Dynamic documents in R

Making figures in Quarto

2023-04-28

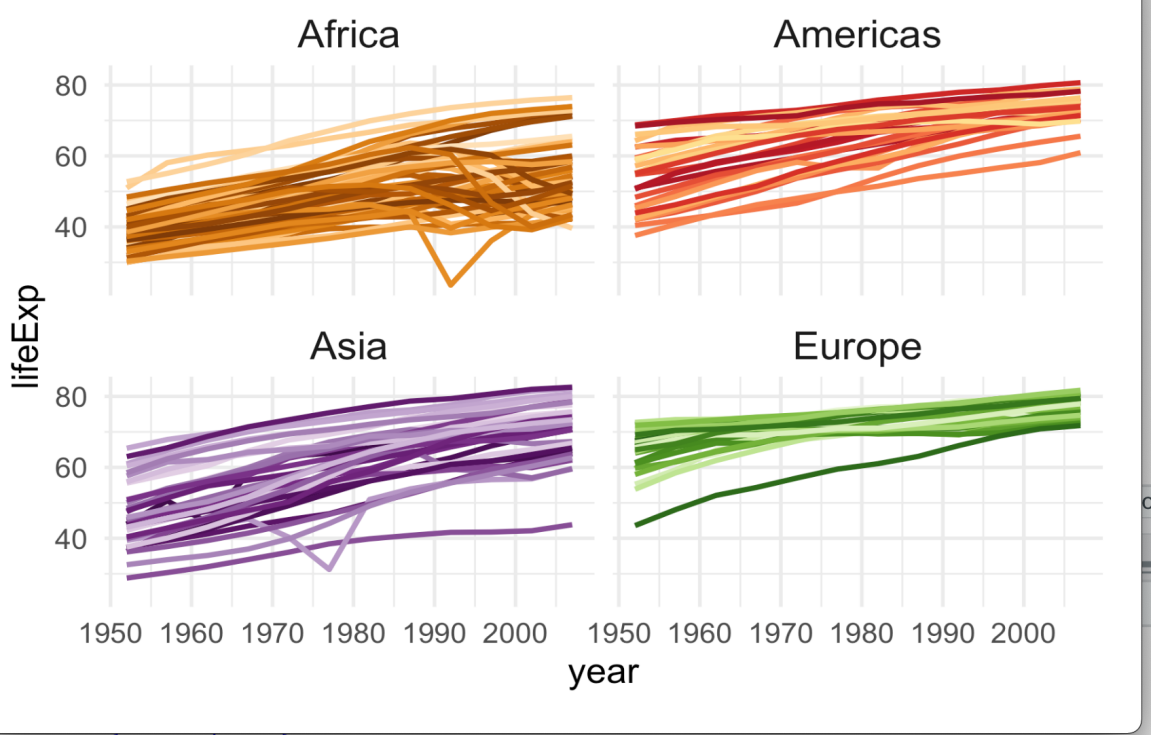
rmarkdown_basics - main - RStudio

Go to file/function | Addins | rmarkdown_basics

exercises.Rmd

Knit | Run

RStudio: Notebook Output

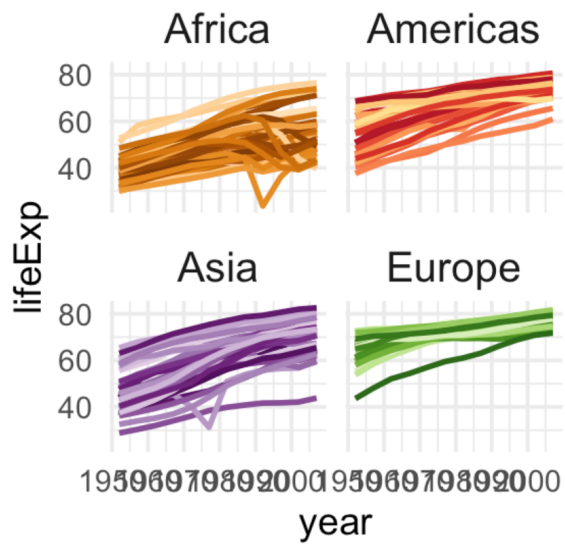



```
+   facet_wrap(~ continent) +
+   # `country_colors` is an object included in gapminder
+   scale_color_manual(values = country_colors) +
+   theme_minimal(14) +
+   theme(strip.text = element_text(size = rel(1.1)))
>
```

Environment | History | Connections | Git | Tutorial

Files | Plots | Packages | Help | Viewer

```
  scale_color_manual(values = country_colors) +
  theme_minimal(14) +
  theme(strip.text = element_text(size = rel(1.1)))
```



Your turn 7

What goes into a figure?

Absolute size: physical dimensions (inches, cm, etc)

Pixel size: no inherent size!

