Table of Contents

Introduction

This document shares our R codes to generate publication-ready, static *linguistic motion charts*. The codes were initially designed to create "Figure 2" in our paper (Primahadi Wijaya R. & Rajeg, 2014) on the distributional change of nominal collocates for *hot* and *warm*. That paper was inspired by a pioneering study by Martin Hilpert (cf. Hilpert, 2011), who is the first to introduce motion charts to diachronic linguistics as tools for visualising the dynamic process of language change. The static version of the charts represents sequentially ordered bubble/scatter plots (Figure 1) of certain linguistic phenomena across different diachronic corpus periods on the basis of the *Corpus of Historical American English* (COHA) (Davies, 2012). We learnt the R function (i.e., the base R `symbols()` (cf., 2.1)) to generate the bubble(/scatter) plot in our paper from section 11.1.4 in Kabacoff's (2011, pp. 278–279) book about the "*bubble plot*".

There have been two accessible tutorials on generating linguistic motion charts. One is from Martin Hilpert himself via his *YouTube* demo and tutorials to generate the *dynamic/interactive* version of the motion charts using the googleVis R package (Gesmann & de Castillo, 2011). The second one is by Desagulier (2016), who demonstrates how one can retrieve the data from COHA and wrangle it in R into the required input format for the `gvisMotionChart()` function in the `googleVis` package. The present document aims to complement these with our R codes to create the *static* version of the motion charts.

We present here two flavours of the codes: (i) the base R (with minor tweak from the 2014 version used in our paper [and in Pichler (2016)]) (2.1); and (ii) the `ggplot2` version (2.2). We first illustrate the codes using our `hotwarm.RData` available for download. Then, we re-

use the (much simpler) `ggplot2` codes to reproduce the static motion chart in Hilpert's (2011, p. 455, Figure 8) case study 2 on the English complement-taking predicates, testing the potential reproducibility of the `ggplot2` version (2.3) (but, readers are encouraged to play around with the base R version).

Main steps

Once the `hotwarm.RData` file has been downloaded, the easiest way to load the data is by double-clicking the file to open an R session. Alternatively, one may load it from an R console using `load()`, provided that the `.RData` is stored in the same working directory with the current R session.

```
load(file = "hotwarm.RData")
```

The R workspace consists of a data frame called `hotwarm` with the following format.

```
head(hotwarm)
```

decade	word	hot	warm	freq	diff
1860	admirer	0.0000000	0.4087567	0.4087567	11.0
1860	affection	0.0000000	0.9343010	0.9343010	25.2
1860	afternoon	0.2335753	0.2919691	0.5255443	1.0
1860	air	0.6423319	1.0510886	1.6934206	1.3
1860	attachment	0.0000000	0.4087567	0.4087567	11.0
1860	bed	0.0583938	0.5255443	0.5839381	7.1

The `decade` represents the COHA decades. In the paper, we track the nominal collocational change of *hot* and *warm* across fifteen decades, from the 1860s to the 2000s. The `hot` and `warm` columns show the normalised co-occurrence frequencies per million words of the nouns (in the `word` column) with each *hot* and *warm* respectively in the [ADJ + NOUN] schema. In other words, we retrieved the R1 collocates of the two adjectives from COHA. The `freq` column contains the joint frequency of the nouns in the specified schema across *hot* and *warm* (i.e., the sum of values in `hot` and `warm`). Finally, the `diff` column uses the values from the "SCORE" column produced by the "compare" feature of COHA (this choice follows the suggestion in Martin Hilpert's tutorial).

Base R version of the static motion chart

The steps for generating the static chart with the base R are described below.

