MSD Homework 2, Problem 3

Your Name (your uni)

2019-03-13 09:46:34

Contents

Description

This is a template for exercise 6 in Chapter 2 of Bit By Bit: Social Research in the Digital Age by Matt Salganik. The problem is reprinted here with some additional comments and structure to facilitate a solution.

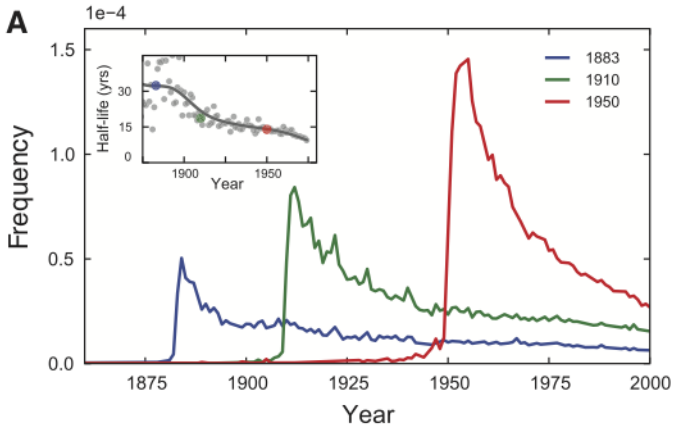
The original problem statement:

> In a widely discussed paper, Michel and colleagues (2011) analyzed the content of more than five million digitized books in an attempt to identify long-term cultural trends. The data that they used has now been released as the Google NGrams dataset, and so we can use the data to replicate and extend some of their work.

In one of the many results in the paper, Michel and colleagues argued that we are forgetting faster and faster. For a particular year, say "1883," they calculated the proportion of 1-grams published in each year between 1875 and 1975 that were "1883". They reasoned that this proportion is a measure of the interest in events that happened in that year. In their figure 3a, they plotted the usage trajectories for three years: 1883, 1910, and 1950. These three years share a common pattern: little use before that year, then a spike, then decay. Next, to quantify the rate of decay for each year, Michel and colleagues calculated the "half-life" of each year for all years between 1875 and 1975. In their figure 3a (inset), they showed that the half-life of each year is decreasing, and they argued that this means that we are forgetting the past faster and faster. They used Version 1 of the English language corpus, but subsequently Google has released a second version of the corpus. Please read all the parts of the question before you begin coding.

This activity will give you practice writing reusable code, interpreting results, and data wrangling (such as working with awkward files and handling missing data). This activity will also help you get up and running with a rich and interesting dataset.

The full paper can be found here, and this is the original figure 3a that you're going to replicate:



Part A

Get the raw data from the Google Books NGram Viewer website. In particular, you should use version 2 of the English language corpus, which was released on July 1, 2012. Uncompressed, this file is 1.4GB.

Get and clean the raw data

Edit the `01_download_1grams.sh` file to download the `googlebooks-eng-all-1gram-20120701-1.gz` file and the `02_filter_1grams.sh` file to filter the original 1gram file to only lines where the ngram matches a year (output to a file named `year_counts.tsv`).

Then edit the `03_download_totals.sh` file to down the `googlebooks-eng-all-totalcounts-20120701.txt` and file and the `04_reformat_totals.sh` file to reformat the total counts file to a valid csv (output to a file named `total_counts.csv`).

Load the cleaned data

Load in the `year_counts.tsv` and `total_counts.csv` files. Use the `here()` function around the filename to keep things portable.Give the columns of `year_counts.tsv` the names `term`, `year`, `volume`, and `book_count`. Give the columns of `total_counts.csv` the names `year`, `total_volume`, `page_count`, and `book_count`. Note that column order in these files may not match the examples in the documentation.

```
ngrams_years <- read_tsv(here('year_counts.tsv'),
                  col_names = c('term',  'year', 'volume', 'book_count'),
                  col_types = 'ciii' )

totals <- read_csv(here('total_counts.csv'),
                  col_names = c('year', 'total_volume', 'page_count', 'book_count'),
                  col_types = 'idii')
```

Your written answer

Add a line below using Rmarkdown's inline syntax to print the total number of lines in each dataframe you've created.

Part B

> Recreate the main part of figure 3a of Michel et al. (2011). To recreate this figure, you will need two files: the one you downloaded in part (a) and the "total counts" file, which you can use to convert the raw counts into proportions. Note that the total counts file has a structure that may make it a bit hard to read in. Does version 2 of the NGram data produce similar results to those presented in Michel et al. (2011), which are based on version 1 data?