Resolution: pixels per inch (ppi) or dots per inch (dpi); links absolute & pixel size

Pointsize: absolute text size (1 pt = 1/72 inch)

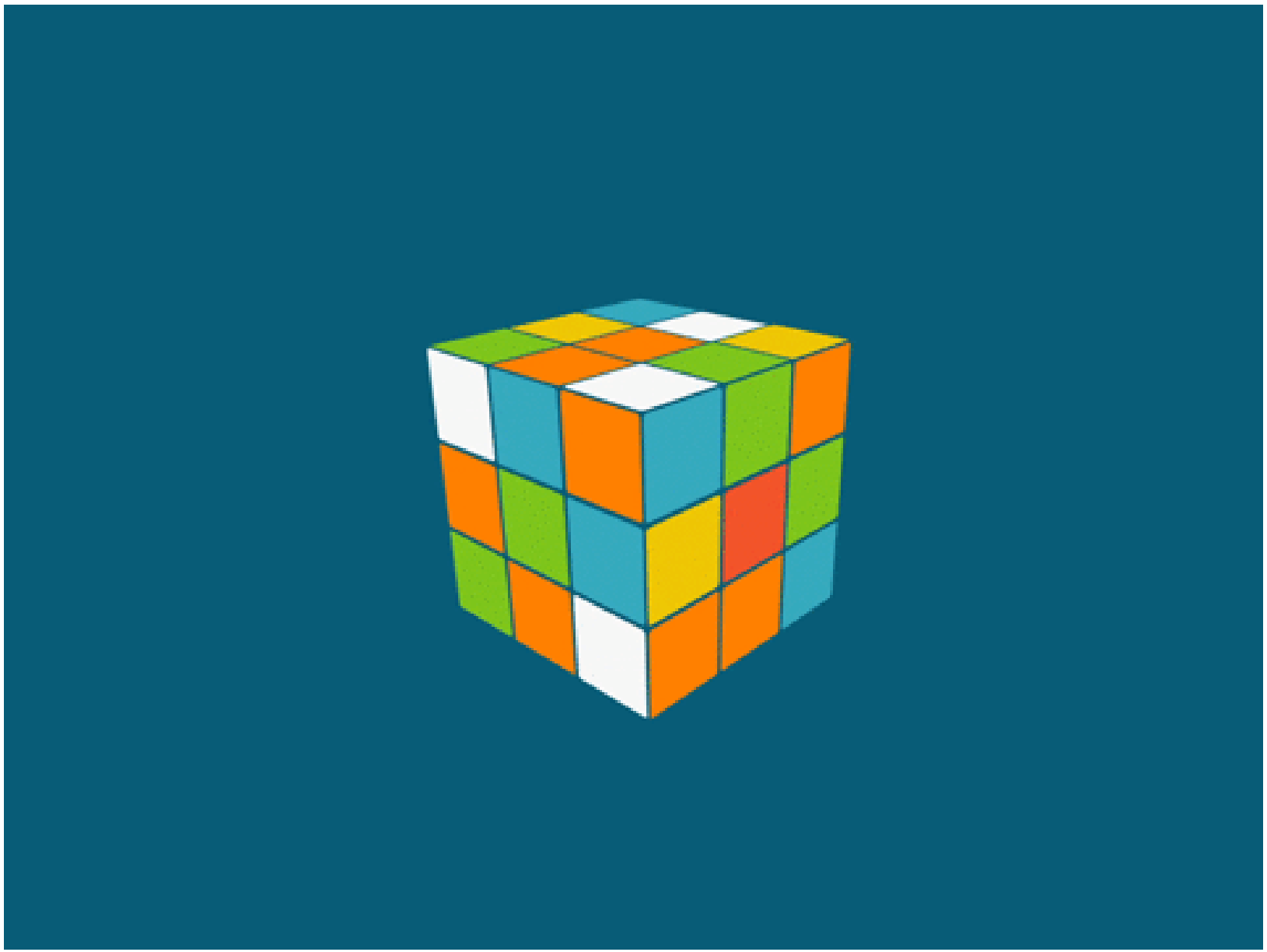
Plot theming and aesthetics: choices about text size, line size, margins, and so on.

Essential options

`fig.height`	Rendered figure height (in)
`fig.width`	Rendered figure width (in)
`fig.asp`	Rendered figure aspect ratio (use with ONE of height or width)
`dpi`	Resolution
`out.height`	Figure container height (in)
`out.width`	Figure container width (in)

See all of them at http://yihui.name/knitr/options/

Tweaking figure options



Getting a figure to look good in RStudio, Word, and slides

A few reasonable defaults in your YAML

```yaml
---
knitr:
  opts_chunk:
    echo: true
    warning: false
    message: false
    dev: "ragg_png"
    dpi: 320
    out.width: "80%"
    fig.width: 6
    fig.asp: 0.618
    fig.retina: 2
    fig.align: "center"
    fig.show: "hold"
---
```

