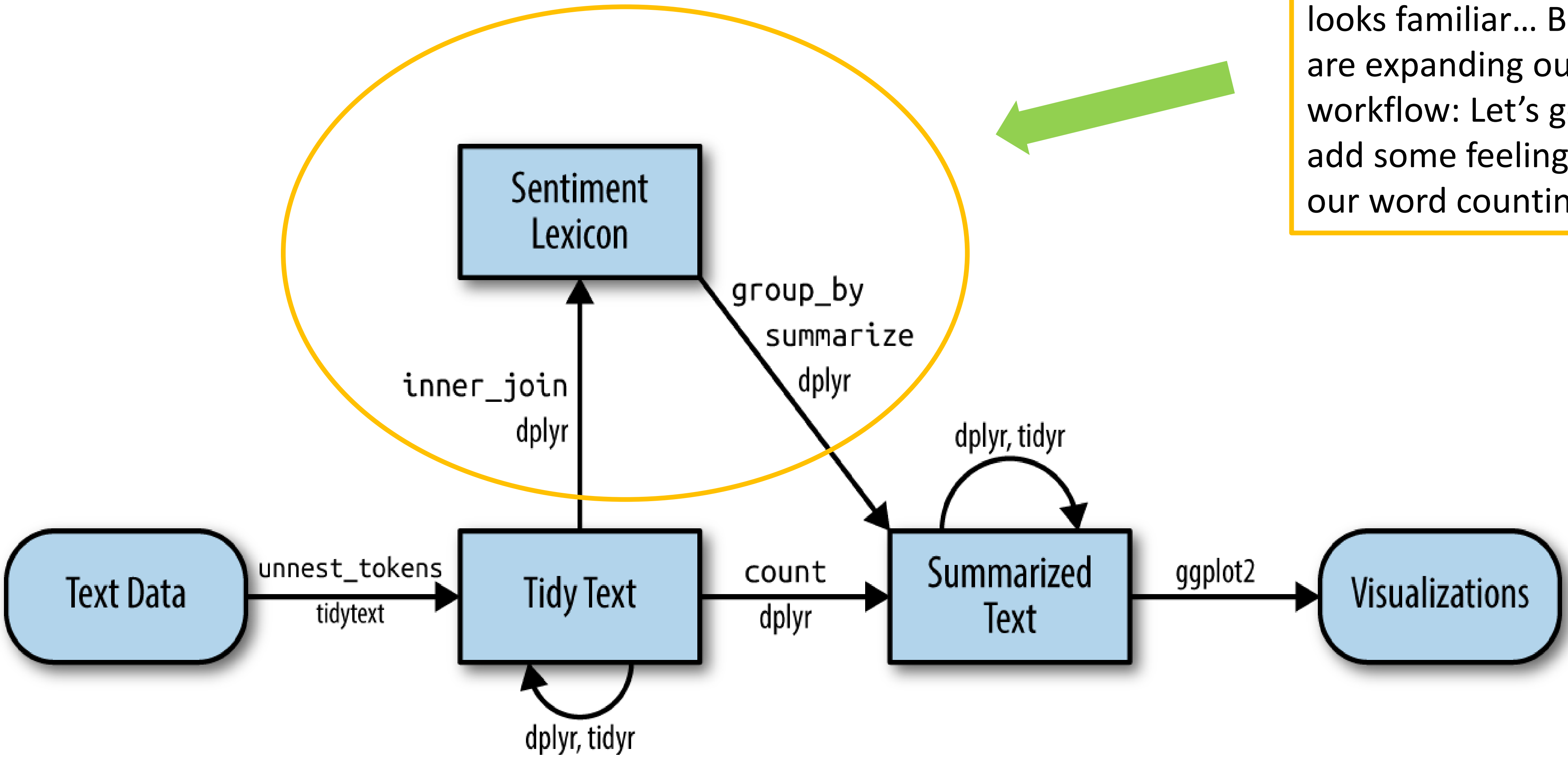
Part 2: Sentiment analysis with tidy text data I

- Your tidy text mining skills so far include…
  - Tidying a text corpus
  - Remove stop words from your data set
  - Find the Most Common Words in a text corpus
  - Compare MCW's across texts
- Ready for the next step? Sure! Let's dive into sentiment analysis, since…

When human readers approach a text, we use our understanding of the emotional intent of words to infer whether a section of text is positive or negative, or perhaps characterized by some other more nuanced emotion like surprise or disgust. We can use the tools of text mining to approach the emotional content of text programmatically.

