Danish Kings homework for week 10

Jeppe Strelow Lund

2025-05-03

The task here is to load your Danish Monarchs csv into R using the `tidyverse` toolkit, calculate and explore the kings' duration of reign with pipes `%>%` in `dplyr` and plot it over time.

Load the kings

Make sure to first create an `.Rproj` workspace with a `data/` folder where you place either your own dataset or the provided `kings.csv` dataset.

```
knitr::opts_chunk$set(echo = TRUE)
```

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.5.1     ✓ tibble    3.2.1
## ✓ lubridate 1.9.4     ✓ tidyr     1.3.1
## ✓ purrr     1.0.4
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to be
come errors
```

```
read_csv2("data/danishmonarchs_group14.csv")
```

```
## ℹ Using "','" as decimal and "'.'" as grouping mark. Use `read_delim()` for more control.
## Rows: 56 Columns: 8── Column specification ────────────────────────────────────────
────────
## Delimiter: ";"
## chr (2): regenter, slaegt_navn
## dbl (6): reg_start, reg_slut, fodsel, dod, Reg_tid, leve_tid
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
## # A tibble: 56 × 8
##    regenter         reg_start reg_slut fodsel   dod Reg_tid leve_tid slaegt_navn
##    <chr>                <dbl>    <dbl>  <dbl> <dbl>   <dbl>    <dbl> <chr>
##  1 Gorm_1_den_gamle       936      958    908   964      22       56 Jelling
##  2 Harald_1_blaata…       958      987    932   985      29       53 Jelling
##  3 Svend_1_Tveskaeg       987     1014    963  1014      27       51 Jelling
##  4 Harald_2              1014     1018    996  1018       4       22 Jelling
##  5 Knud_1_den_store      1018     1035    995  1035      17       40 Jelling
##  6 Hardeknud             1035     1042   1018  1042       7       24 Jelling
##  7 Magnus_den_gode       1042     1047   1024  1047       5       23 Norske
##  8 Svend_2_Estrids…      1047     1074   1019  1076      27       57 Jelling
##  9 Harald_3_hen          1074     1080   1040  1080       6       40 Jelling
## 10 Knud_2_den_hell…      1080     1096   1042  1086      16       44 Jelling
## # ℹ 46 more rows
```

1. Look at the dataset that are you loading and check what its columns are separated by? (hint: open it in plain text editor to see)

List what is the

separator:semicolon

2. Create a `kings` object in R with the different functions below and inspect the different outputs.

- `read.csv()`
- `read_csv()`
- `read.csv2()`
- `read_csv2()`

```
# FILL IN THE CODE BELOW and review the outputs
kings1 <- read.csv("data/danishmonarchs_group14.csv")

kings2 <- read_csv("data/danishmonarchs_group14.csv")
```

```
## Rows: 56 Columns: 1
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## chr (1): regenter;reg_start;reg_slut;fodsel;dod;Reg_tid;leve_tid;slaegt_navn
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
kings3 <- read.csv2("data/danishmonarchs_group14.csv")

kings4 <- read_csv2("data/danishmonarchs_group14.csv")
```

```
## ℹ Using "','" as decimal and "'.'" as grouping mark. Use `read_delim()` for more control.
## Rows: 56 Columns: 8── Column specification ────────────────────────────────────────────────
────────
## Delimiter: ";"
## chr (2): regenter, slaegt_navn
## dbl (6): reg_start, reg_slut, fodsel, dod, Reg_tid, leve_tid
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
class(kings4)
```

```
## [1] "spec_tbl_df" "tbl_df"      "tbl"         "data.frame"
```

Answer: 1. Which of these functions is a `tidyverse` function? Read data with it below into a `kings` object kings3 og kings4 er de korrekte, da den afspejler dataene i vores datasæt, 56 observationer og 8 collums

2. What is the result of running `class()` on the `kings` object created with a tidyverse function. når man kører funktionen class() på kings er resultat= “spec_tbl_df” “tbl_df” “tbl” “data.frame”
3. How many columns does the object have when created with these different functions? 8
4. Show the dataset so that we can see how R interprets each column

```
# COMPLETE THE BLANKS BELOW WITH YOUR CODE, then turn the 'eval' flag in this chunk to TRUE.

