

Datenvisualisierung 3

Darstellung der zusammenfassenden Statistik

Daniela Palleschi

Humboldt-Universität zu Berlin

2024-01-24

Readings

- [Kapital 27 \(A field guide to base R\)](#) in Wickham et al. (2023)
- course website: [Ch. 12: base R](#)

Learning objectives

Today we will...

- learn what base R is
- compare base R and tidyverse
- learn base R equivalents of tidyverse verbs

base R

- basic software containing the R programming language
 - contains the **base** package which is required to run R
- includes several packages such as **utils** and **stats** (among others)
 - installed when you install R

tidyverse

- the [Tidyverse](#) ([Wickham et al., 2019a](#)) is a family of R-packages designed to facilitate cleaning and wrangling data
 - tidyverse packages “haben eine gemeinsame Designphilosophie und eine gemeinsame Grammatik und Datenstruktur, so dass das Erlernen eines Pakets das Erlernen des nächsten erleichtert.” ([Wickham et al., 2019b](#)). - tidyverse was written in the R programming language

base R vs. tidyverse

- main goal of base R is stability
 - not many or often changes to functionality of functions
- the tidyverse is constantly adding, updating, and changing functions
- this means that tidyverse code is prone to “breaking”: tidyverse code that runs today might not run in a few years if some functions or arguments have been “deprecated”

Controversy

- some people strongly prefer to use base R or the tidyverse
 - arguments for tidyverse: more human-readable, tidier, simpler for non-programmers
 - arguments for base R: “truer” R-programming, more stable
- generally, having a strong background in one and at least a basic literacy of the other is wise

Twitter debates

Christopher Zorn · 9. Jan. 2023
@prisonrodeo · [Folgen](#)
Good morning.

If the only things you've ever done with R rely on the "tidyverse," you don't know R, and can't claim to.

Be sure your students know this.

Bodo Winter
@BodoWinter · [Folgen](#)

What a thing to say when modern R is pretty much synonymous with the tidyverse for many in the community!

I was a base R masochist once too.. but there's no need for statements like this when the tidyverse has helped so many of us be more productive and write more readable code.

7:40 nachm. · 10. Jan. 2023

63 Antworten Link kopier.

[Auf X weiterlesen](#)

- in this Tweet we see the original post from Prof. Zorn stating that knowing the tidyverse does not equate knowing R
 - but there were many replies highlighting the benefits of the tidyverse
 - from instructors, professors (like Bodo Winter, who wrote a statistics books for linguists using R ([Winter, 2019](#))), and data scientists working in industry

My preference

- I obviously think learning the tidyverse is important
 - the tidyverse is human centred, and we are not programmers or computer scientists
- not everybody agrees with me, but there are also a lot of people who do

Set-up

```
1 pacman::p_load(  
2   tidyverse,  
3   here  
4 )
```

Read in data

- we'll now see our first comparison of tidyverse code versus base R code

CSV: tidyverse

```
tidyverse: readr::read_csv()
```

```
1 df_tidy <-  
2   read_csv(  
3     here("daten", "languageR_english.csv")  
4   )
```

CSV: base R

```
base R: read.csv()  
1 df_base <-  
2   read.csv(  
3     here("daten", "languageR_english.csv")  
4   )
```

Comparing outcome

- how many columns?

1 length(df_tidy)
1 length(df_base)

[1] 7
[1] 7

- what are the column names?

1 names(df_base)
1 names(df_tidy)

[1] "AgeSubject"	"Word"	
"LengthInLetters"	"WrittenFrequency"	
[5] "WordCategory"	"RTlexdec"	"RTnaming"
[1] "AgeSubject"	"Word"	
"LengthInLetters"	"WrittenFrequency"	
[5] "WordCategory"	"RTlexdec"	"RTnaming"

- how many rows?