Join ngram year counts and totals

Join the raw year term counts with the total counts and divide to get a proportion of mentions for each term normalized by the total counts for each year.

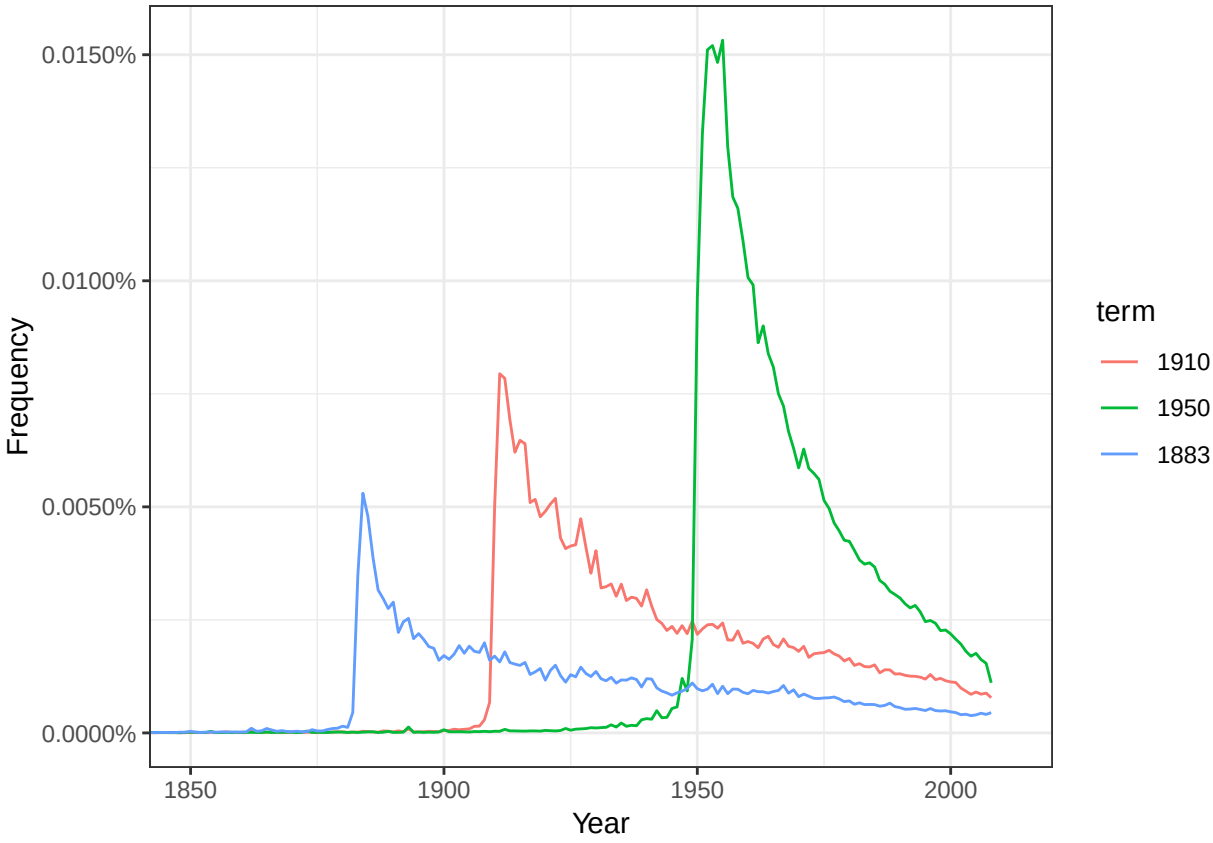
```
ngrams_years <- ngrams_years %>%
  left_join(totals, by = "year") %>%
  mutate(frac_total = volume / total_volume) %>%
  select(term, year, volume, total_volume, frac_total)
```

Plot the main figure 3a

Plot the proportion of mentions for the terms "1883", "1910", and "1950" over time from 1850 to 2012, as in the main figure 3a of the original paper. Use the `percent` function from the `scales` package for a readable y axis. Each term should have a different color, it's nice if these match the original paper but not strictly necessary.

```
color_order <- c(1910, 1950, 1883)
ngrams_years %>%
  filter(term %in% c(1883, 1910, 1950)) %>%
  mutate(term = factor(as.character(term), levels = color_order)) %>%
  ggplot(aes(x = year, y = frac_total, color = term)) +
  geom_line() +
```

```
  scale_y_continuous(label = percent) +
  coord_cartesian(xlim = c(1850, 2012)) +
  labs(x = 'Year', y = 'Frequency')
```



Your written answer

Write up your answer to Part B here.

