ggplot – geoms, labels, scales

Professor Halterman

Michigan State University

PLS 397 Analyzing and Visualizing Data
Fall 2023

Today's quick checkin:

https://forms.gle/BLat7SguVcj1gXhx9

This should take about 5 minutes and the point is:

- To encourage you to think about things we cover in lecture and in the reading
- As a participation grade
- To help me understand where everyone is

Next week's reading

- Monday: Section 4 intro, plus 4.1–4.5
- Wednesday: 4.6, 4.7
- Following Monday: 5.2

Table of Contents

1. More geoms
2. geoms with more units
3. Labels
4. Scales
5. Coming soon...

Getting Started

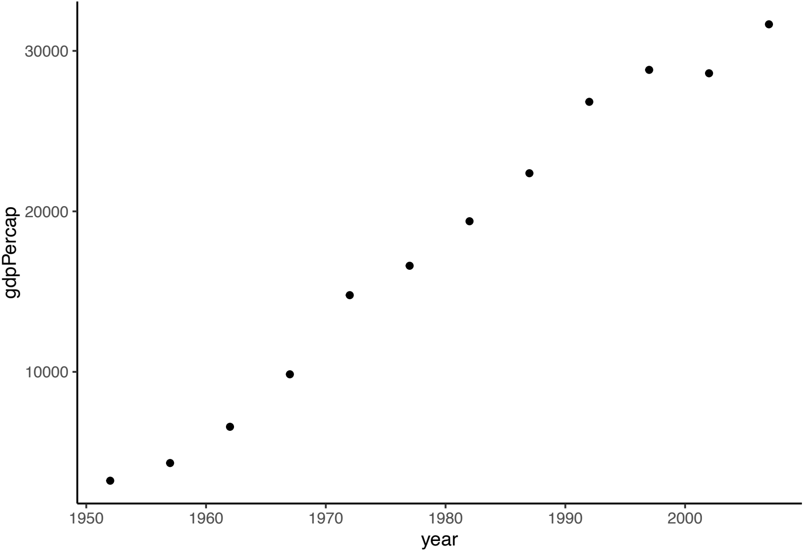
- Open the in-class exercise Rmd from D2L in Rstudio
- Run the beginning code to load the libraries we need and create the `japan` dataframe that we'll be using.
- Plot: Create a scatterplot showing the change in per-capita GDP over time.

A geom zoo

We can show the same aesthetics (x = year, y = GDP per capita) in many different ways:

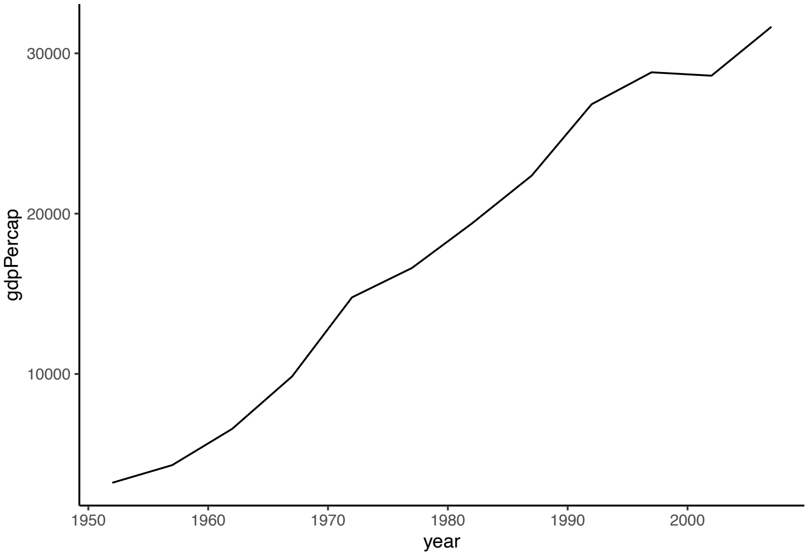
- `geom_point`
- `geom_line`
- `geom_col` (a bar plot)
- combining points and lines

+ geom_point()



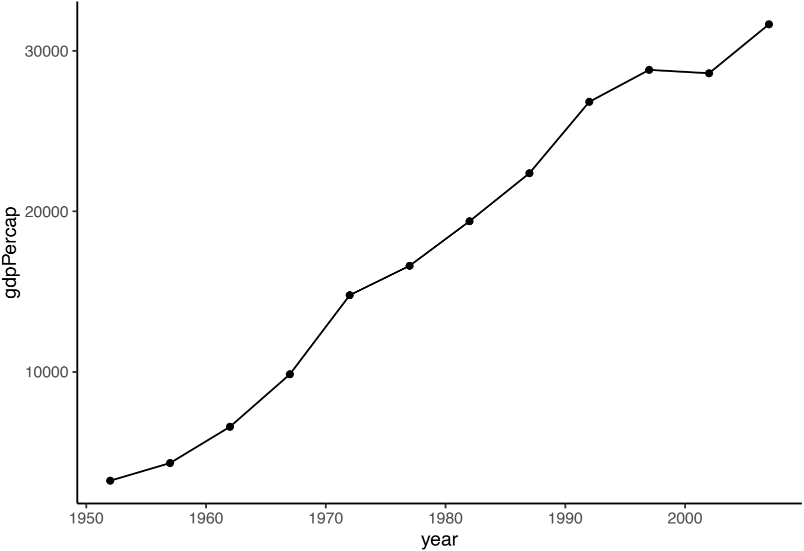
30000
20000
10000
gdpPercap
1950
1960
1970
1980
1990
2000
year

+ geom_line()