kings <- kings4

class(kings)
```

```
## [1] "spec_tbl_df" "tbl_df"      "tbl"         "data.frame"
```

```
ncol(kings)
```

```
## [1] 8
```

```
glimpse(kings)
```

```
## Rows: 56
## Columns: 8
## $ regenter    <chr> "Gorm_1_den_gamle", "Harald_1_blaatand", "Svend_1_Tveskaeg…
## $ reg_start   <dbl> 936, 958, 987, 1014, 1018, 1035, 1042, 1047, 1074, 1080, 1…
## $ reg_slut    <dbl> 958, 987, 1014, 1018, 1035, 1042, 1047, 1074, 1080, 1096, …
## $ fodsel      <dbl> 908, 932, 963, 996, 995, 1018, 1024, 1019, 1040, 1042, 105…
## $ dod         <dbl> 964, 985, 1014, 1018, 1035, 1042, 1047, 1076, 1080, 1086, …
## $ Reg_tid     <dbl> 22, 29, 27, 4, 17, 7, 5, 27, 6, 16, 9, 8, 30, 3, 9, 11, 25…
## $ leve_tid    <dbl> 56, 53, 51, 22, 40, 24, 23, 57, 40, 44, 45, 48, 69, 47, 46…
## $ slaegt_navn <chr> "Jelling", "Jelling", "Jelling", "Jelling", "Jelling", "Je…
```

Calculate the duration of reign for all the kings in your table

You can calculate the duration of reign in years with `mutate` function by subtracting the equivalents of your `startReign` from `endReign` columns and writing the result to a new column called `duration`. But first you need to check a few things:

- Is your data messy? Fix it before re-importing to R
- Do your start and end of reign columns contain NAs? Choose the right strategy to deal with them: `na.omit()`, `na.rm=TRUE`, `!is.na()`

Create a new column called `duration` in the kings dataset, utilizing the `mutate()` function from tidyverse. Check with your group to brainstorm the options.

```
# YOUR CODE
kings %>%
  mutate(duration= reg_slut - reg_start) %>%
  select(duration)
```

```
## # A tibble: 56 × 1
##    duration
##       <dbl>
##  1       22
##  2       29
##  3       27
##  4        4
##  5       17
##  6        7
##  7        5
##  8       27
##  9        6
## 10       16
## # ℹ 46 more rows
```

Calculate the average duration of reign for all rulers

Do you remember how to calculate an average on a vector object? If not, review the last two lessons and remember that a column is basically a vector. So you need to subset your `kings` dataset to the `duration` column. If you subset it as a vector you can calculate average on it with `mean()` base-R function. If you subset it as a tibble, you can calculate average on it with `summarize()` tidyverse function. Try both ways!

- You first need to know how to select the relevant `duration` column. What are your options?
- Is your selected `duration` column a tibble or a vector? The `mean()` function can only be run on a vector. The `summarize()` function works on a tibble.
- Are you getting an error that there are characters in your column? Coerce your data to numbers with `as.numeric()`.
- Remember to handle NAs: `mean(X, na.rm=TRUE)`

```
# YOUR CODE
kings %>%
  mutate(duration = reg_slut - reg_start) %>%
  summarise(mean_duration = mean(duration, na.rm=TRUE))
```

```
## # A tibble: 1 × 1
##   mean_duration
##           <dbl>
## 1          19.6
```

How many and which kings enjoyed a longer-than-average duration of reign?

You have calculated the average duration above. Use it now to `filter()` the `duration` column in `kings` dataset. Display the result and also count the resulting rows with `count()`

```
# YOUR CODE
kings %>%
  mutate(duration = reg_slut - reg_start) %>%
  filter(duration > 19.5537) %>%
  count()
```

```
## # A tibble: 1 × 1
##       n
##   <int>
## 1    27
```

How many days did the three longest-ruling monarchs rule?

- Sort kings by reign `duration` in the descending order. Select the three longest-ruling monarchs with the `slice()` function
- Use `mutate()` to create `Days` column where you calculate the total number of days they ruled
- BONUS: consider the transition year (with 366 days) in your calculation!

```
# YOUR CODE
kings %>%
  mutate(duration = reg_slut - reg_start) %>%
  arrange(desc(duration)) %>%
  slice(c(1, 2, 3)) %>%
  mutate(regtid_dage = 365 * duration)
```

```
## # A tibble: 3 × 10
##   regenter reg_start reg_slut fodsel   dod Reg_tid leve_tid slaegt_navn duration
##   <chr>        <dbl>    <dbl>  <dbl> <dbl>   <dbl>    <dbl> <chr>          <dbl>
## 1 Christi…      1588     1648   1577  1648      60       71 Oldenborg         60
## 2 Margret…      1972     2024   1940    NA      52       NA Glucksburg        52
## 3 Erik_7_…      1396     1439   1382  1459      43       77 Jelling           43
## # ℹ 1 more variable: regtid_dage <dbl>
```

