Week 3: Grouping, Summarizing, and Custom Visuals

Ronnie Bailey-Steinitz

2025-10-06

Skills Learning – Lecture

This week we introduce only **two new functions** — `group_by()` and `summarise()` — and then spend most of the time visualizing data.

We'll use the **gapminder** dataset, which contains demographic and economic data from multiple countries between 1952 and 2007. Each row represents a country–year combination.

0. Load Required Packages

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   4.0.0     v tibble    3.3.0
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
# install.packages("here")
library(here) # this package will help overcome issues with knitting and your working directory
```

```
## here() starts at /Users/rsteinitz/Documents/github/R Data Analysis Course
```

1. Load Data

```r
# note the inclusion of the here() command around the file path to help with knitting conflicts
data <- read_csv(here("Week 3/gapminder.csv")) %>%
  janitor::clean_names()
```

```
## Rows: 1704 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): country, continent
## dbl (4): year, lifeExp, pop, gdpPercap
##
## 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.
```

```r
glimpse(data)   # quick look at variables
```

```
## Rows: 1,704
## Columns: 6
## $ country    <chr> "Afghanistan", "Afghanistan", "Afghanistan", "Afghanistan",~
## $ continent  <chr> "Asia", "Asia", "Asia", "Asia", "Asia", "Asia", "Asia", "As~
## $ year       <dbl> 1952, 1957, 1962, 1967, 1972, 1977, 1982, 1987, 1992, 1997,~
## $ life_exp   <dbl> 28.801, 30.332, 31.997, 34.020, 36.088, 38.438, 39.854, 40.~
## $ pop        <dbl> 8425333, 9240934, 10267083, 11537966, 13079460, 14880372, 1~
## $ gdp_percap <dbl> 779.4453, 820.8530, 853.1007, 836.1971, 739.9811, 786.1134,~
```

Now that we can filter and select data, we'll learn how to summarize it by group. Think of this as splitting data into smaller pieces (groups), calculating statistics within each, and then combining the results.

2. Grouping and Summarizing

2.1 group_by() + summarise()

First, let's look at a single overall mean. Then we'll use group_by() to see how the mean changes across groups.

```r
# We can get a summary statistic
mean(data$life_exp) # quick and easy way
```

```
## [1] 59.47444
```

```r
data %>% summarize(mean(life_exp, na.rm = TRUE)) # within a pipe
```

```
## # A tibble: 1 x 1
##   ‘mean(life_exp, na.rm = TRUE)‘
##                            <dbl>
## 1                           59.5
```

```r
### Q: Now what happens when we add group_by()? =====================
# Example: average life expectancy by continent
data %>%
  dplyr::group_by(continent) %>%
  dplyr::summarise(mean_life_exp = mean(life_exp, na.rm = TRUE))
```

```
## # A tibble: 5 x 2
##   continent mean_life_exp
##   <chr>             <dbl>
## 1 Africa             48.9
## 2 Americas           64.7
## 3 Asia               60.1
## 4 Europe             71.9
## 5 Oceania            74.3
```

```
### A: We get the mean by group, in this case, continent =====================


# We can combine filter() and group_by() in one pipeline to focus on a subset of the data before summar
data1 <- data %>%
  filter(year == 2007) %>%
  dplyr::group_by(continent) %>%
  dplyr::summarise(mean_life_exp = mean(life_exp, na.rm = TRUE),
                   n_countries = n())