Inspired by R for Data Science and Jumping Rivers

Plot scaling

```
1 ggplot(mpg, aes(displ, hwy)) + geom_point()
```

figure.width = 4

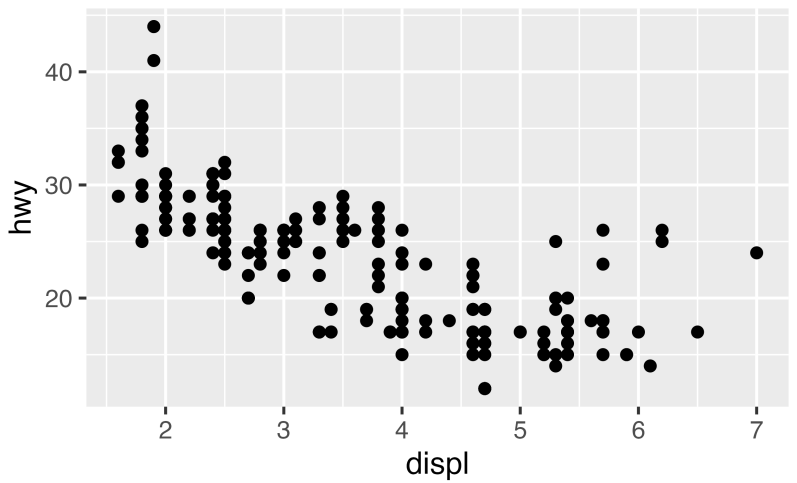


figure.width = 6

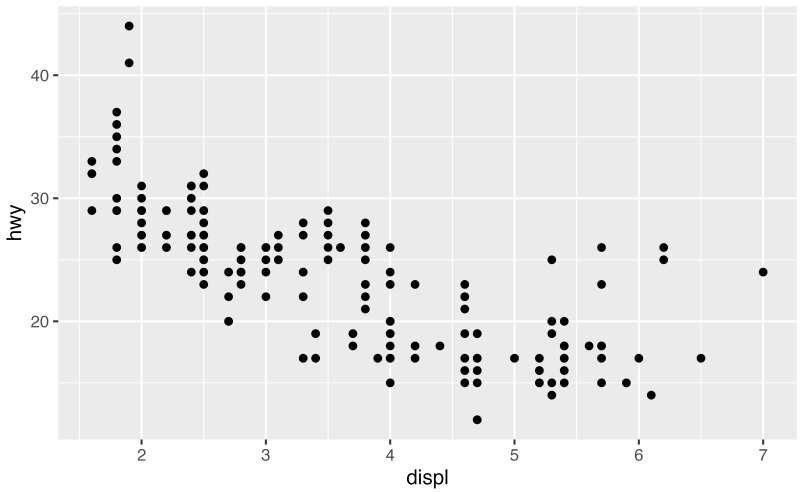


figure.width = 8

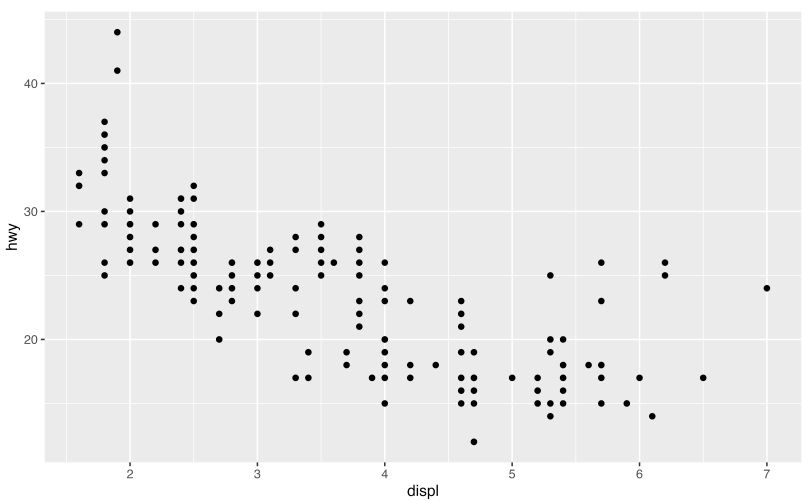
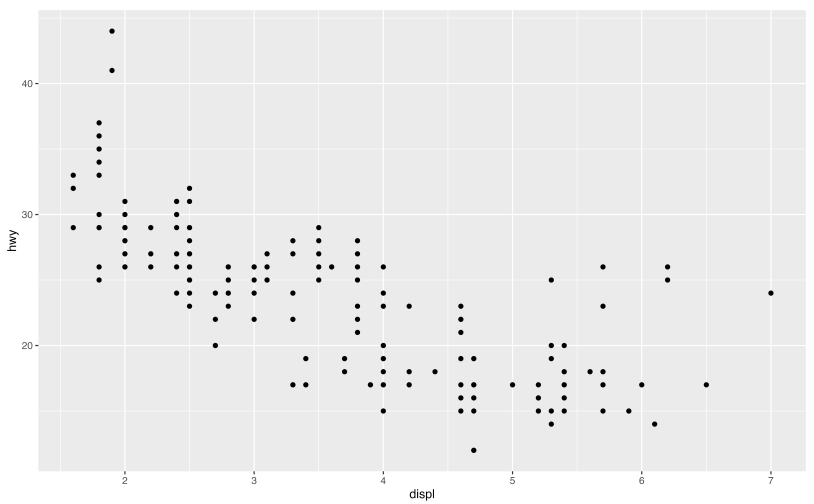


figure.width = 10



Scaling saved files

`ggsave()`: Set the `scale` option

`ragg::agg_png()`: Set the `scaling` option

Warning: these arguments work differently from one another!