(1) Generate character vectors for the labels of each decade.

```
decade.label <- as.character(unique(hotwarm$decade))
decade.label
```

```
##  [1] "1860" "1870" "1880" "1890" "1900" "1910" "1920" "1930" "1940" "1950"
## [11] "1960" "1970" "1980" "1990" "2000"
```

(2) Create a character vector containing collocates whose distribution with *hot* and *warm* will be traced through. This vector will be used as the selected labels for the bubbles in the scatter plots.

```
selected.collocs <- c("bath", "day", "dog",
                      "heart", "pursuit", "smile",
                      "spot", "water", "welcome")
```

(3) Subset the data points for the selected collocate-labels defined in step (2) from the `hotwarm` data frame.

```
dframe.label <- subset(hotwarm, word %in% selected.collocs)
head(dframe.label)
```

	decade	word	hot	warm	freq	diff
16	1860	day	0.9926948	1.3430577	2.3357525	1.1
26	1860	heart	0.0583938	2.3941463	2.4525401	32.2
32	1860	pursuit	0.8175134	0.0000000	0.8175134	35.6
45	1860	water	5.8977751	1.1094824	7.0072575	6.8
47	1860	welcome	0.0000000	0.5255443	0.5255443	14.2
54	1870	bath	0.2149364	1.5582886	1.7732249	5.6

(4) Split the `hotwarm` data frame into a list of subset data frame by decades.

```
# data frame for all data points
df.main <- split(hotwarm, hotwarm$decade)

# check the type of the data and the length
typeof(df.main)

## [1] "list"

length(df.main)

## [1] 15
```

(5) Split also the `dframe.label` produced in (3) above into a list of data frame per decade, containing the data for the collocates labels.

```
# data frame for the selected collocates
df.labs <- split(dframe.label, dframe.label$decade)
```

(6) Now comes the `for` loop to generate the scatter plots for each of the fifteen decades. Readers might do some adjustment from the codes below, such as the limits, or tick-labels, of the *x*- (`xlim`) and *y*-axis (`ylim`), given the nature of their data. Before running the `for` loop, we need to define a number of graphical parameters for the ploting (with `par()`, cf. 6.1 in the chunk below), especially for the number of row and column in the plot. The `hotwarm` data consists of fifteen decades. So we divide the plot into 5-by-3 matrix (hence, `mfrow = c(5, 3)`). Details for the list of arguments in the `par()` call below can be reviewed by typing `?par()` in the console.

```r
# 6.1 Define plotting parameter---------
par(mfrow = c(5, 3), # set a 5-by-3 plotting space
    cex = 0.425,
    mar = c(2.5, 2, 1, 1),
    mgp = c(3, 0.75, 0),
    tck = -0.025,
    las = 1)

# 6.2 "for" loop to generate the sequential scatterplots---------
for (i in seq_along(decade.label)) {
# loop sequence is derived from the total number of (15) decades

  # preparing empty plot field from the dataset of all decades
    plot(df.main[[i]][,3], # column [,3] for the x-axis
         df.main[[i]][,4], # column [,4] for the y-axis
         type = "n",
         xlim = c(-0.5, max(hotwarm$hot)+1),
         ylim = c(-0.5, max(hotwarm$warm)+0.5),
         yaxt = "n",
         xaxt = "n",
         ann = F)

  # adding major y-axis to the empty plot
    axis(side = 2, at = c(0, 1, 2, 3), labels = c(0, 1, 2, 3))

  # (OPTIONAL) adding minor y-axis to the empty plot
    #axis(side = 2, at = c(1, 3), labels = FALSE, tcl = -0.2)

  # adding major x-axis to the empty plot
    axis(side = 1,
         at = c(0, 2, 4, 6, 8),
         labels = c(0, 2, 4, 6, 8),
         tick = TRUE)

  # (OPTIONAL) adding minor x-axis to the empty plot
    #axis(side = 1, at = c(1, 3, 5, 7), labels = FALSE, tcl = -0.2)

  # adding gridlines onto the empty plot field
    grid(lwd = 1.15)

  # adding the decade labels onto the plot
    text(4, 3, labels = decade.label[i],
         col = "grey85", cex = 4.15, font = 2)

  # adding the bubble symbol onto the plot from all data points
    # (cf. Kabacoff, 2011, pp. 278-279)
    symbols(df.main[[i]][,3],
            df.main[[i]][,4],
            inches = 0.150,
```

```
          fg = "white",
          bg = "grey85",
          # below, column [,5] ("freq") for bubble size
          circles = sqrt(df.main[[i]][,5]/pi),
          add = TRUE)

# adding the bubble symbol onto the plot
  # darker colour for the labelled data points
  symbols(df.labs[[i]][,3],
          df.labs[[i]][,4],
          inches = 0.150,
          fg = "white",
          bg = "grey65",
          circles = sqrt(df.labs[[i]][,5]/pi),
          add = TRUE)

# adding the selected collocate labels onto the plot
  text(df.labs[[i]][,3],
       df.labs[[i]][,4],
       labels = df.labs[[i]][,2],
       cex = 1.25)

  } # end of "for-loop"
```