1 nrow(df_tidy)
1 nrow(df_base)

[1] 4568
[1] 4568

- the data structure is identical

Wrangling columns and rows

- we'll see base R alternatives to our most common **dplyr** verbs

Extract variables: tidyverse

```
tidyverse: dplyr::select()
1 df_tidy |>
2   select(AgeSubject)

# A tibble: 10 × 1
  AgeSubject
  <chr>
1 young
2 young
3 young
4 young
5 young
6 young
7 young
8 young
9 young
10 young
```


Extract variables: base R

- the dollar sign (\$) can be used to extract a column from a dataframe (or tibble)
- this will give us a vector, whereas `dplyr::select()` preserves the dataframe/tibble attributes of the column

```
base R: dataframe$column_name
1 df_base$AgeSubject
[1] "young" "young" "young" "young" "young" "young" "young" "young" "young"
[10] "young" "young" "young" "young" "young" "young" "young" "young" "young"
```

Extract variables: base R

- or we can use `dataframe[row, column]`
- we can use the name of a column in quotation marks

```
base R: dataframe[, 'column_name']
```

```
1 # using variable name  
2 df_base[, "AgeSubject"]
```

```
[1] "young" "young" "young" "young" "young" "young" "young" "young" "young"  
[10] "young" "young" "young" "young" "young" "young" "young" "young" "young"
```

- or we can give the index of the column, where 1 means the first column, 2 means the second column, and so on

```
base R: dataframe[, index]
```

```
1 # using variable index  
2 df_base[, 1]
```

```
[1] "young" "young" "young" "young" "young" "young" "young" "young" "young"  
[10] "young" "young" "young" "young" "young" "young" "young" "young" "young"
```

Multiple variables: tidyverse

```
tidyverse: dplyr::select()
1 df_tidy |>
2   select(AgeSubject, RTlexdec)

# A tibble: 10 × 2
  AgeSubject RTlexdec
  <chr>      <dbl>
1 young      695.
2 young      600.
3 young      547.
4 young      617.
5 young      633.
6 young      687.
7 young      584.
8 young      527.
9 young      741.
10 young      536.
```

Multiple variables: baseR

- for this we need `c()`

```
base R: dataframe[,c('column_name1','column_name2')]
```

```
1 # using variable name
2 df_base[,c("AgeSubject", "RTlexdec")]
```

	AgeSubject	RTlexdec
1	young	694.89
2	young	600.40
3	young	547.27
4	young	616.60
5	young	633.08
6	young	686.75
7	young	584.40
8	young	526.82
9	young	741.48
10	young	536.38

```
base R: dataframe[,c(index1,index2)]
```

```
1 # using variable index
2 df_base[,c(1, 6)]
```

	AgeSubject	RTlexdec
1	young	694.89
2	young	600.40
3	young	547.27
4	young	616.60
5	young	633.08
6	young	686.75
7	young	584.40
8	young	526.82
9	young	741.48
10	young	536.38

Extract/Filter observations: tidyverse

- with the `filter()` function from `dplyr`

```
tidyverse: dplyr::filter()

1 df_tidy |>
2   filter(RTlexdec > 600 & RTnaming < 480)

# A tibble: 856 × 7
  AgeSubject Word LengthInLetters WrittenFrequency WordCategory RTlexdec
  <chr>      <chr>          <dbl>          <dbl> <chr>          <dbl>
1 young    doe             3             3.91 N             695.
2 young    pork             4             5.02 N             617.
3 young    prop             4             4.77 N             687.
4 young    arc              3             4.89 N             741.
5 young    tile             4             4.08 N             647.
6 young    slope            5             5.80 N             633.
7 young    pith             4             2.48 N             696.
8 young    blitz            5             4.19 N             672.
9 young    port             4             6.08 N             683.
10 young    plan             4             7.46 N             636.
# i 846 more rows
# i 1 more variable: RTnaming <dbl>
```

Extract/Filter observations: base R

- add these conditional statements into `[,]`
 - we need to include the dataframe name with the dollar sign preceding the column name

```
base R: dataframe[,]  
1 df_base[df_base$RTlexdec > 600 & df_base$RTnaming < 480,]  
  
  AgeSubject Word LengthInLetters WrittenFrequency WordCategory RTlexdec  
1      young  doe                3          3.912023             N    694.89  
4      young  pork                4          5.017280             N    616.60  
6      young  prop                4          4.770685             N    686.75  
9      young  arc                 3          4.890349             N    741.48  
17     young  tile                4          4.077537             N    647.07  
18     young  slope               5          5.802118             N    632.54  
22     young  pith                4          2.484907             N    695.86  
26     young  blitz               5          4.189655             N    671.59  
29     young  port                4          6.084499             N    683.36  
34     young  plan                4          7.462789             N    636.10  
  
  RTnaming  
1      466.4  
4      460.3  
6      477.1  
~      ~ ~
```