A few reasonable defaults in your YAML

```yaml
knitr:
  opts_chunk:
    echo: true
    warning: false
    message: false
    dev: "ragg_png"
    dpi: 320
    out.width: "80%"
    fig.width: 6
    fig.asp: 0.618
    fig.retina: 2
    fig.align: "center"
    fig.show: "hold"
```

Inspired by R for Data Science and Jumping Rivers

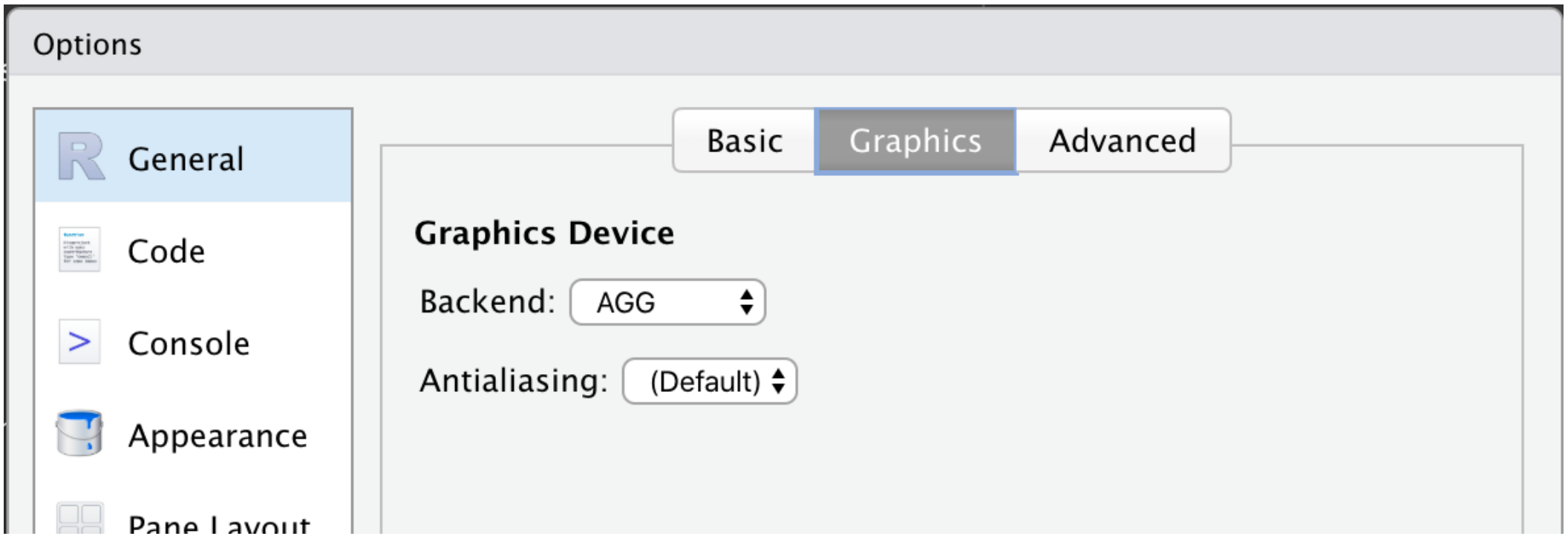
ragg: AGG Graphic Devices

Faster than grDevices or Cairo

Better system font access and text rendering

System independent rendering

Setting ragg as your default in RStudio



This sets the default for the *viewer*, not Quarto

Your Turn 1

Using the chunk option defaults we discussed, set the global chunk options in the YAML metadata

Render this document, and take a look at the first three figures. Do you like how they look?

Modify `fig.width` for each chunk until you're satisfied

What affects ggplot2 sizing?

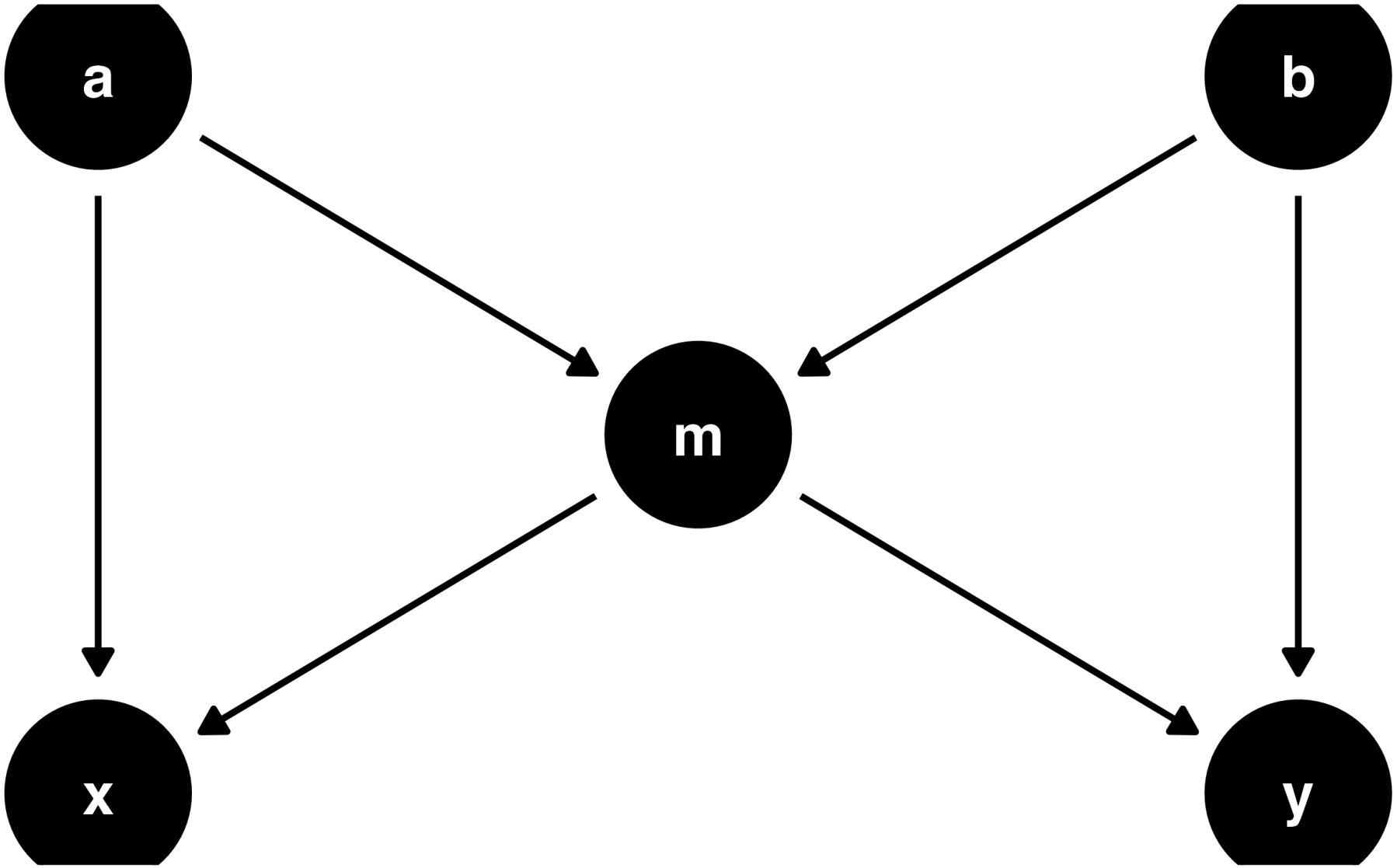
1. geoms
2. themes
3. scales and axes
4. clipping