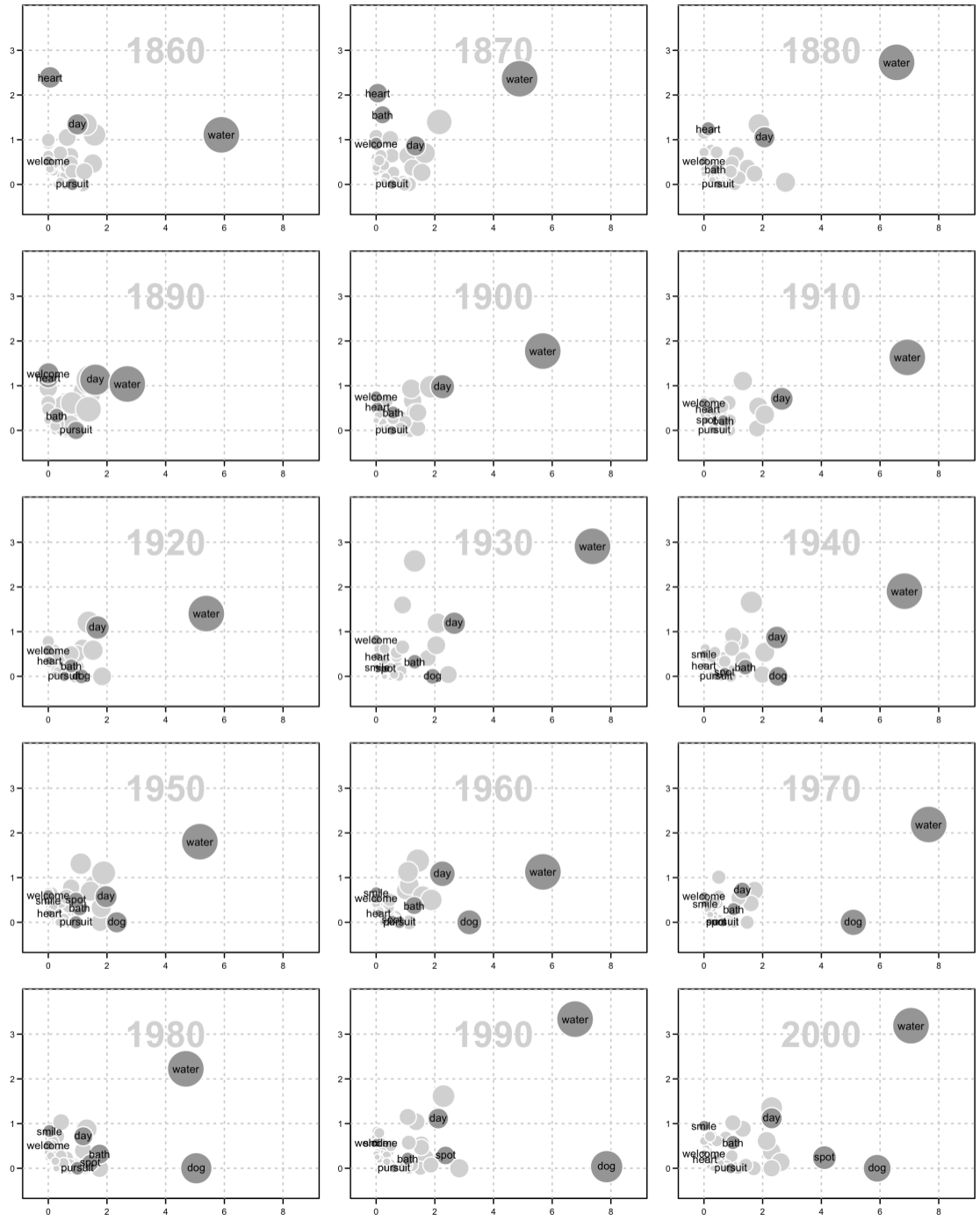

Figure 1 Changes in the collocational profiles of hot (x-axis) and warm (y-axis), COHA 1860s-2000s (Primahadi Wijaya R. & Rajeg, 2014, p. 253).