Select single data points: tidyverse

- use `filter()` and `select()` (which we've already done before)

```
tidyverse: filter() |> select()  
1 df_tidy |>  
2   filter(RTlexdec > 600, RTnaming < 480) |>  
3   select(AgeSubject, RTlexdec)
```

```
# A tibble: 10 × 2  
  AgeSubject RTlexdec  
  <chr>      <dbl>  
1 young      695.  
2 young      617.  
3 young      687.  
4 young      741.  
5 young      647.  
6 young      633.  
7 young      696.  
8 young      672.  
9 young      683.  
10 young      636.
```

Select single data points: base R

- combine row and column values in `[,]`

```
base R: dataframe[conditions,columns]
```

```
1 df_base[df_base$RTlexdec > 600 & df_base$RTnaming < 480, c("AgeSubject", "RTlexdec")]
```

	AgeSubject	RTlexdec
1	young	694.89
4	young	616.60
6	young	686.75
9	young	741.48
17	young	647.07
18	young	632.54
22	young	695.86
26	young	671.59
29	young	683.36
34	young	636.10

Select single data points: base R

- again, you can replace the column names with the index value

```
base R: dataframe[conditions,index]
```

```
1 df_base[df_base$RTlexdec > 600 & df_base$RTnaming < 480,c(1, 6)]
```

	AgeSubject	RTlexdec
1	young	694.89
4	young	616.60
6	young	686.75
9	young	741.48
17	young	647.07
18	young	632.54
22	young	695.86
26	young	671.59
29	young	683.36
34	young	636.10

Create new variables: tidyverse

- with the `mutate()` function from `dplyr`

```
tidyverse: dplyr::mutate()

1 df_tidy |>
2   mutate(rt_lexdec_s = RTlexdec/1000)

# A tibble: 4,568 × 8
  AgeSubject Word      LengthInLetters WrittenFrequency WordCategory RTlexdec
  <chr>      <chr>          <dbl>          <dbl> <chr>          <dbl>
1 young     doe              3           3.91 N           695.
2 young     whore             5           4.52 N           600.
3 young     stress            6           6.51 N           547.
4 young     pork              4           5.02 N           617.
5 young     plug              4           4.89 N           633.
6 young     prop              4           4.77 N           687.
7 young     dawn              4           6.38 N           584.
8 young     dog               3           7.16 N           527.
9 young     arc               3           4.89 N           741.
10 young    skirt             5           5.93 N           536.
# i 4,558 more rows
# i 2 more variables: RTnaming <dbl>, rt_lexdec_s <dbl>
```

Create new variables: tidyverse

- define the new variable name (with `dataframe$variable`) and assign the value with the assignment operator `<-`

base R

```
1 df_base$rt_lexdec_s <- df_base$RTlexdec/1000
```

Summarise: tidyverse

- `summarise()` from `dplyr`

```
tidyverse: dplyr::summarise()

1 df_tidy |>
2 summarise(
3   mean_lexdec = mean(RTlexdec),
4   sd_lexdec = sd(RTlexdec),
5   mean_naming = mean(RTnaming, na.rm = T),
6   sd_naming = sd(RTnaming, na.rm = T)
7 )

# A tibble: 1 × 4
  mean_lexdec sd_lexdec mean_naming sd_naming
  <dbl>      <dbl>      <dbl>      <dbl>
1      708.      115.      566.      101.
```

Summarise: tidyverse

- we have to create new objects containing the value of each operation + combine them into a data frame using the `data.frame()` function
- there are many alternative ways to do this, but this is the simplest if we only want to produce a few summary statistics

```
base R
1 data.frame(mean_lexdec = mean(df_base$RTlexdec),
2             sd_lexdec = sd(df_base$RTlexdec),
3             mean_naming = mean(df_base$RTnaming, na.rm = T),
4             sd_naming = sd(df_base$RTnaming, na.rm = T))
  mean_lexdec sd_lexdec mean_naming sd_naming
1    708.1336  114.8599    565.9233  100.8153
```