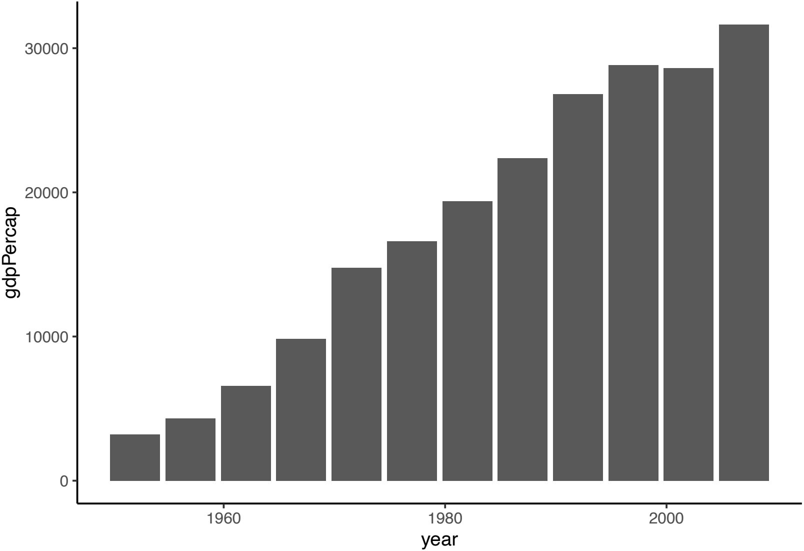


30000
20000
10000
gdpPercap
1950
1960
1970
1980
1990
2000
year

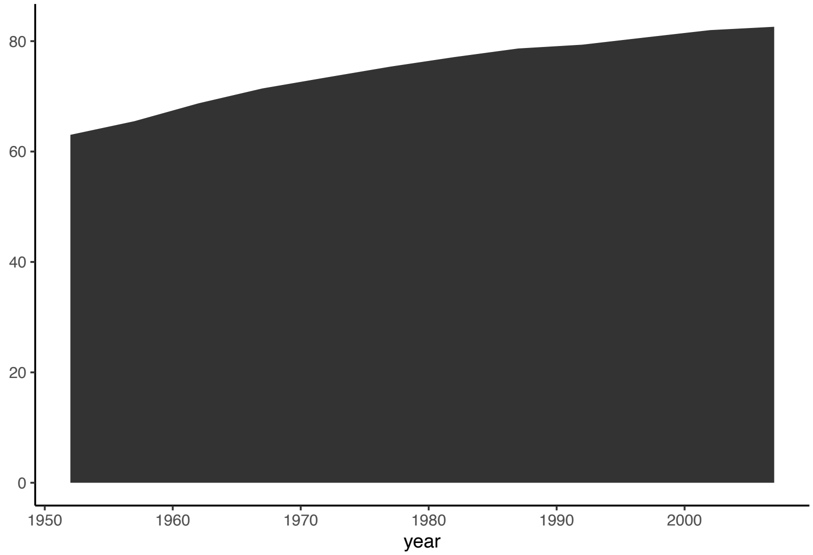
+ geom_line() + geom_point()



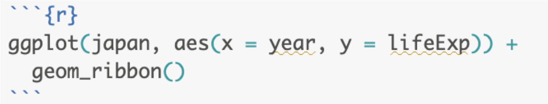
+ geom_col()



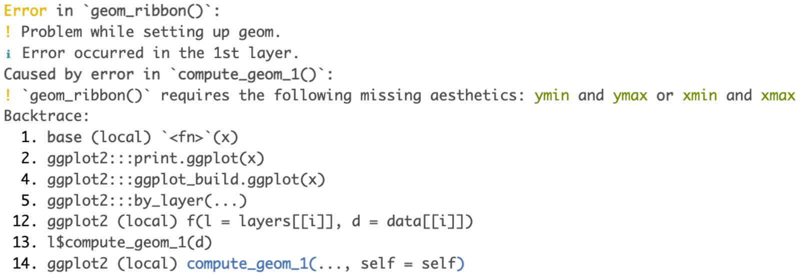
+ geom_ribbon()



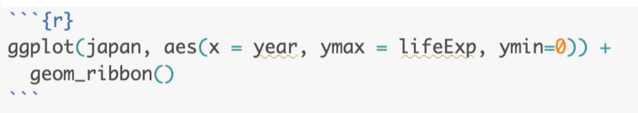
+ geom_ribbon() is tricky!



~~~
```{r}
ggplot(japan, aes(x = year, y = lifeExp)) +
  geom_ribbon()
```
~~~

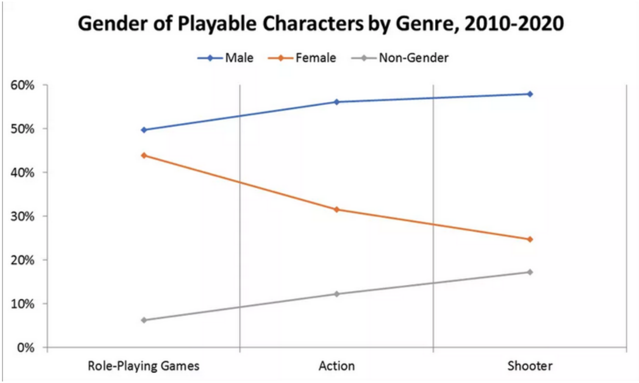


```
Error in `geom_ribbon()`:
! Problem while setting up geom.
ℹ Error occurred in the 1st layer.
Caused by error in `compute_geom_1()`:
! `geom_ribbon()` requires the following missing aesthetics: ymin and ymax or xmin and xmax
Backtrace:
  1. base (local) `<fn>`(x)
  2. ggplot2:::print.ggplot(x)
  4. ggplot2:::ggplot_build.ggplot(x)
  5. ggplot2:::by_layer(...)
 12. ggplot2 (local) f(l = layers[[i]], d = data[[i]])
 13. l$compute_geom_1(d)
 14. ggplot2 (local) compute_geom_1(..., self = self)
```



~~~
```{r}
ggplot(japan, aes(x = year, ymax = lifeExp, ymin=0)) +
  geom_ribbon()
```
~~~