write_csv(data1, "Mean Life Expectancy by Continent.csv")
```

Exporting results with `write_csv()` is useful if you want to save summary tables for later analysis or visualization.

2.2 Summarize and Arrange

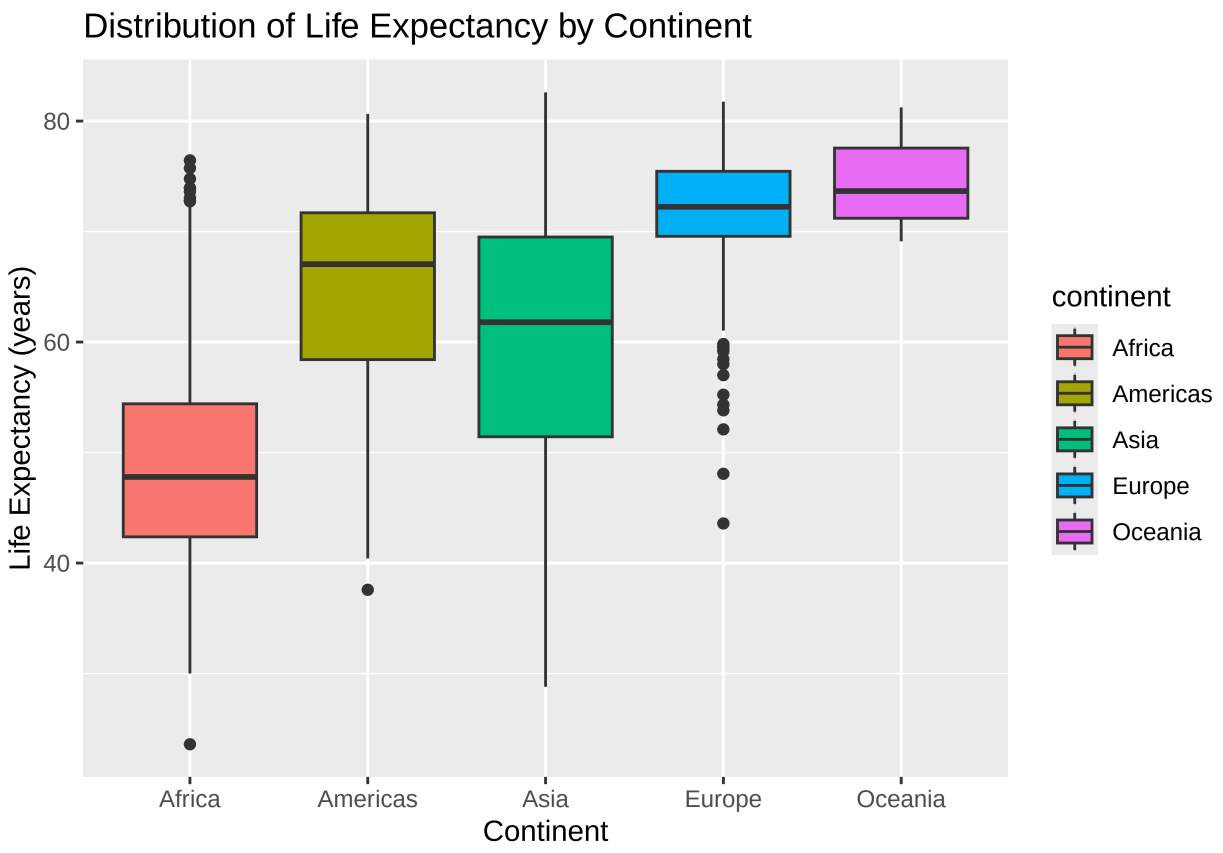
Let's practice combining summaries with sorting functions. arrange() helps reorder your results so the patterns are easier to read.

```
data1 <- data %>%
  filter(continent != "Asia") %>%
  group_by(country) %>%
  summarise(max(pop))

# We can also add the information to a new variable in the dataset
data2 <- data %>%
  filter(continent != "Asia") %>%
  group_by(country) %>%
  summarize(max_pop = max(pop, na.rm = TRUE)) # summarize the maximum population size logged for this c
# notice that data2 is automatically arranged by the **name of the country**

# you can change the arrangement to the **population**!
data2 <- data %>%
  filter(continent != "Asia") %>%
  group_by(country) %>%
  summarize(max_pop = max(pop, na.rm = TRUE)) %>%
  arrange(max_pop) # <------------- ADD THIS ------------------
# now, data2 is arranged by increasing maximum population size