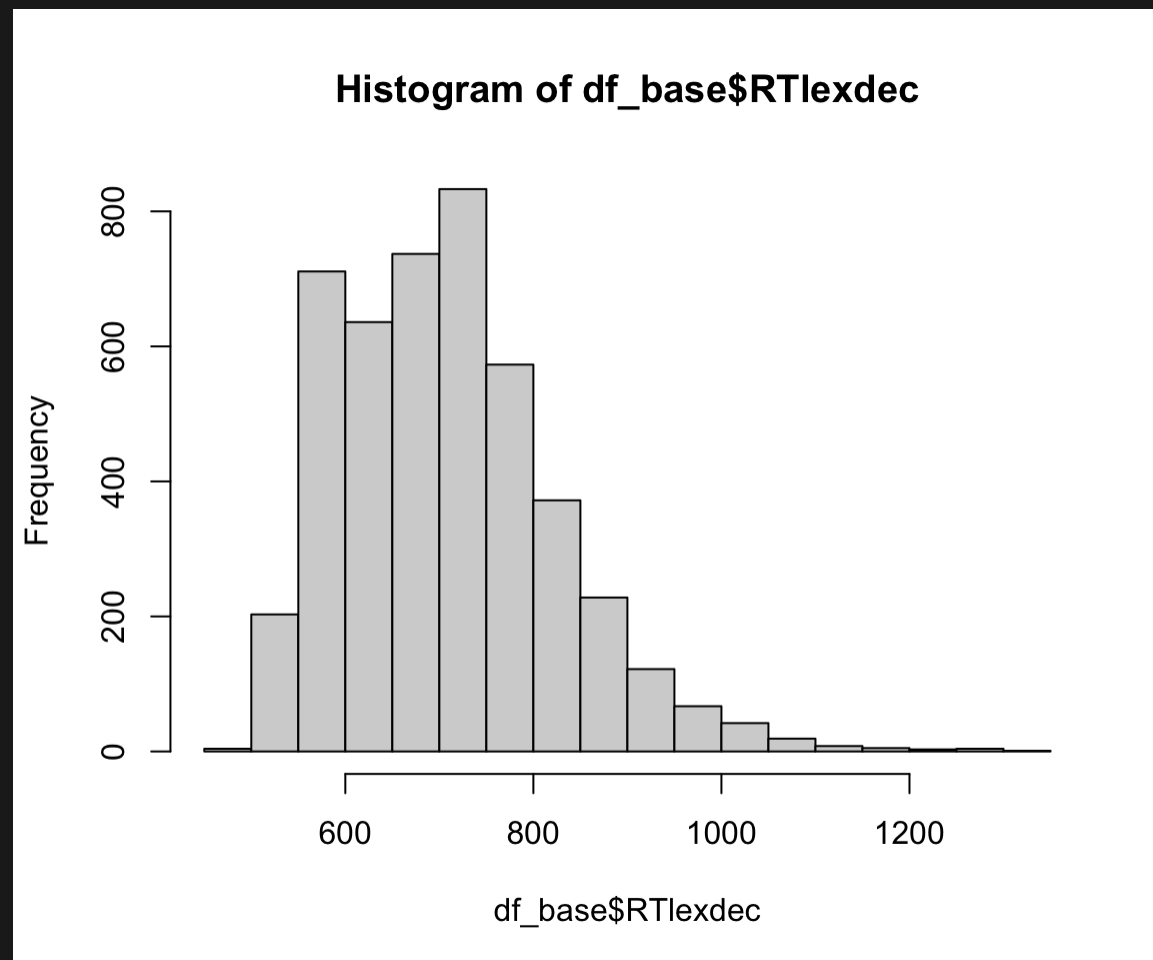
Plots

- `ggplot2` is popular even among people who don't use the tidyverse + this is because it has some useful features and a clean look