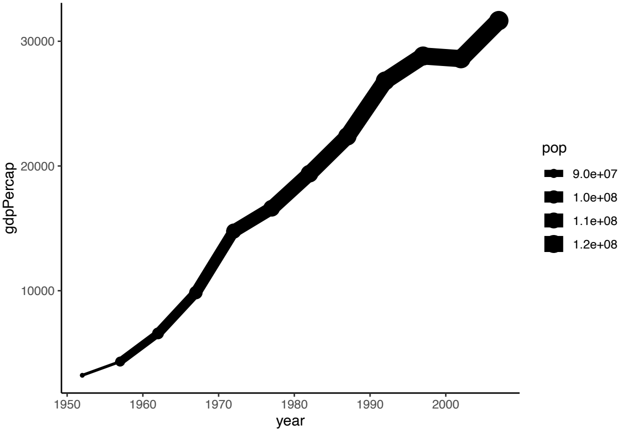
Which one is best?



- Line graphs emphasize change over time.
- Bar charts are common, but they can be tricky with height vs. volume

adding aes

Make a line + point plot showing GDP per capita over time, with the size set by population.

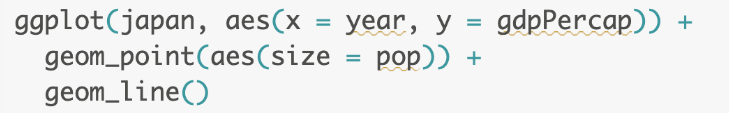


Looks weird! What happened?

Changing aesthetics per geom

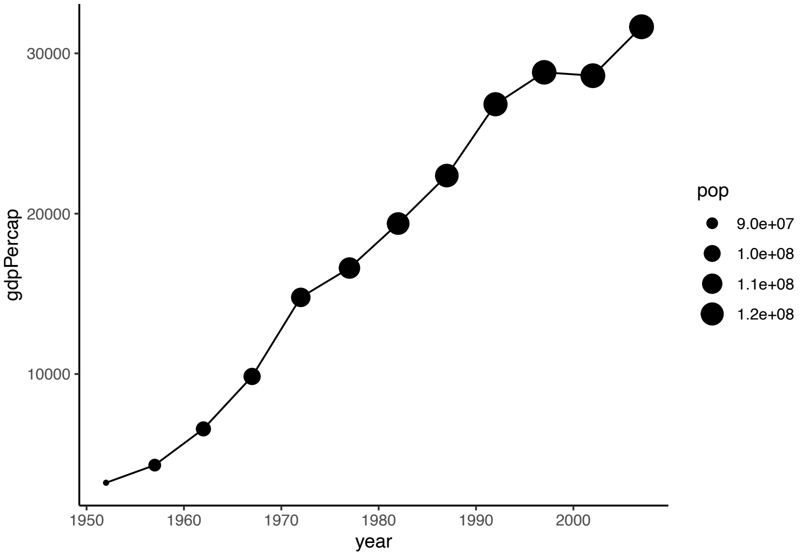
Remember: we set the `aes`thetics at the beginning, and those get used in each subsequent `geom`.

But we can set aes separately within the geom:



```
ggplot(japan, aes(x = year, y = gdpPercap)) +
  geom_point(aes(size = pop)) +
  geom_line()
```

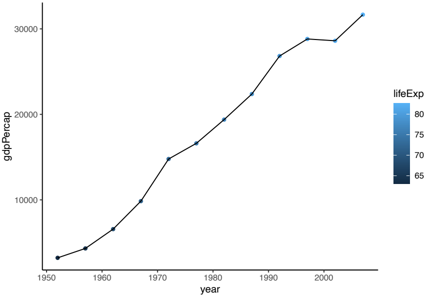
Per-geom aes

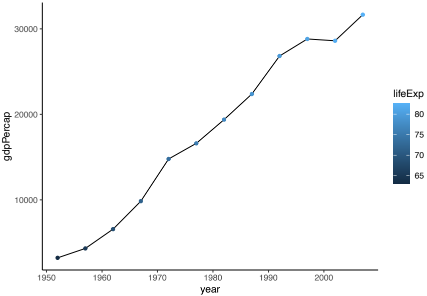


30000
20000
10000
gdpPercap
1950
1960
1970
1980
1990
2000
year
pop
9.0e+07
1.0e+08
1.1e+08
1.2e+08

Layer order

Now let’s make a figure that has both dots and lines, with color set by life expectancy.

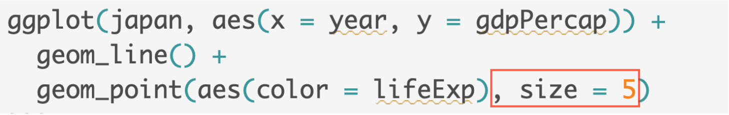




Problem: the points are really small!

Changing visual features **without** aes

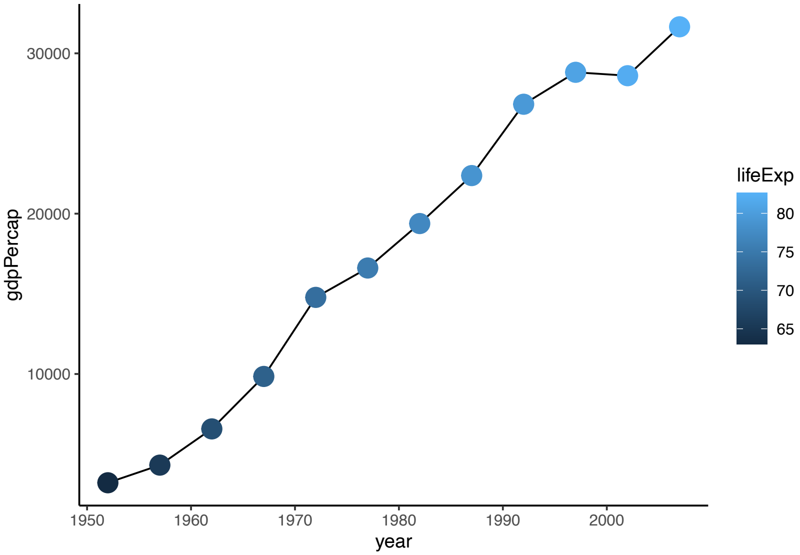
- We usually want to set visual features (e.g. color, size) using our data.
- But sometimes we want to manually set these attributes.
- To do that, we can use our regular arguments to `aes`, but outside aes.
- Example:



```
ggplot(japan, aes(x = year, y = gdpPercap)) +
  geom_line() +
  geom_point(aes(color = lifeExp), size = 5)
```