```
# Herunder kan man tjekke efter om det passer, og det gør det
# 60*365 = 21900
# 52*365 = 18980
# 43*365 = 15695
```

Challenge: Plot the kings' duration of reign through time

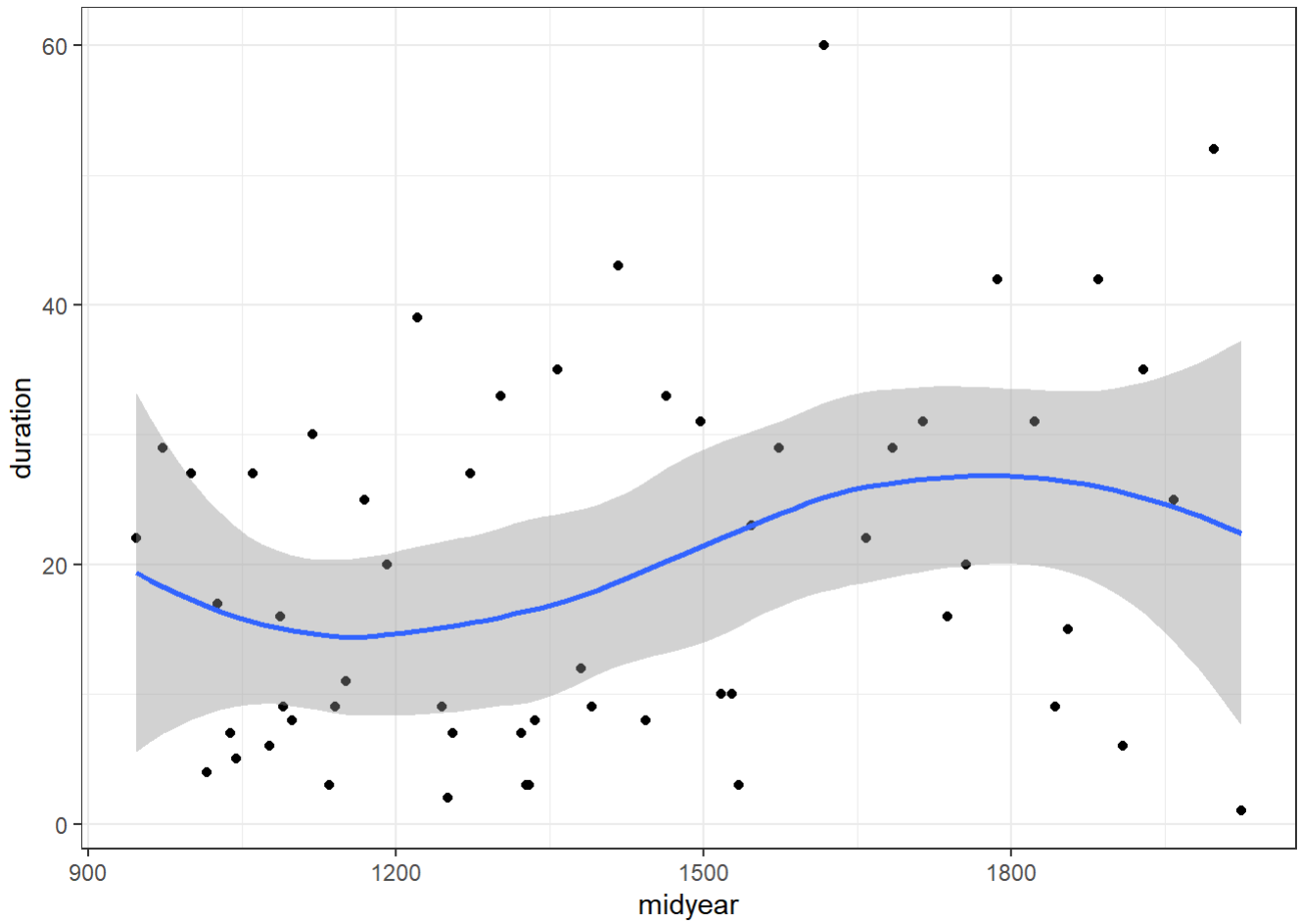
What is the long-term trend in the duration of reign among Danish monarchs? How does it relate to the historical violence trends ?

- Try to plot the duration of reign column in `ggplot` with `geom_point()` and `geom_smooth()`
- In order to peg the duration (which is between 1-99) somewhere to the x axis with individual centuries, I recommend creating a new column `midyear` by adding to `startYear` the product of `endYear` minus the `startYear` divided by two (`startYear + (endYear-startYear)/2`).
- Now you can plot the kings dataset, plotting `midyear` along the x axis and `duration` along y axis
- BONUS: add a title, nice axis labels to the plot and make the theme B&W and font bigger to make it nice and legible!

```
# YOUR CODE

kings %>%
  mutate(duration= reg_slut - reg_start) %>%
  mutate(midyear= reg_start + duration/2) %>%
  ggplot(aes(x = midyear, y = duration))+
  geom_point()+
  geom_smooth()+
  theme_bw()
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



And to submit this rmarkdown, knit it into html. But first, clean up the code chunks, adjust the date, rename the author and change the `eval=FALSE` flag to `eval=TRUE` so your script actually generates an output. Well done!