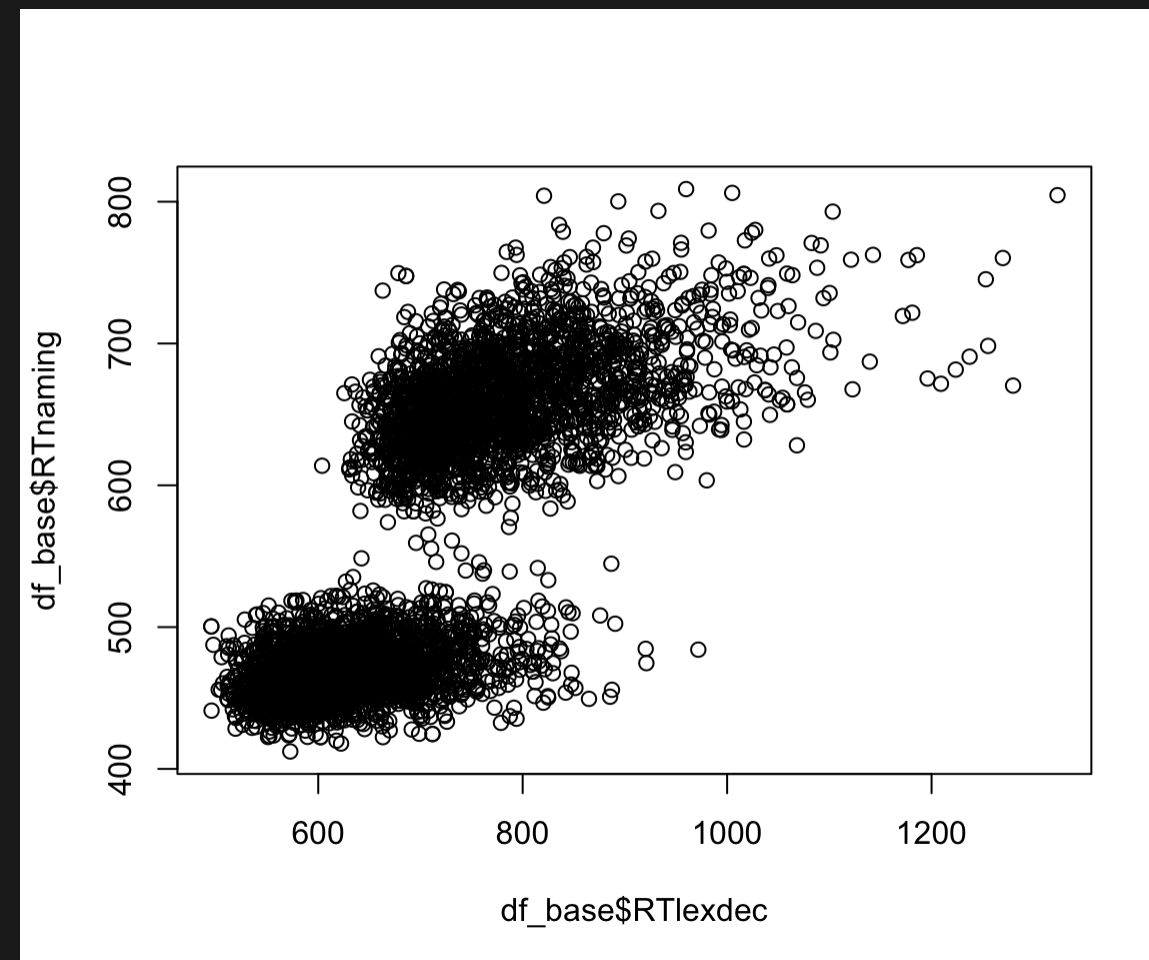
Plots: base R

- can be useful when you want to produce simple plots to get a first look at your data
 - the most useful functions are `hist()` and `plot()`
 - note that these functions work with vectors, which is why we have to use `$` to extract the columns from the data frame

```
1 hist(df_base$RTlexdec)
```



```
1 plot(df_base$RTlexdec, df_base$RTnaming)
```