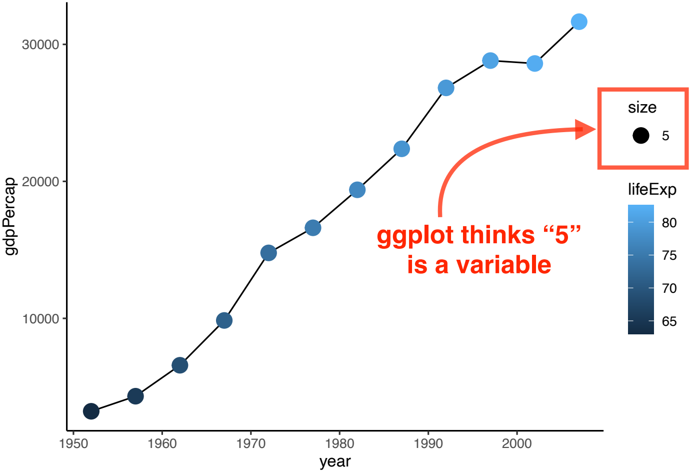
- Including `aes(size = 5)` can make ggplot confused.

Manual point size



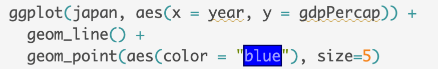
Manual point size–if we put it in `aes`

We don't want to put it inside `aes`!

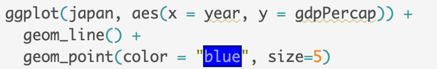


ggplot thinks "5" is a variable

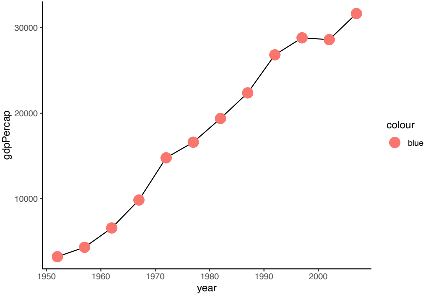
It’s weirder with color

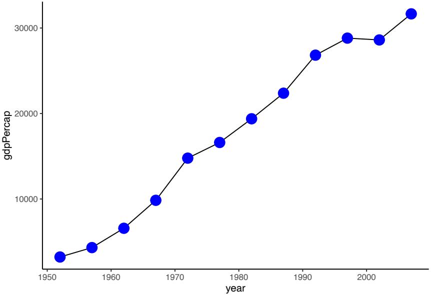


```
ggplot(japan, aes(x = year, y = gdpPercap)) +
  geom_line() +
  geom_point(aes(color = "blue"), size=5)
```



```
ggplot(japan, aes(x = year, y = gdpPercap)) +
  geom_line() +
  geom_point(color = "blue", size=5)
```





Takeaway: if you’re manually setting color/size/etc, do it outside `aes`.