Part C

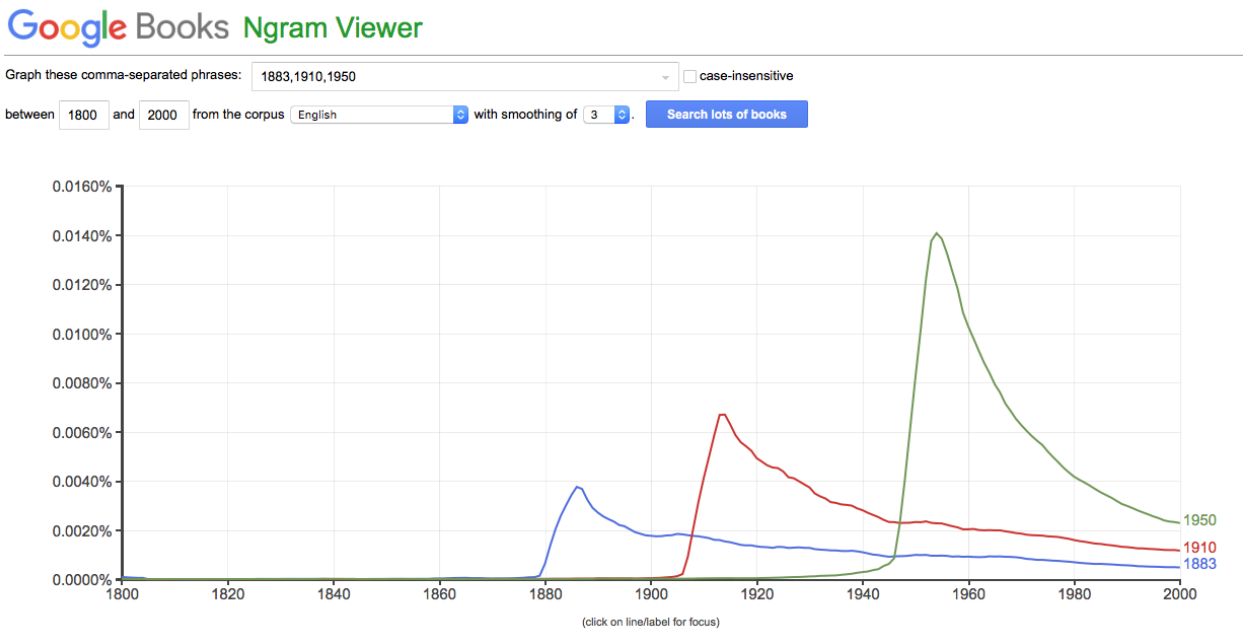
Now check your graph against the graph created by the NGram Viewer.

Compare to the NGram Viewer

Go to the ngram viewer, enter the terms “1883”, “1910”, and “1950” and take a screenshot.

Your written answer

Add your screenshot for Part C below this line using the `![](figure_filename.png)` syntax and comment on similarities / differences.