Sentiment analysis with tidy text data II



Well, this flowchart looks familiar… But we are expanding our workflow: Let's go and add some feeling to our word counting!

Sentiment analysis with tidy text data III

- To analyze the sentiment of a text we view the text as a combination of its individual words
- We define the sentiment content of the whole text as the sum of the sentiment content of the individual words
- The tidytext package provides access to several sentiment lexicons (sentiments datasets) . These lexicons are based on unigrams (single words) and contain many English words and the words are assigned scores for positive/negative sentiment, and also possibly emotions like joy, anger, sadness, and so forth.

Sentiment analysis with tidy text data IV

- Three general-purpose lexicons (sentiments datasets) are
  - AFINN from Finn Årup Nielsen
  - bing from Bing Liu and collaborators
  - nrc from Saif Mohammad and Peter Turney

Exercise 9

9a. The function get_sentiments allows us to get specific sentiment lexicons with the appropriate measures for each one.

Can you call on the tidytext package and then use the function mentioned above to get tibbles of the three lexicons mentioned on this slide? This only requires calling on the package and using the function for an individual sentiments dataset. Have a go at it!

Have a look at the different outputs these lexicons provide. Can you characterize the various ways they score sentiment?

Sentiment analysis of *Emma* I

9b. Let’s ask ourselves: What are the most common joy words in Austen’s novel *Emma*? Run this code in order to make your data tidy first and do some real code work in the next exercise!

```
library(janeaustenr)
library(dplyr)
library(stringr)

tidy_books <- austen_books() %>%
  group_by(book) %>%
  mutate(
    linenumber = row_number(),
    chapter = cumsum(str_detect(text,
                                regex("^chapter [\\divxlc]",
                                      ignore_case = TRUE)))) %>%
  ungroup() %>%
  unnest_tokens(word, text)
```

Sentiment analysis of *Emma* II

9c. The text is now in a tidy format with one word per row: We are ready to do sentiment analysis! **We want to know what the most common joy words in *Emma* are.** Can you complete the code and run the script based on these pointers?

i) Use the nrc lexicon and filter() for the joy words

ii) filter() the data frame with the text from the books for the words from *Emma*

iii) Use inner_join() to perform the sentiment analysis

iv) And last but not least, let’s count() the most common joy words in *Emma*

Don’t worry, you can click for the next slide and some fun code to complete!

Sentiment analysis of *Emma* III

```
nrc_joy <- ???_???(“???") %>%
  filter(sentiment == “???")

tidy_books %>%
  ???(book == “???") %>%
  ???_???(nrc_joy) %>%
  ???(word, sort = TRUE)
```

← Did you get a tibble with words and a word count? Congratulations, you have found “joy” in *Emma*! But there is more to explore…

Sentiment analysis of Austen’s novels I

9d. We can also examine how sentiment changes throughout each of Austen’s novels. We can do this with just a handful of lines that are mostly dplyr functions. Can you complete the code and run the script based on these pointers?

i) Use the bing lexicon to find a sentiment score for each word

ii) Define an index of 80 lines, so we count up how many positive and negative words there are in defined sections of each book

iii) Use pivot_wider() so that we have negative and positive sentiment in separate columns

iv) Press ‘Run’ and calculate a net sentiment (positive - negative)

In for completing some code? Go the next slide!

Sentiment analysis of Austen’s novels II

```
library(tidyr)


jane_austen_sentiment <- tidy_books %>%
  inner_join(???_???(“???")) %>%
  count(book, index = linenumber %/% ??, sentiment) %>%
  ???_???(names_from = sentiment, values_from = n, values_fill = 0)
%>%
  mutate(sentiment = positive - negative)
```

…and then →

Sentiment analysis of Austen’s novels III

9e. It’s time for some more visualization to (literally) bring the results of your sentiment analysis into view. We can plot the sentiment scores you just calculated across the plot trajectory of each novel. Notice that we are plotting against the index (the 80 line increment) on the x-axis that keeps track of narrative time in sections of text. They only thing you need to do is call on the ggplot2 package (do you remember how to call on a package? You have done so numerous times before!) and then run the following code:

```
ggplot(jane_austen_sentiment, aes(index, sentiment, fill = book)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~book, ncol = 2, scales = "free_x")
```

← If you were successful, you see 6 different plots, corresponding to the titles of Austen’s 6 novels. Can you discern any trends or differences in the novels’ sentiment structures?