# you can also look at the summary stats of a filtered item
china <- data %>%
  dplyr::filter(country == "China") %>%
  dplyr::summarise(mean_life_exp = mean(life_exp),
                   mean_gdp_percap = mean(gdp_percap))
```

3. Visualizing Summaries

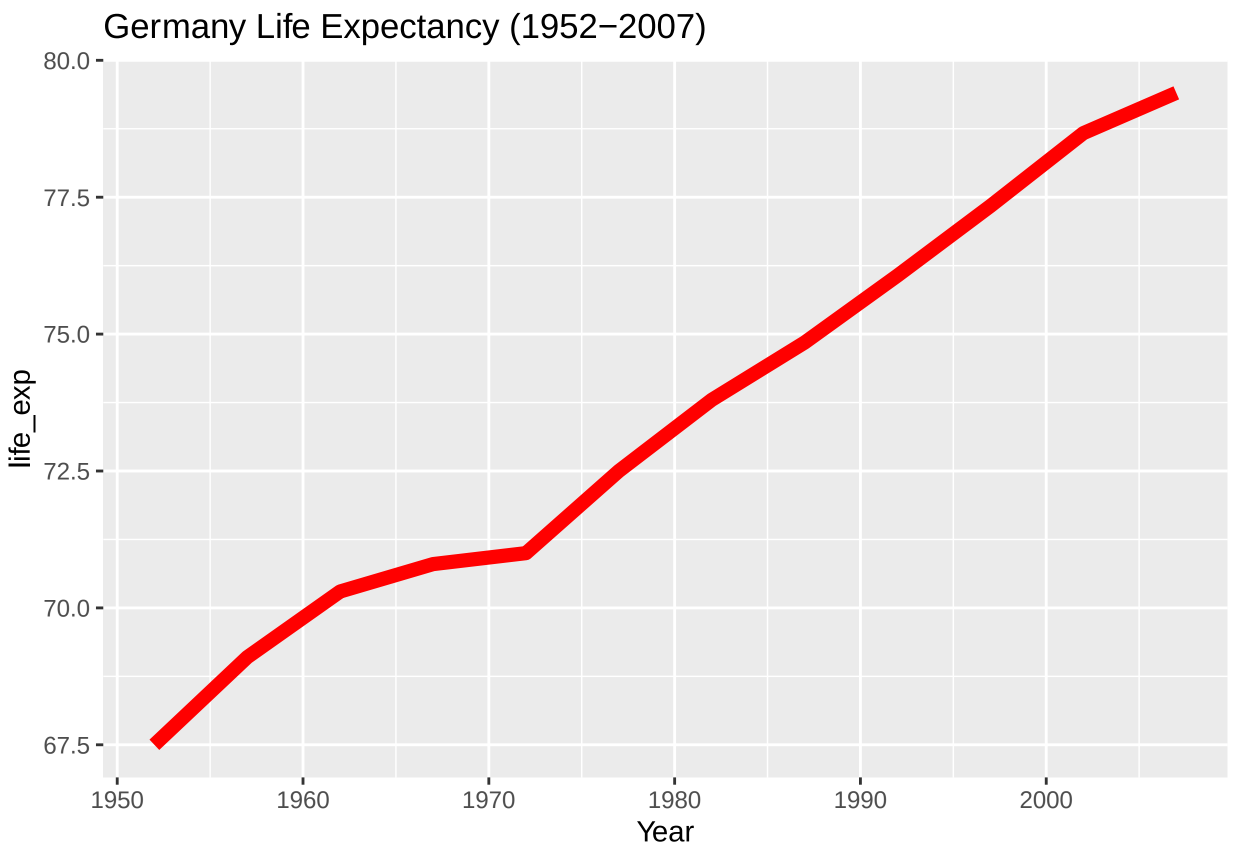
3.1 Boxplots – geom_boxplot()

```
# Show distribution of life expectancy by continent
ggplot(data, aes(x = continent, y = life_exp, fill = continent)) +
  geom_boxplot() +
  labs(title = "Distribution of Life Expectancy by Continent",
       x = "Continent", y = "Life Expectancy (years)")
```



3.2 Line plot – geom_line()

```
# let's first filter the data to Germany only
germany <- data %>%
   dplyr::filter(country == "Germany")