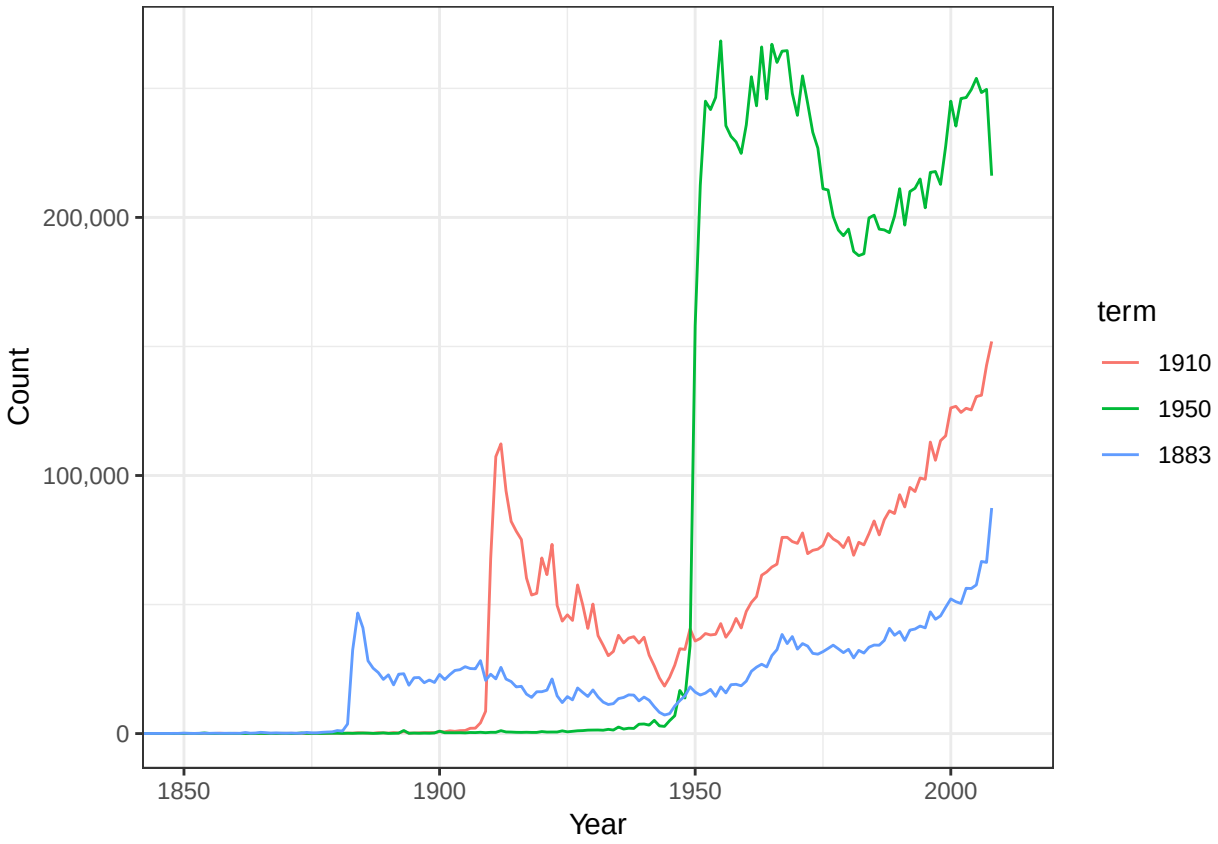
Part D

Recreate figure 3a (main figure), but change the y-axis to be the raw mention count (not the rate of mentions).

Plot the main figure 3a with raw counts

Plot the raw counts for the terms "1883", "1910", and "1950" over time from 1850 to 2012. Use the `comma` function from the `scales` package for a readable y axis. The colors for each term should match your last plot, and it's nice if these match the original paper but not strictly necessary.

```
color_order <- c(1910, 1950, 1883)
ngrams_years %>%
  filter(term %in% c(1883, 1910, 1950)) %>%
  mutate(term = factor(as.character(term), levels = color_order)) %>%
  ggplot(aes(x = year, y = volume, color = term)) +
  geom_line() +
  scale_y_continuous(label = comma) +
  coord_cartesian(xlim = c(1850, 2012)) +
  labs(x = 'Year', y = 'Count')
```