Table of Contents

1. More geoms

2. geoms with more units

3. Labels

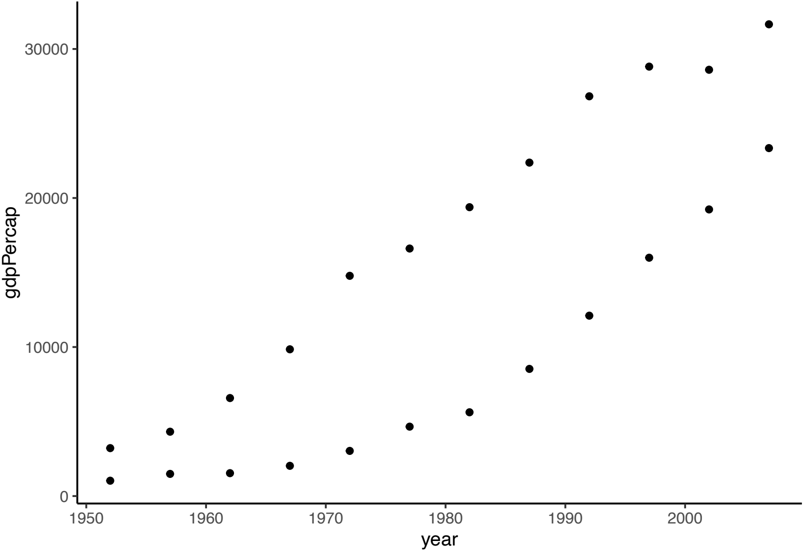
4. Scales

5. Coming soon...

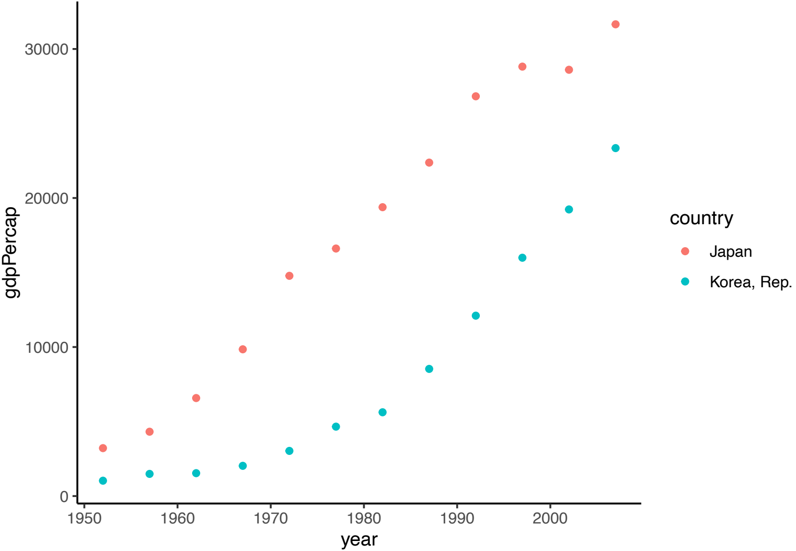
More complicated data

- Let’s experiment with some slightly more complicated data.
- Find the code in your .Rmd file that creates `jsk`–a dataframe with data for both Japan and South Korea.
- Using that dataframe, make a simple scatterplot showing GDP over time.
- Once you make the figure, what’s wrong with it? Think of a way to improve it, then write the code.

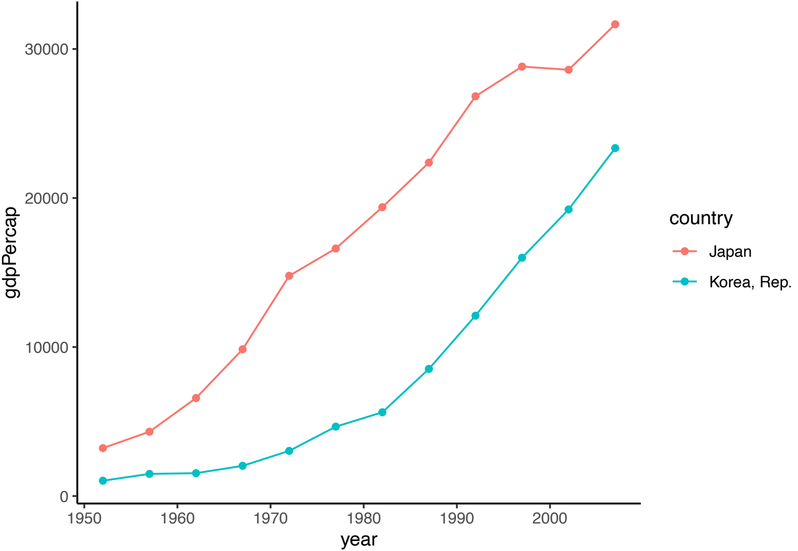
Simple Japan and South Korea scatterplot



Japan and South Korea colored scatterplot

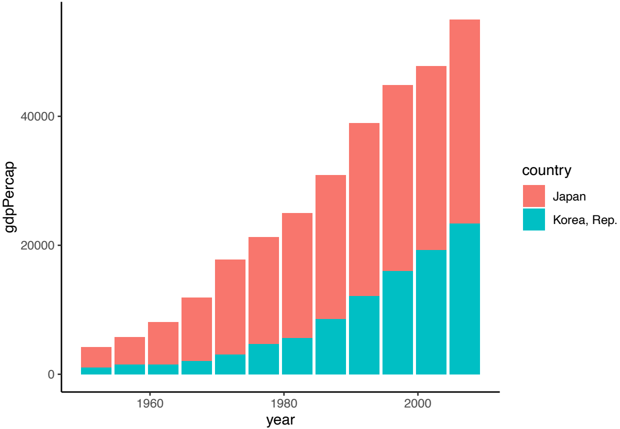


Japan and South Korea colored scatterplot with lines



Making a bar plot

Now let's make a bar plot. (Remember what the geom is?)



How do we interpret the heights of these bars?

What happened?

- We told ggplot to use year as the x-axis and GDP per capita as the y axis.
- It stacked the GDPs of the two.
- When would this be useful? When would it not?
- If we want the bars next to each other, we can tell ggplot explicitly: `geom_col(position="dodge")`
- Try that out and see which you prefer.
- (If you finish that, try `geom_col(position="fill")`. What does that give us?)

Side-by-side bars

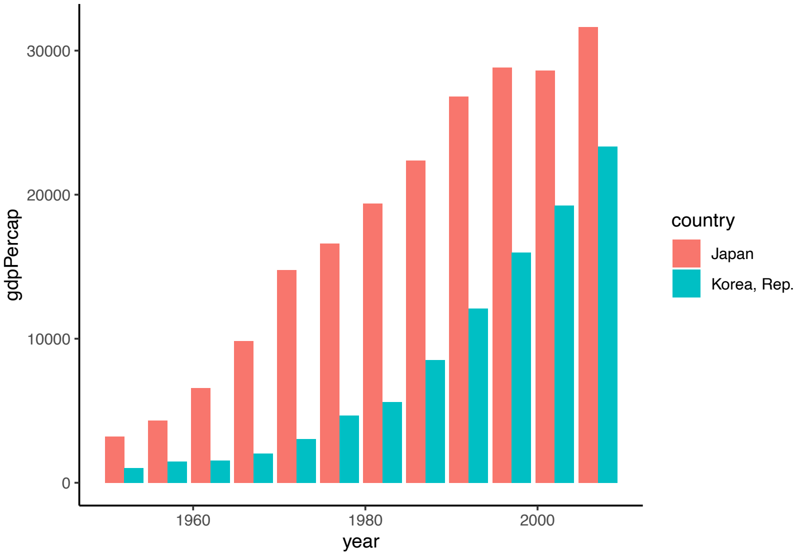
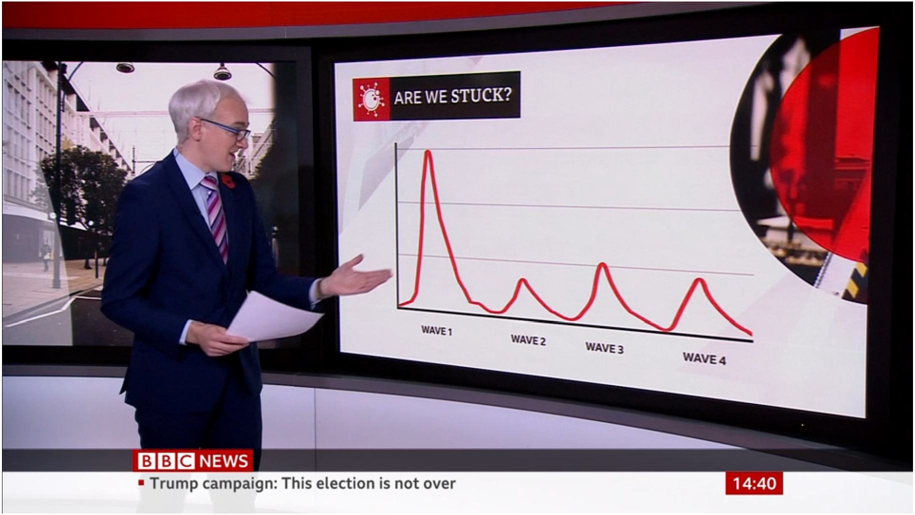