# Line plots are ideal for showing changes over time within a single country or group.
ggplot(germany, aes(x = year, y = life_exp)) +
  geom_line(color = "red", size = 2.5) +
  labs(title = "Germany Life Expectancy (1952-2007)",
       x = "Year")
```



Now we'll combine both grouping variables — continent and year — to visualize regional trends over time.

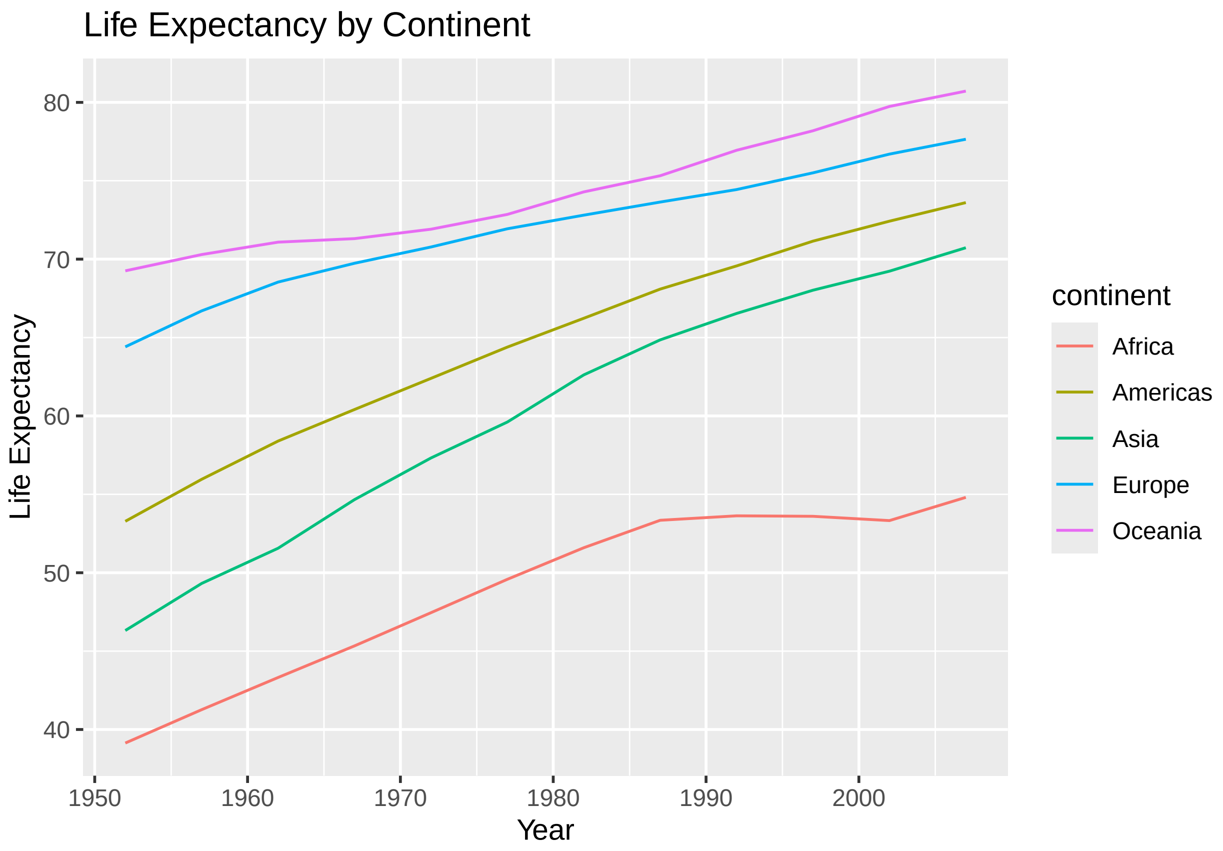
Summarize by continent, year

```
# summarizing relevant data
data3 <- data %>%
  group_by(continent) %>%
  summarize(mean_life_exp = mean(life_exp, na.rm = TRUE))
# but this only gives us a global mean per continent. I want to see change over time...