Theme sizing

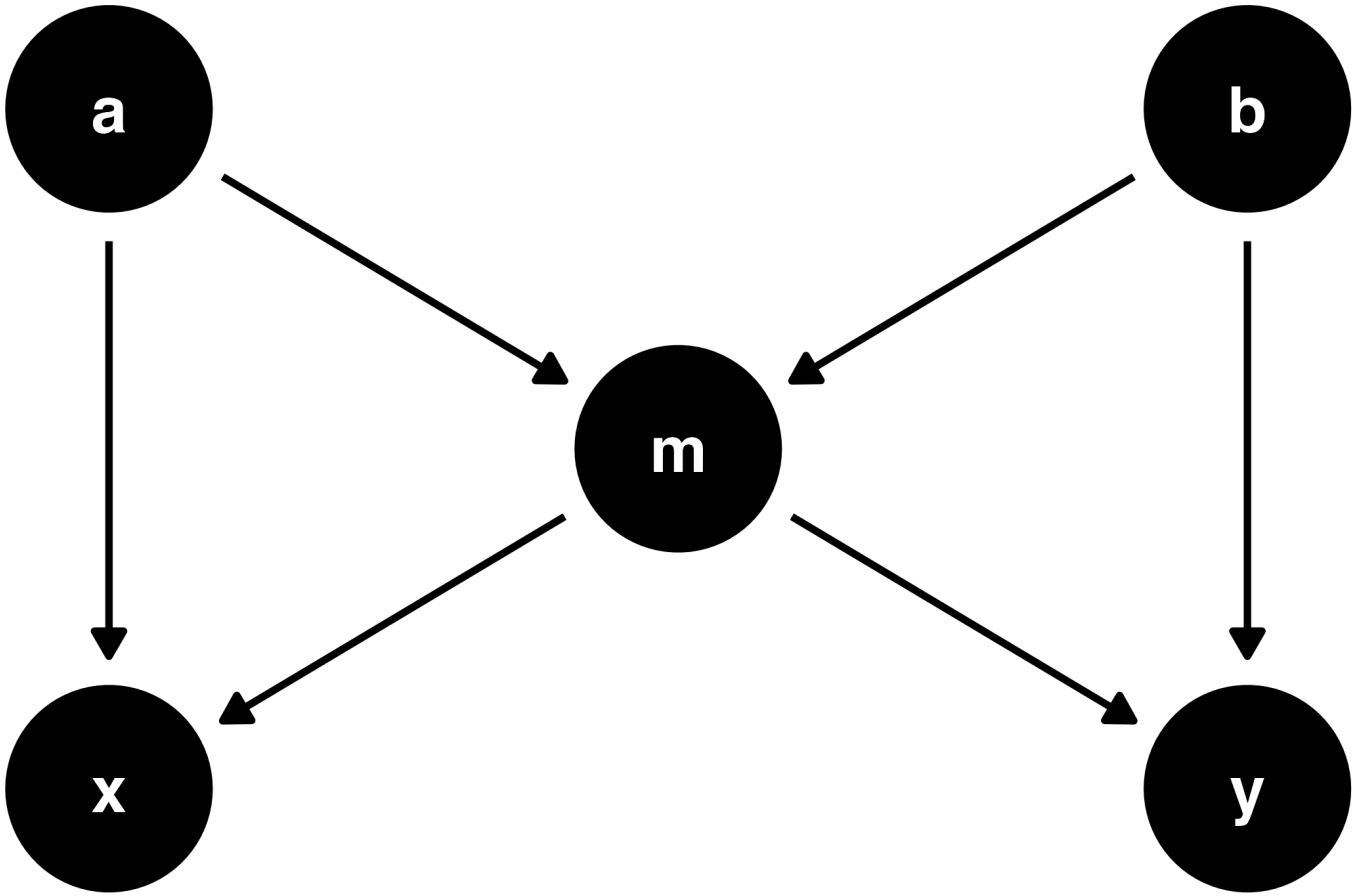
ggplot2 themes all have a `base_size` argument, e.g. `theme_minimal(base_size = 14)`