Table of Contents

1. More geoms
2. geoms with more units
3. Labels
4. Scales
5. Coming soon...

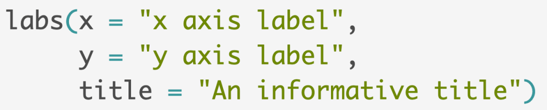
Labeling your figure is crucial!



Credit: r/dataisugly

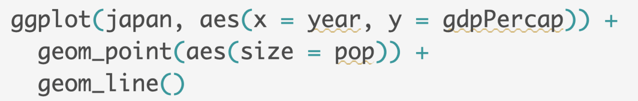
ggplot's labs layer

ggplot has a `labs` layer to easily add axis labels and a title to your plot:



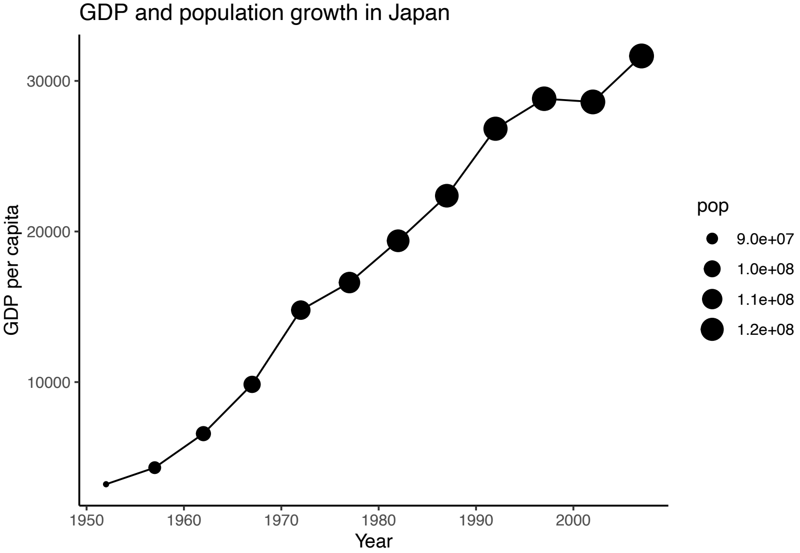
```
labs(x = "x axis label",
     y = "y axis label",
     title = "An informative title")
```

Let's revisit the plot where we show year, GDP per capita, and population from above:



```
ggplot(japan, aes(x = year, y = gdpPercap)) +
  geom_point(aes(size = pop)) +
  geom_line()
```

Add a `labs` layer to provide a title and axis labels to the plot. (Hint: remember all layers get added with a + sign)


GDP and population growth in Japan
GDP per capita
30000
20000
10000
1950
1960
1970
1980
1990
2000
Year
pop
9.0e+07
1.0e+08
1.1e+08
1.2e+08

Advanced `labs` options

- But notice that our legend title is still a raw variable name!
- The `labs` function can take other arguments besides `x`, `y`, and `title`.
- In this case, we need to set a label for the attribute shown in the legend. Which aesthetic does the legend show?
- Other `labs` options:
  - subtitle
  - caption
- Experiment with those and see what happens!

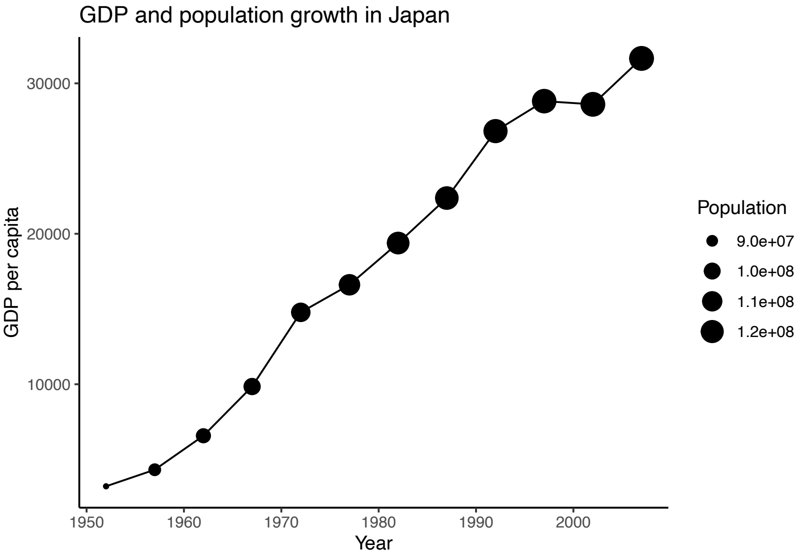

GDP and population growth in Japan
GDP per capita
30000
20000
10000
1950
1960
1970
1980
1990
2000
Year
Population
9.0e+07
1.0e+08
1.1e+08
1.2e+08