# View(data3)


# so if we add year to the grouping factor, we can create the per-year change for each continent
data3 <- data %>%
  group_by(continent, year) %>% # ADD YEAR!!!!!!!
  summarize(mean_life_exp = mean(life_exp, na.rm = TRUE))
```

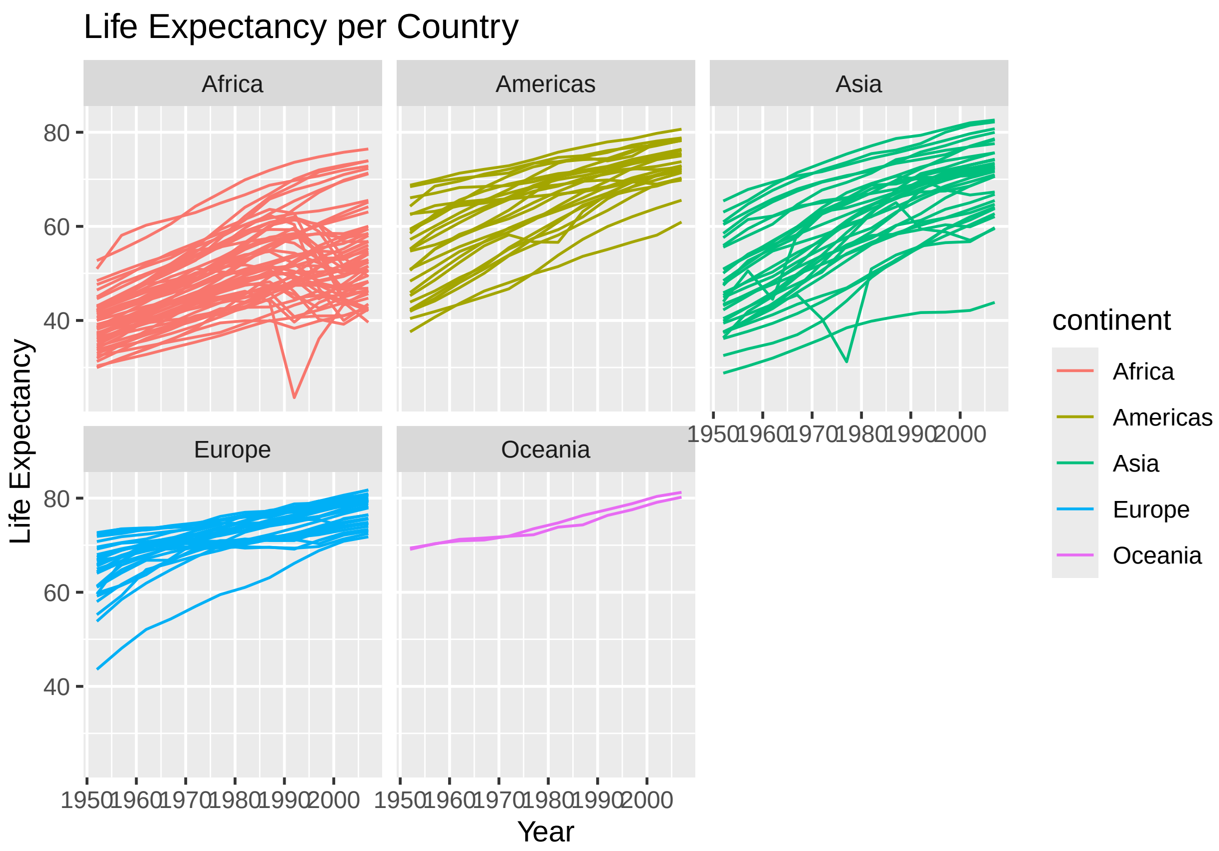
```
## `summarise()` has grouped output by 'continent'. You can override using the
## `.groups` argument.
```

```
# View(data3)
```

```r
# line plot to show change by continent
ggplot(data3, aes(x = year, y = mean_life_exp, color = continent)) +
  geom_line() +
  labs(title = "Life Expectancy by Continent",
       x = "Year", y = "Life Expectancy")
```



```r
# faceting by continent
ggplot(data, aes(x = year, y = life_exp, color = continent, group = country)) +
  geom_line() +
  facet_wrap(~continent) +
  labs(title = "Life Expectancy per Country",
       x = "Year", y = "Life Expectancy")
```



Faceting allows us to create small multiples, e.g., one panel per continent, making it easier to compare trends side by side.

Skills Application – Lab

Create a **new R Markdown file** in your project > Week 3.
Name it: `Lastname_Firstname_Week3`.

Follow the prompts below. For each plot, remember to include **labels**:
`labs(title = ..., x = ..., y = ...)`.

1. Summarize

- Use `group_by()` and `summarise()` to calculate an **average** of a numeric variable for each individual or each group.
- Then use `arrange()` to order results from highest to lowest (hint: desc() is largest to smallest values).

2. Visualize Group Means

- Make a bar plot (`geom_col()`) showing the **average** life expectancy for each group.
- Change the fill colors to something you like.

3. **Boxplot Challenge**

   - Create a boxplot comparing values across groups. Pick any numeric vairable and summarize it's mean across groups of observations (the groups can be social group, individual name, country, observer, and so on).
   - Try filtering to only a particular year.
   - Add axis labels and a title.

4. **Histogram vs Density**

   - Plot the distribution of life expectancy using both histogram and density.
   - Which gives you more insight? Write a short note in your Markdown file.

5. **Your Choice**

   - Pick *any numeric variable.*
   - Summarize by continent or year.
   - Make one visualization that best shows the differences.

This Week's Takeaway

- `group_by()` + `summarise()` create summary tables.
- Visualizations (bar, boxplot, line) can reveal group differences and changes over time.
- Next week: **joins** — combining datasets together.