Part E

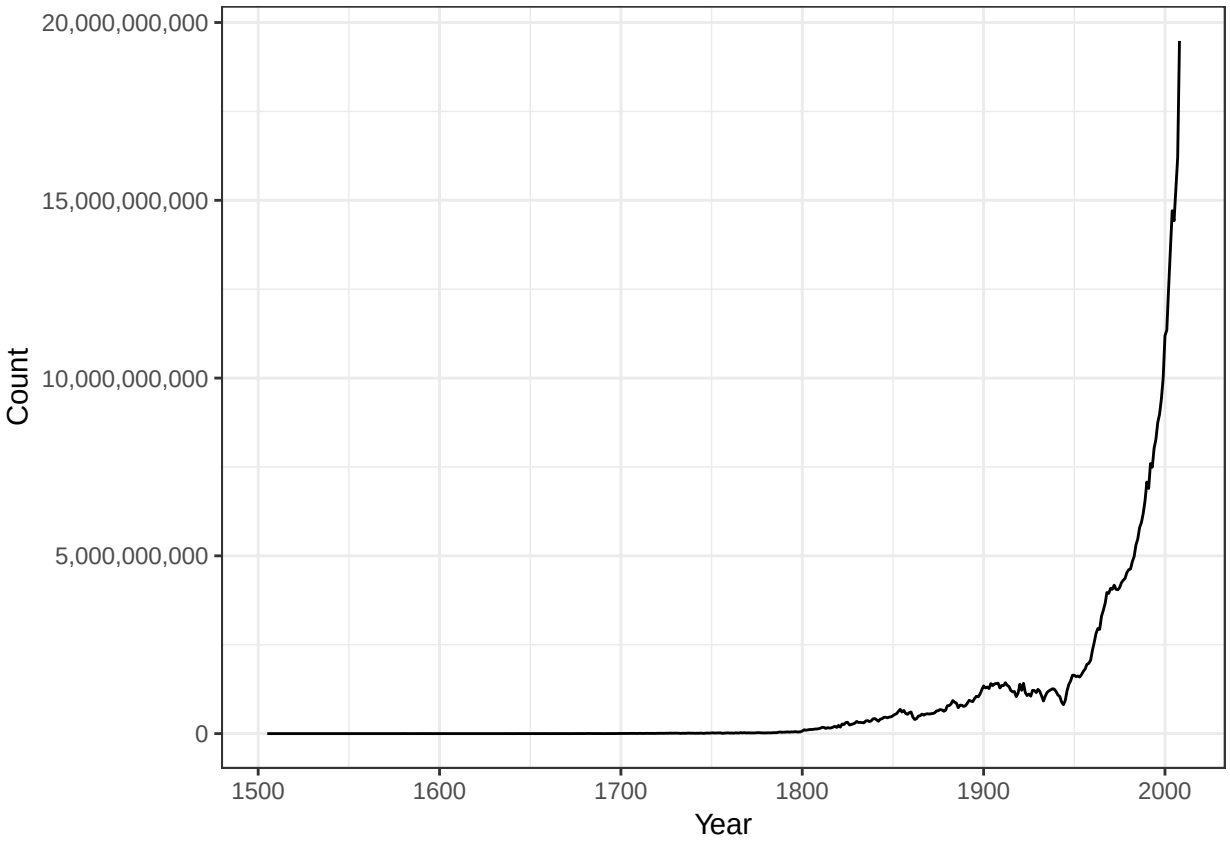
> Does the difference between (b) and (d) lead you to reevaluate any of the results of Michel et al. (2011). Why or why not?

As part of answering this question, make an additional plot.

Plot the totals

Plot the total counts for each year over time, from 1850 to 2012. Use the `comma` function from the `scales` package for a readable y axis. There should be only one line on this plot (not three).

```
ggplot(totals, aes(x = year, y = total_volume)) +
  geom_line() +
  scale_y_continuous(label = comma) +
  labs(x = 'Year', y = 'Count')
```



Your written answer

Write up your answer to Part E here.

Part F

> Now, using the proportion of mentions, replicate the inset of figure 3a. That is, for each year between 1875 and 1975, calculate the half-life of that year. The half-life is defined to be the number of years that pass before the proportion of mentions reaches half its peak value. Note that Michel et al. (2011) do something more complicated to estimate the half-life—see section III.6 of the Supporting Online Information—but they claim that both approaches produce similar results. Does version 2 of the NGram data produce similar results to those presented in Michel et al. (2011), which are based on version 1 data? (Hint: Don't be surprised if it doesn't.)

Compute peak mentions

For each year term, find the year where its proportion of mentions peaks (hits its highest value). Store this in an intermediate dataframe.

```
# filtering for year >= 1800 to get rid of a noisy maxima in the 1500s
# could do this differently by looking for peak_year >= year

peaks <- ngrams_years %>%
  filter(year >= 1800) %>%
```

```
  group_by(term) %>%
  filter(frac_total == max(frac_total)) %>%
  select(term, peak_year = year, peak_frac = frac_total)
```

Compute half-lifes

Now, for each year term, find the minimum number of years it takes for the proportion of mentions to decline from its peak value to half its peak value. Store this in an intermediate data frame.

```
# half_life calculation is a bit ambiguous from the problem statement
# could be year at half max - year of term
# or could be year at half max - peak year
# doesn't make a substantive difference, either is fine

half_lifes <- ngrams_years %>%
  left_join(peaks) %>%
  filter(year >= peak_year, frac_total <= peak_frac / 2) %>%
  group_by(term) %>%
  filter(year == min(year)) %>%
  summarize(half_life = year - as.integer(term))
```

```
## Joining, by = "term"
```