Table of Contents

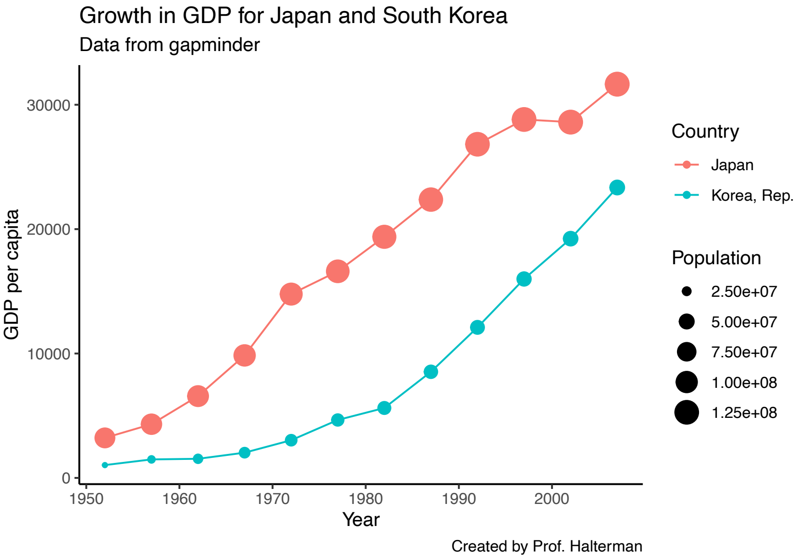
1. More geoms
2. geoms with more units
3. Labels
4. Scales
5. Coming soon...

One more plot

Let's re-create our plot of our `jsk` data that shows two sets of points connected by a line: one for each country, with time on the x-axis and GDP on the y axis. The lines should be colored by country and the points should be sized by population. Make sure to label the plot!

Hint: we did something very similar above, and one of the secrets behind programming is to copy your own code whenever you can!

Our plot



Growth in GDP for Japan and South Korea
Data from gapminder
GDP per capita
30000
20000
10000
0
1950
1960
1970
1980
1990
2000
Year
Country
Japan
Korea, Rep.
Population
2.50e+07
5.00e+07
7.50e+07
1.00e+08
1.25e+08
Created by Prof. Halterman

Adding scales

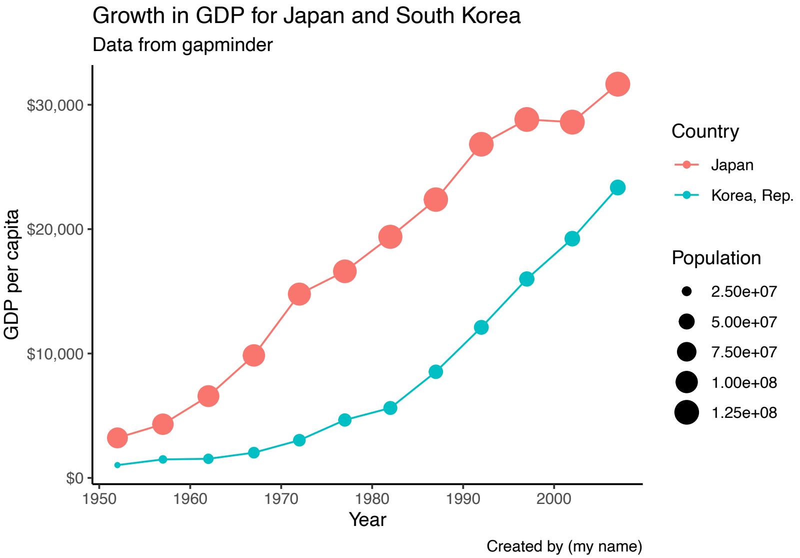
- Take a look at the last figure you made.
- The axes are labeled now, which is great. But what about the numbers on the axes?
- ggplot lets you customize the axis and legend numbers by specifying a scale.
- Note that these are tricky: I always have to look up the docs.
- Example: `+ scale_y_continuous(), scale_size_continuous(), scale_x_discrete()...)`

Changing the y axis

Growt
Data fr
GDP per capita
30000
20000
10000
0
1950

- Let's start with the y axis: what would make make the numbers clearer?
- Now try adding `scale_y_continuous(labels = scales::label_dollar())`
- (If you're ahead, experiment with using + `scale_y_log10()` or `scale_y_reverse()` instead.)

Way nicer!

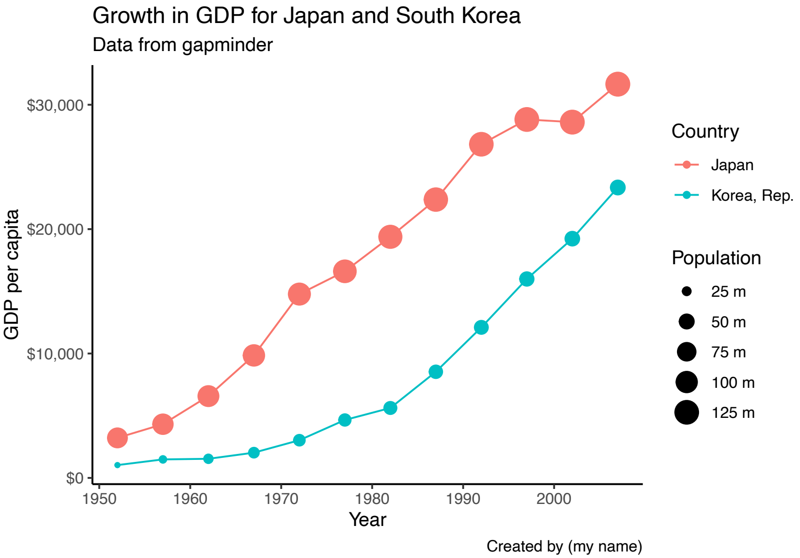


Growth in GDP for Japan and South Korea
Data from gapminder
GDP per capita
$30,000
$20,000
$10,000
$0
1950
1960
1970
1980
1990
2000
Year
Country
Japan
Korea, Rep.
Population
2.50e+07
5.00e+07
7.50e+07
1.00e+08
1.25e+08
Created by (my name)