In order to save the above plot, first set up a `.png` file (or other formats, such as `.jpeg` or `.pdf`) for the resulting plot, before executing the above codes, and define the plot's resolution and sizes. After that, run the previous codes in `6.1` and `6.2` (i.e., step (6)), and end with `dev.off()` to close the graphic device and save the plot. See the code-chunk below.

```
# set up a .png file for the resulting plot
# including the plot's resolution and sizes
png("hotwarmPLOT-loop.png",
    res = 450,
    width = 6.5 * 500,
    height = 8.5 * 500)

# Insert here (and then run):
# (i) the plotting parameter call (6.1) and
# (ii) the for loop (6.2)

# close the active graphic device to save the plot
dev.off ()
```

Note that to achieve the desired results in terms of the size and resolution of the plot, a couple of trial-and-errors should be expected.

The ggplot2 version of the static motion chart

The ggplot2 package offers a *facetting* feature that is the key to a (much) simpler code for producing the static motion charts. In ggplot2, we can use the facet_wrap() function, which can be directly fed with the decade column and let ggplot2 do the job to split the plots by decades. To run the following codes, readers need to either install ggplot2 package separately or together as a part of a suite of packages called the tidyverse (Wickham & Grolemund, 2017). Optionally, readers might also want to install the ggrepel package that is used here to handle the overlapping labels.

Here are the steps for the ggplot2 version.

1. Prepare two data frames: (i) one for the selected collocate/item labels (df.label), and (ii) one for the remaining items (df.others), which will not be labelled in the plot. Here, we also generate the axes labels.

```
# get the data for the selected collocates
df.label <- subset(hotwarm, word %in% selected.collocs)

# get the data for the remaining collocates
df.others <- subset(hotwarm, !word %in% selected.collocs)

# x-axis labels
xaxis <- expression(paste("Co-occurrence frequency per million words with ",
                          italic("hot"),
                          sep = ""))

# y-axis labels
yaxis <- expression(paste("Co-occurrence frequency per million words with ",
                          italic("warm"),
                          sep = ""))
```

2. Generate the first ggplot() call for the non-labelled data and save the output into p.

```r
# load the ggplot2 via tidyverse (or individually) and load the ggrepel
library(ggplot2)
library(ggrepel)

# generate plot for the non-labelled data
p <- ggplot(data = df.others,
            aes(x = hot, y = warm)) +
  geom_point(aes(size = freq),
             alpha = 1/8,
             show.legend = FALSE) +
  facet_wrap(~decade) # here is the key to split the plot by decades!
```

3. Add the labelled data points to the previous plot (`p`) on the basis of the `df.label` data frame.

```r
p <- p + geom_point(data = df.label,
                aes(size = freq),
                alpha = 1/2,
                colour = "royalblue",
                show.legend = FALSE) +
  # if you are not using `ggrepel`, use `geom_text()`
  geom_text_repel(data = df.label,
                  aes(label = word),
                  size = 2.5) +
  labs(x = xaxis, # insert the `xaxis` data
       y = yaxis) + # insert the `yaxis` data
  theme_bw()

# call out the plot
p
```