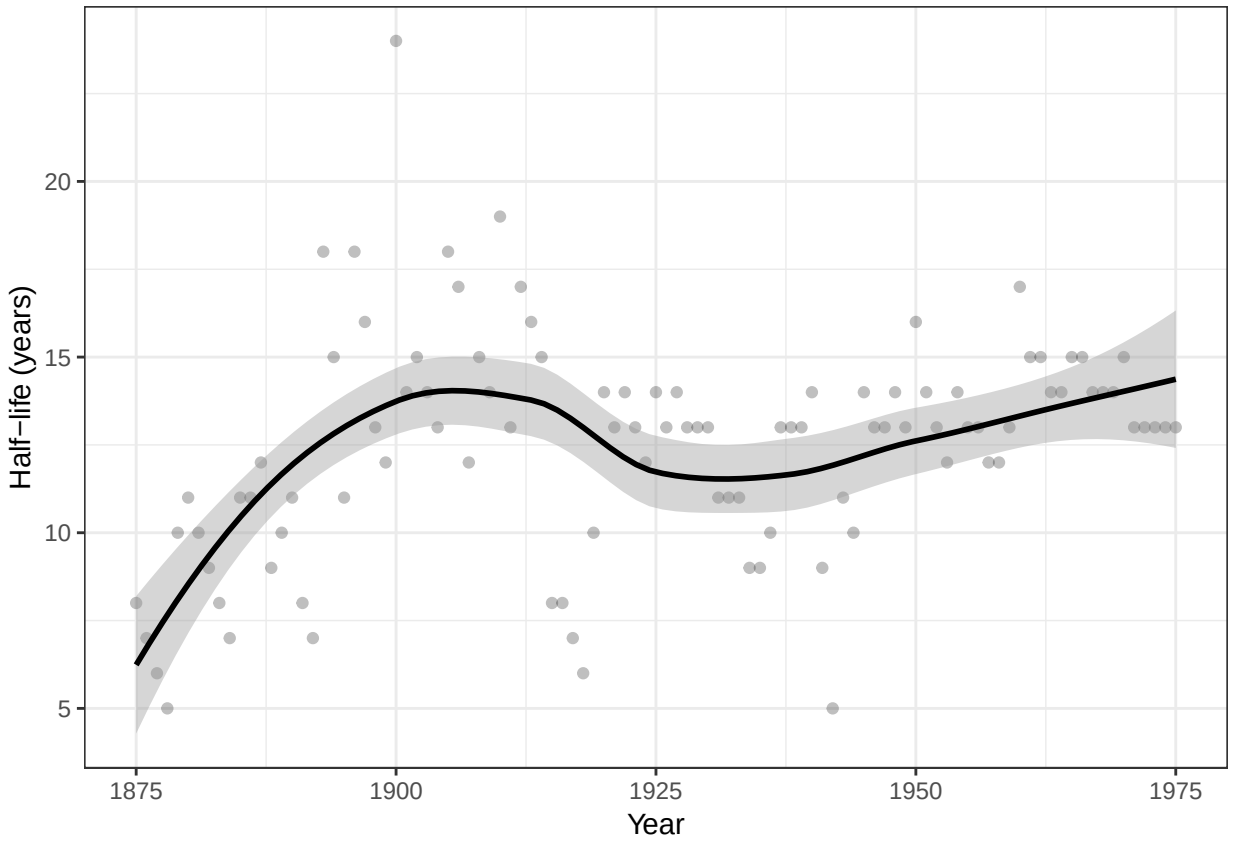
Plot the inset of figure 3a

Plot the half-life of each term over time from 1850 to 2012. Each point should represent one year term, and add a line to show the trend using `geom_smooth()`.

```
# how you do the filtering by year here matters
# below is equivalent to using xlim to filter data before plotting
# using coord_cartesian without filter will give a different (flatter) result

half_lifes %>%
  mutate(term = as.integer(term)) %>%
  filter(term >= 1875, term <= 1975) %>%
  ggplot(aes(x = as.integer(term), y = half_life)) +
  geom_point(alpha = 0.25) +
  geom_smooth(color = 'black') +
  labs(x = 'Year', y = 'Half-life (years)')
```

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



Your written answer

Write up your answer to Part F here.

Part G

> Were there any years that were outliers such as years that were forgotten particularly quickly or particularly slowly? Briefly speculate about possible reasons for that pattern and explain how you identified the outliers.

Your written answer

Write up your answer to Part G here. Include code that shows the years with the smallest and largest half-lifes.

Makefile

Edit the `Makefile` in this directory to execute the full set of scripts that download the data, clean it, and produce this report. This must be turned in with your assignment such that running `make` on the command line produces the final report as a pdf file.