Other scales

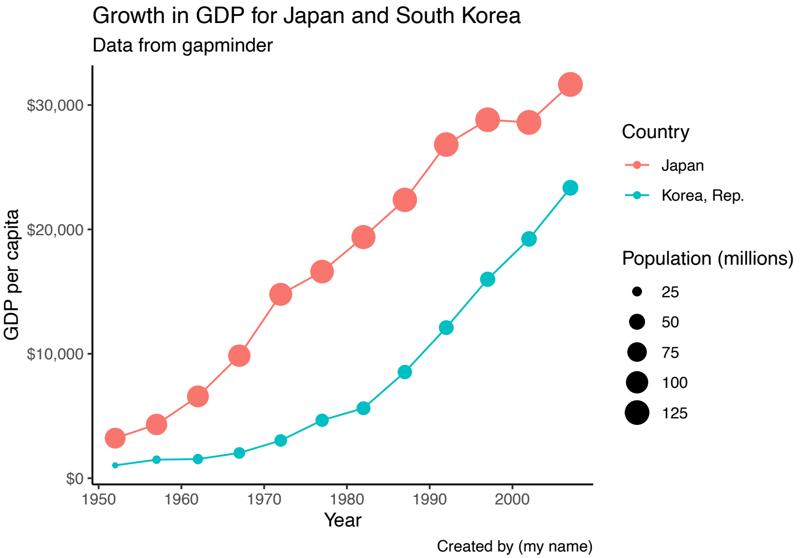
- The `labels` argument changed how we formatted the dollar values on the y axis.
- Next, we probably want to make the population values less hideous.
- To change the scale for the y axis, we used `scale_y_continuous`. How should we change the scale for size?
- `scale_size_continuous(labels = scales::unit_format(unit = "m", scale = 1e-6))`

Formatting population in millions



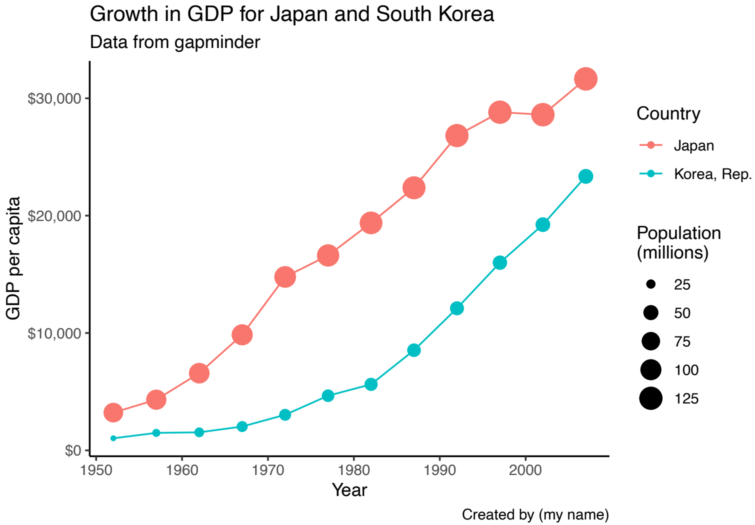
Another idea: leave out the m

How would we do this?



Even fancier!

Hint: adding `\n` to your label will make R return to a new line.



Labels vs. breaks

- So far, we've been changing the label argument to `scale`
- But we can also change the breaks
- Try `scale_x_continuous(breaks = seq(1950, 2007, by = 5))`

Changed year breaks

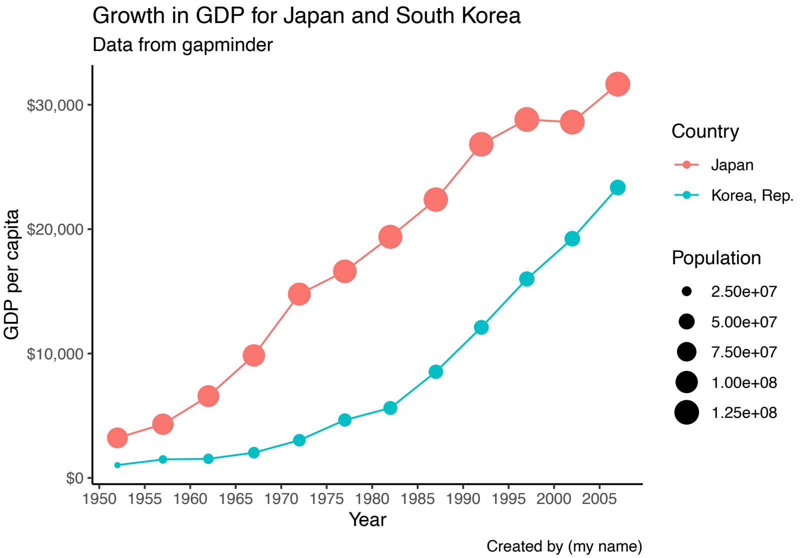


Table of Contents

1. More geoms
2. geoms with more units
3. Labels
4. Scales
5. Coming soon...

Coming soon

- Working with discrete data
- Facets
- Setting x and y axis limits
- Annotating data with labels
- Modifying data for plotting