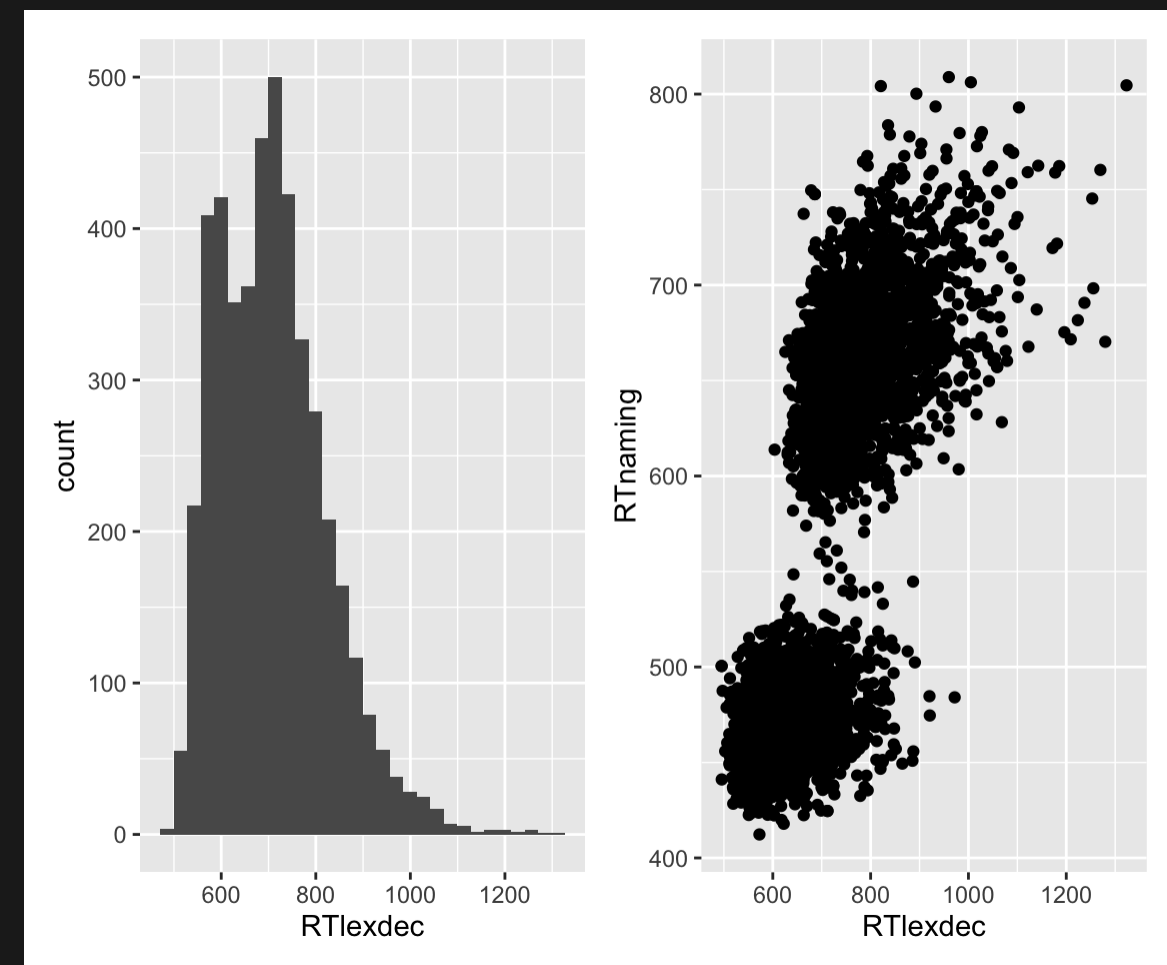


Plots: tidyverse

- as we've seen before:




```
1 library(patchwork)
2
3 # histogram
4 fig_hist <-
5   df_base |>
6   ggplot() +
7   aes(x = RTlexdec) +
8   geom_histogram()
9
10 # scatter plot
11 fig_scatter <-
12   df_base |>
13   ggplot() +
14   aes(x = RTlexdec, y = RTnaming) +
15   geom_point()
16
17 fig_hist + fig_scatter
```

Abbildung 1: Histogram and scatterplot with ggplot2



Learning objectives

In this chapter we will...