Consider well-proportioned cowplot themes, e.g. `*theme_minimal_grid()`

Expanding scales (`fig.width = 4`)

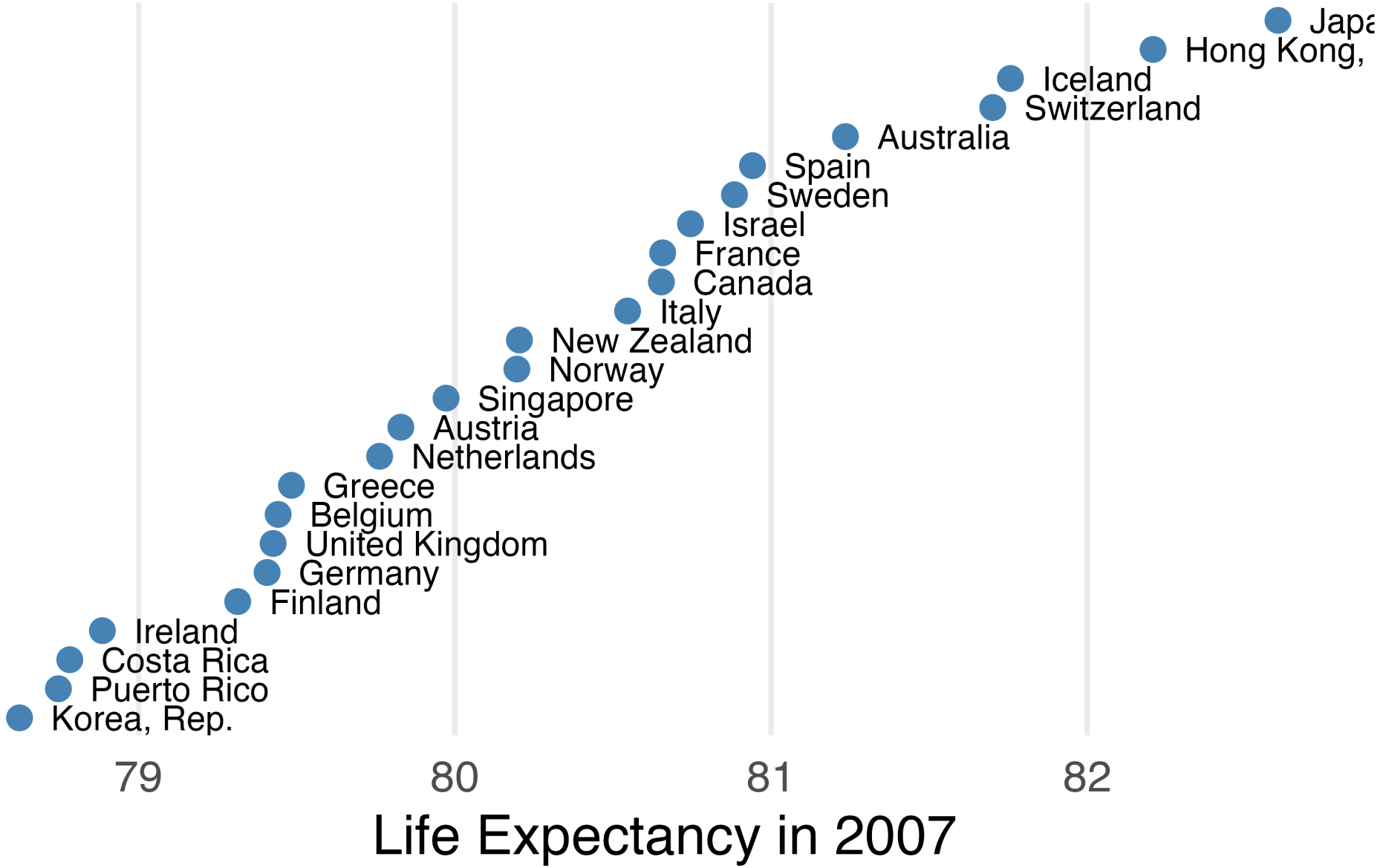


```r
library(ggdag, warn.conflicts = FALSE)
ggdag(butterfly_bias()) +
  theme_dag() +
  scale_x_continuous(expand = expansion(.2)) +
  scale_y_continuous(expand = expansion(.2))
```