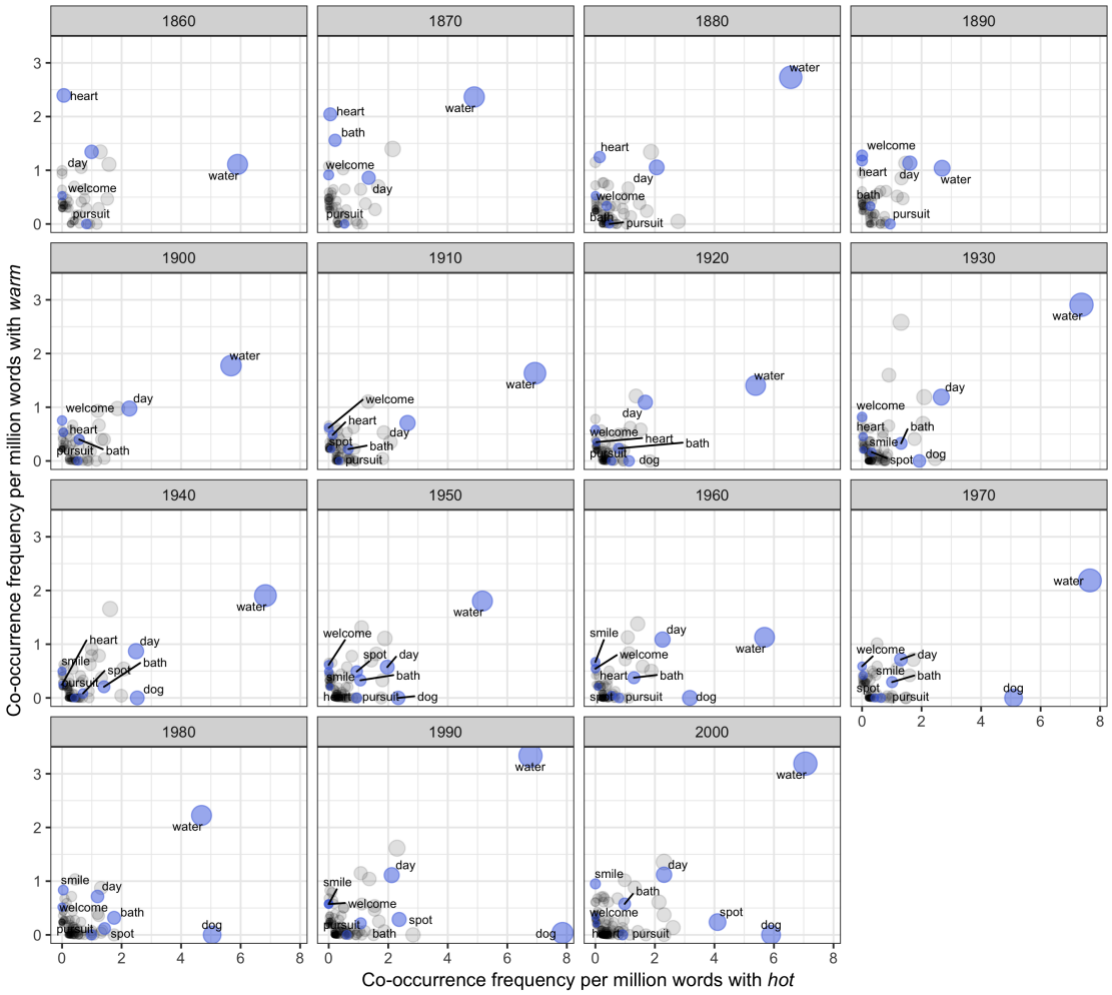

Figure 2 Changes in the nominal collocational profiles of hot and warm, COHA 1860s-2000s (Primahadi Wijaya R. & Rajeg, 2014, p. 253).

4. Save the plot as `.png` file using `ggsave()`.

```
ggsave(file = "hotwarm.png", plot = p,
       width = 20, height = 20, unit = "cm", dpi = 600)
```

Note that trial and errors are expected to get the desired results in terms of the size and resolution. For the previous plot, the parameters for `width`, `height`, and resolution (i.e., `dpi`) given above produce a desirable result from our own perspective.