- learn what Base R is 
- compare Base R and Tidyverse 
- get to know the Base R equivalents of Tidyverse verbs 

Tasks

In the Console

```
1 rm(list=ls())  
2 .rs.restartR()
```

Convert the following tidyverse code to base R. We will again use the `languageR_english.csv` dataset.

Read-in

```
1 df_eng <-  
2   read_csv(here("daten", "languageR_english.csv"))
```

Selecting columns

```
1 df_eng |>  
2   select(Word, WrittenFrequency)
```

```
# A tibble: 10 × 2
```

	Word	WrittenFrequency
	<chr>	<dbl>
1	doe	3.91
2	whore	4.52
3	stress	6.51
4	pork	5.02
5	plug	4.89
6	prop	4.77
7	dawn	6.38
8	dog	7.16
9	arc	4.89
10	skirt	5.93

Filtering rows

```
1 df_eng |>
2   filter(WrittenFrequency > 5.6)
```

A tibble: 10 × 7

	AgeSubject	Word	LengthInLetters	WrittenFrequency	WordCategory	RTlexdec
	<chr>	<chr>	<dbl>	<dbl>	<chr>	<dbl>
1	young	stress	6	6.51	N	547.
2	young	dawn	4	6.38	N	584.
3	young	dog	3	7.16	N	527.
4	young	skirt	5	5.93	N	536.
5	young	are	3	11.3	N	611.
6	young	pipe	4	6.00	N	563.
7	young	guard	5	6.59	N	559.
8	young	slope	5	5.80	N	633.
9	young	pile	4	6.16	N	595.
10	young	tide	4	6.08	N	598.

i 1 more variable: RTnaming <dbl>

Selecting rows and columns

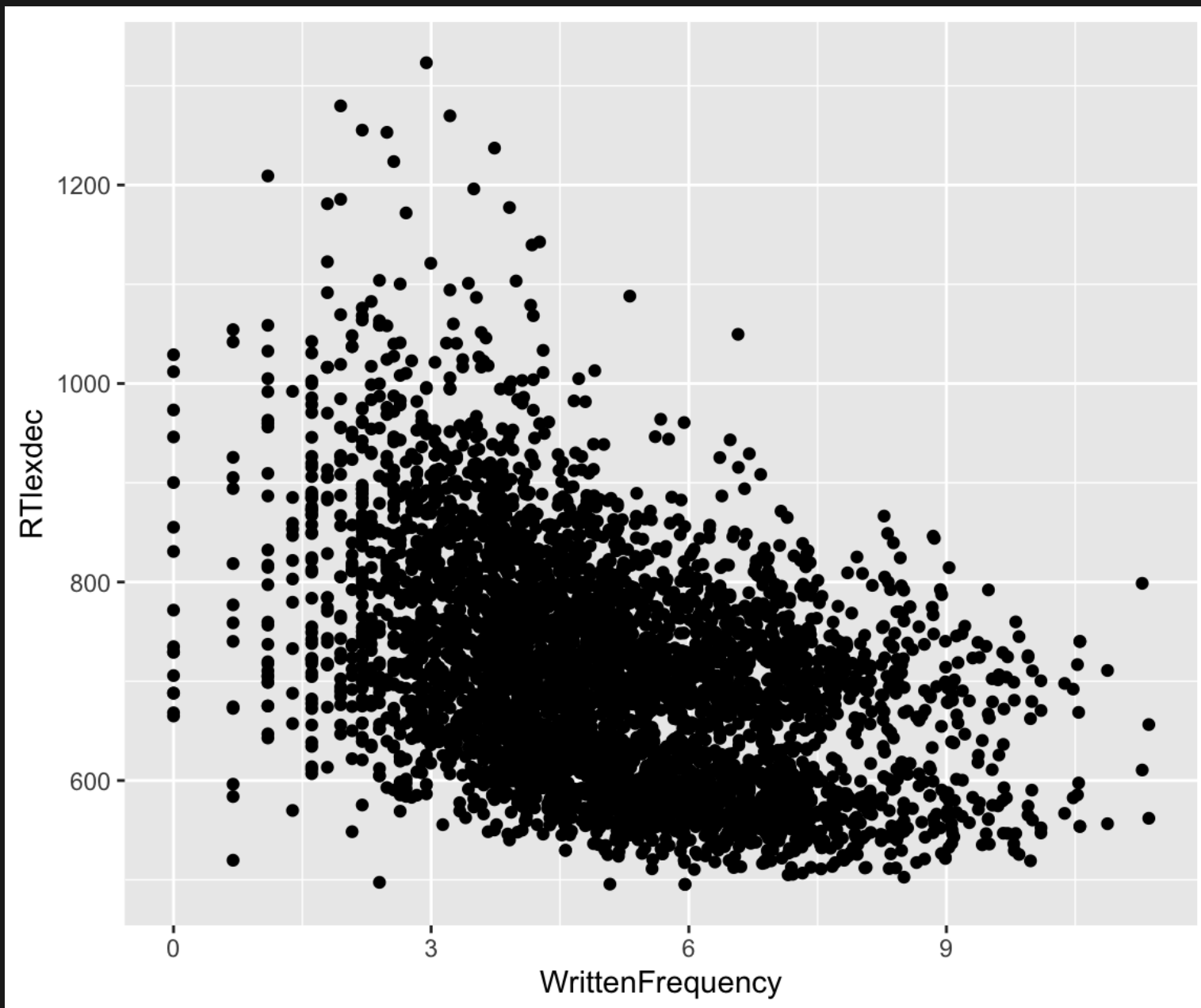
```
1 df_eng |>
2   filter(WrittenFrequency > 5.6 & AgeSubject == "old") |>
3   select(AgeSubject, Word, WrittenFrequency)
```

```
# A tibble: 10 × 3
```