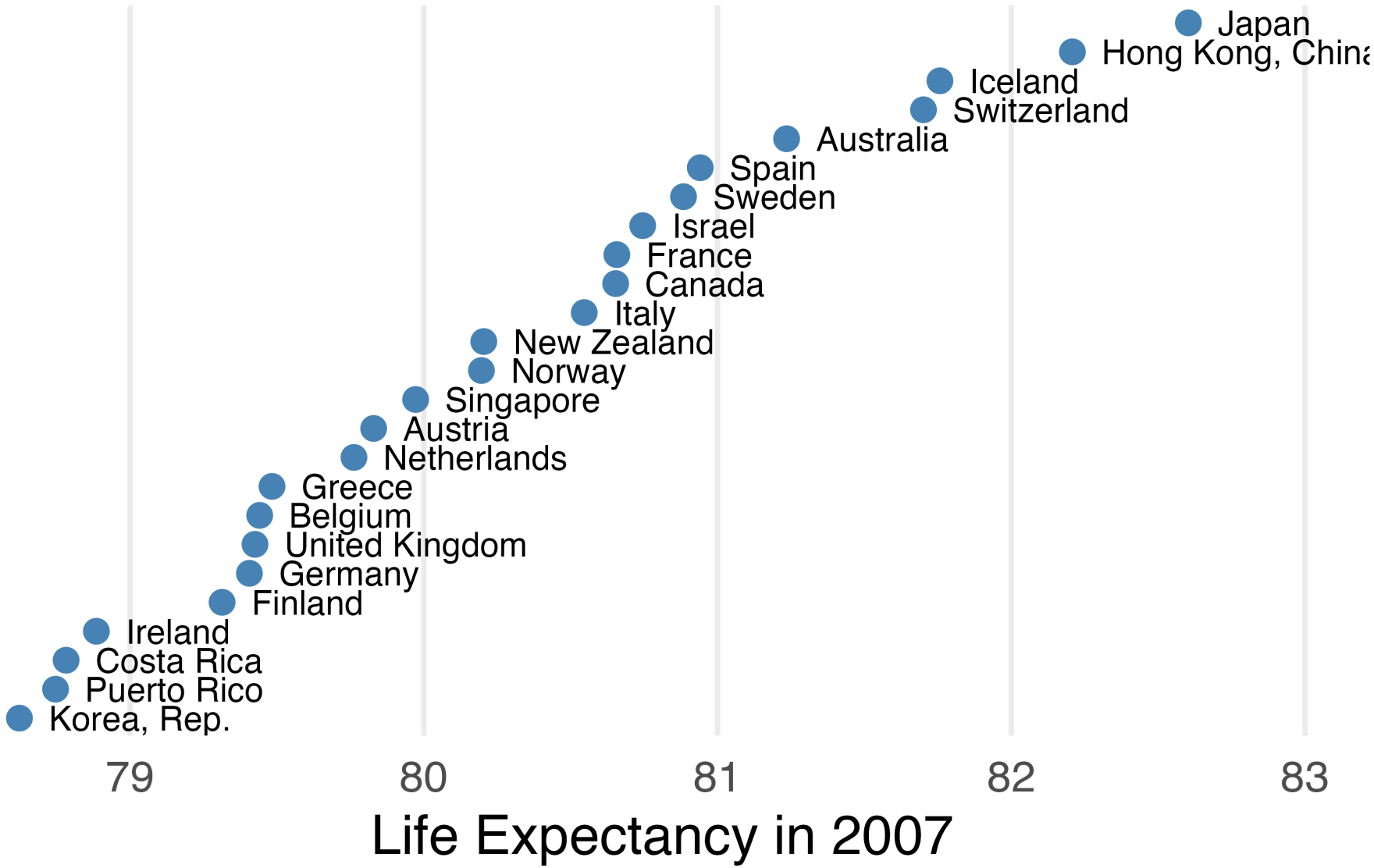

a
b
m
x
y

```r
p <- gapminder |>
  filter(year == 2007) |>
  slice_max(lifeExp, n = 25) |>
  mutate(country = fct_rev(fct_inorder(fct_drop(country)))) |>
  ggplot(aes(lifeExp, country)) +
  geom_point(size = 3, color = "steelblue") +
  geom_text(aes(label = country), hjust = 0, nudge_x = .1, size = 3.5) +
  theme_minimal(16) +
  theme(
    axis.title.y = element_blank(),
    axis.text.y = element_blank(),
    panel.grid.minor = element_blank(),
    panel.grid.major.y = element_blank()
  ) +
  xlab("Life Expectancy in 2007")
```