Recreating previous motion chart in Hilpert (2011)

To recreate the chart (i.e., "Figure 8") in Hilpert's (2011) paper, we need to download the data from Martin Hilpert's motion charts resource page. The data is available as zip file, which contains the `mot.RData` workspace. After the downloads, open/load the R workspace that contains two data frames: (i) `convdata` and (ii) `compdata`. The former is the data for Hilpert's (2011, p. 444) case study 1 on the "ambicategorical nouns and verbs". We will

instead use the `compdata` to produce the static motion charts for the English complement-taking predicates (readers can try the codes for themselves with the `convdata`).

```
load("mot.RData")
head(compdata)
```

x	y	decade	verb	freq
-0.2937999	0.1350332	1860	acknowledge	12.583587
-0.3892971	0.2318209	1860	admit	34.346505
-0.5485611	0.3116948	1860	affirm	8.206687
0.0154595	-0.6818823	1860	appreciate	1.519757
0.0874103	-0.7591102	1860	await	5.957447
-0.2872019	0.2074660	1860	believe	178.115501

The *x*- and *y*-axis represent the dimensions of the metric Multidimensional Scaling (MDS) from the (dis)similarity measures between the verbs on the basis of their complementation profiles (Hilpert, 2011, pp. 451–452, 454). The `freq` column indicates the normalised text frequencies of the verbs (Hilpert, 2011, p. 452). The steps to generate the `ggplot2` motion charts from the `compdata` data are as follows.

1. As above, define two data frames, one for the labelled items and one for the remainder of the data points. We also store the labels for the axes (cf., Hilpert, 2011, p. 453, for the interpretation given on the *x*- and *y*-axes).

```
# selected complement-taking predicates
predicates <- c("confirm", "await",
                "enjoy", "demand",
                "consider", "remember",
                "hope", "dislike",
                "want", "try")

# get the data for the selected predicates
df.label <- subset(compdata, verb %in% predicates)

# get the data for the remaining predicates
df.others <- subset(compdata, !verb %in% predicates)

# xaxis label
xaxis <- expression(paste(italic("that"),
                          "-clauses <----------    ----------> ",
                          italic("to"), "-infinitives",
                          sep = ""))

# yaxis label
yaxis <- "nominal compl. <----------    ----------> complex verbal/clausal
compl."
```

2. Generate the first `ggplot()` call for the non-labelled data and save the output into `p`.

```
# generate plot for the non-labelled data
p <- ggplot(data = df.others,
            aes(x = x, y = y)) +
  geom_point(aes(size = freq),
             alpha = 1/8,
             show.legend = FALSE) +
  facet_wrap(~decade) # here is the key to split the plot by decades!
```

3. Add the labelled data points to the previous plot (`p`) on the basis of the `df.label` data frame.

```
p <- p + geom_point(data = df.label,
                aes(size = freq),
                alpha = 1/2,
                colour = "royalblue",
                show.legend = FALSE) +
  # if you are not using `ggrepel`, use `geom_text()`
  geom_text_repel(data = df.label,
                  aes(label = verb),
                  size = 2.5) +
  labs(x = xaxis, # insert the `xaxis` data
       y = yaxis) + # insert the `yaxis` data
  theme_bw()

# call out the plot
p
```

Figure 3 Two-dimensional MDS solution for the English complement-taking predicates, COHA 1860s-2000s (Hilpert, 2011, p. 455, Figure 8).

We can save the plot using the `ggsave()` function following the code in section 2.2 with the `hotwarm` data.