	AgeSubject	Word	WrittenFrequency
	<chr>	<chr>	<dbl>
1	old	stress	6.51
2	old	dawn	6.38
3	old	dog	7.16
4	old	skirt	5.93
5	old	are	11.3
6	old	pipe	6.00
7	old	guard	6.59
8	old	slope	5.80
9	old	pile	6.16
10	old	tide	6.08

Scatterplot

```
1 df_eng |>  
2   ggplot() +  
3   aes(x = WrittenFrequency, y = RTlexdec) +  
4   geom_point()
```



Tidyverse versus base R

What is your impression of base R versus the tidyverse? Based on what you've seen, would you prefer one over the other, or would you prefer one in certain cases only? There's no correct answer here.

Session Info

Hergestellt mit R version 4.3.0 (2023-04-21) (Already Tomorrow) und RStudioversion 2023.9.0.463 (Desert Sunflower).

```
1 print(sessionInfo(), locale = F)
```

R version 4.3.0 (2023-04-21)

Platform: aarch64-apple-darwin20 (64-bit)

Running under: macOS Ventura 13.2.1

Matrix products: default

BLAS: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRblas.0.dylib

LAPACK: /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/lib/libRlapack.dylib; LAPACK version 3.11.0

attached base packages:

[1] stats graphics grDevices utils datasets methods base

other attached packages:

[1] patchwork_1.1.3 janitor_2.2.0 here_1.0.1 lubridate_1.9.2
[5] forcats_1.0.0 stringr_1.5.0 dplyr_1.1.3 purrr_1.0.2
[9] readr_2.1.4 tidyr_1.3.0 tibble_3.2.1 ggplot2_3.4.3

Literaturverzeichnis

- Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Golemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., ... Yutani, H. (2019a). Welcome to the tidyverse. *Journal of Open Source Software*, 4(43), 1686. <https://doi.org/10.21105/joss.01686>
- Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L., François, R., Golemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T., Miller, E., Bache, S., Müller, K., Ooms, J., Robinson, D., Seidel, D., Spinu, V., ... Yutani, H. (2019b). Welcome to the Tidyverse. *Journal of Open Source Software*, 4(43), 1686. <https://doi.org/10.21105/joss.01686>
- Wickham, H., Çetinkaya-Rundel, M., & Golemund, G. (2023). *R for Data Science* (2. Aufl.).
- Winter, B. (2019). Statistics for Linguists: An Introduction Using R. In *Statistics for Linguists: An Introduction Using R*. Routledge. <https://doi.org/10.4324/9781315165547>