Coda

We hope this document can be useful for those interested in creating other motion charts, especially the static ones. The `.Rmd` source file of this document, including the `.bib` and `csl` files, are available in our *figshare* accounts (cf. here and/or here). This MS Word document is rendered with Xie's (2016) `bookdown` R package in RStudio. For any feedback and comments, please drop us email at primahadiwijaya@gmail.com. We would be happy to hear from you!

```
# session info for this R document
# just updating to R 3.4.3 and the codes still work!
> devtools::session_info()
```

```
Session info -------------------------------------------------------------
--
 setting  value
 version  R version 3.4.3 (2017-11-30)
 system   x86_64, darwin15.6.0
 ui       RStudio (1.1.383)
 language (EN)
 collate  en_US.UTF-8
 tz       Australia/Melbourne
 date     2018-02-09

Packages -----------------------------------------------------------------
--
 package    * version date       source
 base       * 3.4.3   2017-12-07 local
 colorspace   1.3-2   2016-12-14 CRAN (R 3.4.0)
 compiler     3.4.3   2017-12-07 local
 datasets   * 3.4.3   2017-12-07 local
 devtools     1.13.4  2017-11-09 CRAN (R 3.4.2)
 digest       0.6.15  2018-01-28 CRAN (R 3.4.3)
 ggplot2    * 2.2.1   2016-12-30 CRAN (R 3.4.0)
 ggrepel    * 0.7.0   2017-09-29 CRAN (R 3.4.2)
 graphics   * 3.4.3   2017-12-07 local
 grDevices  * 3.4.3   2017-12-07 local
 grid         3.4.3   2017-12-07 local
 gtable       0.2.0   2016-02-26 CRAN (R 3.4.0)
 knitr        1.19    2018-01-29 CRAN (R 3.4.3)
 labeling     0.3     2014-08-23 CRAN (R 3.4.0)
 lazyeval     0.2.1   2017-10-29 CRAN (R 3.4.2)
 memoise      1.1.0   2017-04-21 CRAN (R 3.4.0)
 methods    * 3.4.3   2017-12-07 local
 munsell      0.4.3   2016-02-13 CRAN (R 3.4.0)
 pillar       1.1.0   2018-01-14 CRAN (R 3.4.3)
 plyr         1.8.4   2016-06-08 CRAN (R 3.4.0)
 Rcpp         0.12.15 2018-01-20 CRAN (R 3.4.3)
 rlang        0.1.6   2017-12-21 CRAN (R 3.4.3)
 scales       0.5.0   2017-08-24 CRAN (R 3.4.1)
 stats      * 3.4.3   2017-12-07 local
 tibble       1.4.2   2018-01-22 CRAN (R 3.4.3)
 tools        3.4.3   2017-12-07 local
 utils      * 3.4.3   2017-12-07 local
 withr        2.1.1   2017-12-19 CRAN (R 3.4.3)
 yaml         2.1.16  2017-12-12 CRAN (R 3.4.3)
```

References

Davies, M. (2012). Expanding horizons in historical linguistics with the 400-million word Corpus of Historical American English. *Corpora*, *7*(2), 121–157.

Desagulier, G. (2016). A motion chart of Captain Kirk's split infinitive with R and googleVis. Retrieved January 31, 2018, from http://rstudio-pubs-static.s3.amazonaws.com/153634_658c5b5686cd43e591c4e31c0437f724.html

Gesmann, M., & de Castillo, D. (2011). GoogleVis: Interface between r and the google visualisation api. *The R Journal*, *3*(2), 40–44. Retrieved from https://journal.r-project.org/archive/2011-2/RJournal_2011-2_Gesmann+de~Castillo.pdf

Hilpert, M. (2011). Dynamic visualizations of language change. Motion charts on the basis of bivariate and multivariate data from diachronic corpora. *International Journal of Corpus Linguistics*, *16*(4), 435–461.

Kabacoff, R. I. (2011). *R in Action. Data analysis and graphics with R*. Shelter Island, New York: Manning.

Pichler, K. (2016). *A diachronic perspective on synonymy* (Diploma Thesis). University of Vienna, Vienna, Austria. Retrieved from http://othes.univie.ac.at/40365/

Primahadi Wijaya R., G., & Rajeg, I. M. (2014). Visualising diachronic change in the collocational profiles of lexical near-synonyms. In I. N. Sudipa & G. Primahadi Wijaya R. (Eds.), *Cahaya Bahasa: A festschrift in honour of Prof. I Gusti Made Sutjaja* (pp. 247–258). Denpasar: Swasta Nulus. Retrieved from http://doi.org/10.4225/03/564A9F35BEDB4

Wickham, H., & Grolemund, G. (2017). *R for Data Science*. Canada: O'Reilly. Retrieved from http://r4ds.had.co.nz/

Xie, Y. (2016). *Bookdown: Authoring Books and Technical Documents with R Markdown*. Chapman and Hall/CRC.