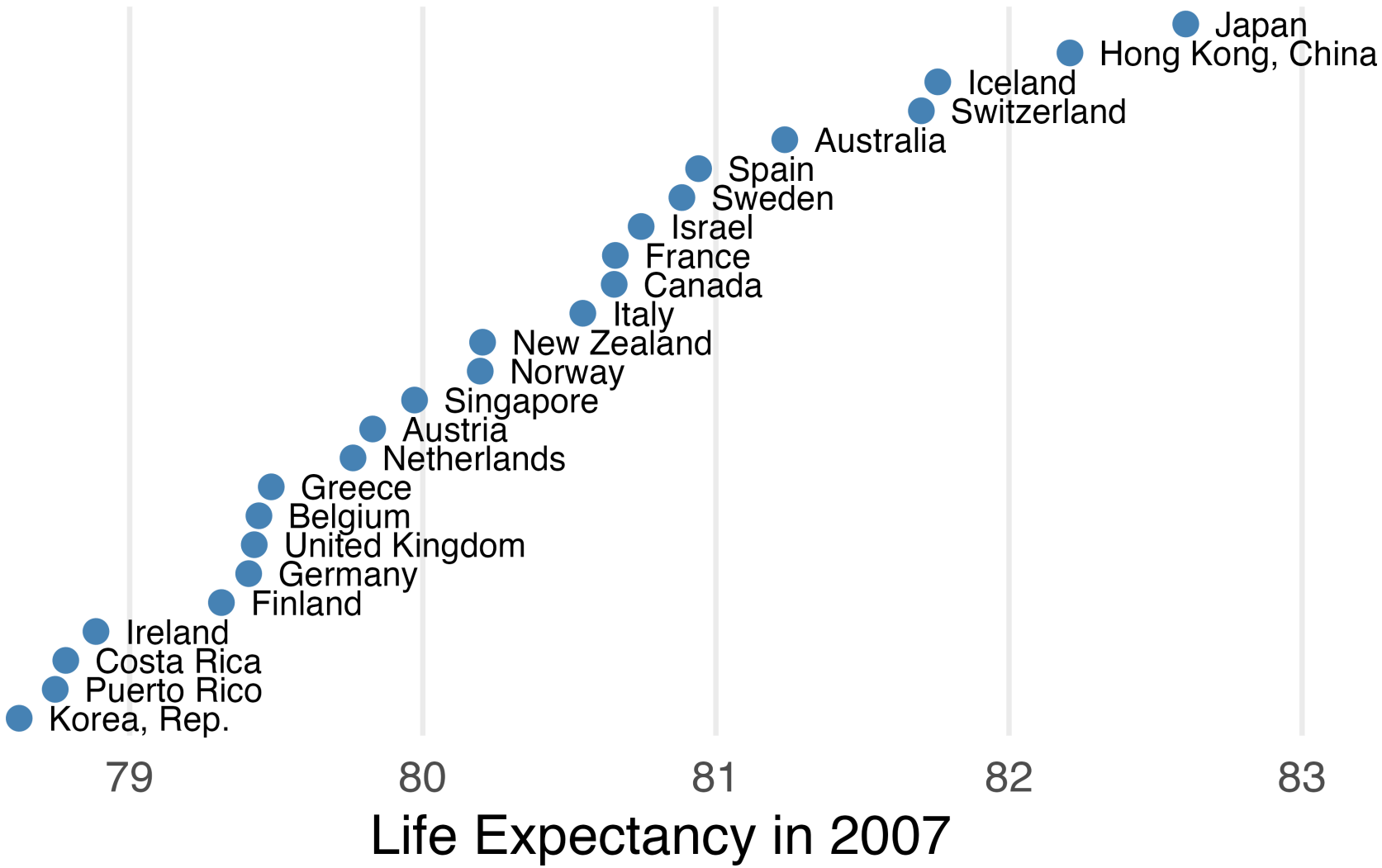
```
1 p
```


Japa
Hong Kong,
Iceland
Switzerland
Australia
Spain
Sweden
Israel
France
Canada
Italy
New Zealand
Norway
Singapore
Austria
Netherlands
Greece
Belgium
United Kingdom
Germany
Finland
Ireland
Costa Rica
Puerto Rico
Korea, Rep.
79
80
81
82
Life Expectancy in 2007

```r
p +
   xlim(NA, 83)
```


Japan
Hong Kong, China
Iceland
Switzerland
Australia
Spain
Sweden
Israel
France
Canada
Italy
New Zealand
Norway
Singapore
Austria
Netherlands
Greece
Belgium
United Kingdom
Germany
Finland
Ireland
Costa Rica
Puerto Rico
Korea, Rep.
79
80
81
82
83
Life Expectancy in 2007

```r
p +
  xlim(NA, 83) +
  coord_cartesian(clip = "off")
```



Specify where Quarto writes figures

`fig.path = “folder/prefix-”

Use with chunk labels!


here:
find your
PATH!
seriously, just here.
"/NOT/HERE"
here!
DATA
HERE
files
IT'S ALL RIGHT HERE!
Horst '18

Detour: The here package

Find files from the root up, particularly with *RStudio projects*

```
here("data", "file.csv")
```

Detour: The here package

Really convenient with Quarto, which sets a local directory

See Why should I use the here package when I'm already using projects?

How do I create an RStudio Project again?

In RStudio: `File > New Project`

Or, in the console:

```
usethis::create_project("path/to/projec
```

```
my_project
|-- data
    |-- data.csv
|-- figures
    |-- figure1.png
|-- reports
    |-- manuscript.Rmd
|-- R
    |-- read_data.R
|-- my_project.Rproj
```

```
source("../R/read_data.R")

read_csv("data/data.csv")

ggsave("../figures/figure1.png")
```

```
my_project
|-- data
    |-- data.csv
|-- figures
    |-- figure1.png
|-- reports
    |-- manuscript.Rmd
|-- R
    |-- read_data.R
|-- my_project.Rproj
```

```
source(here("R", "read_data.R"))

read_csv(here("data", "data.csv"))
```

Why here?

Works from the project up

Robust to other ways people open and run your code

Writes paths safely across operating systems

Your Turn 2

Globally set `fig.path` to "figures/figure-". This will tell knitr to create figures in the "figures" folder with a prefix of "figures-".

Render this document and take a look at the images in the `figures` folder.

Cross-referencing figures

1. A figure caption (`#| fig.cap = "Plot title"`)
2. A named code chunk (`#| label: fig-chunk-name`)
3. Reference with `@fig-chunk-name`

Cross-referencing figures

Also sets `fig.alt = fig.cap` (See Writing Alt Text for Data Visualization)

Your Turn 3

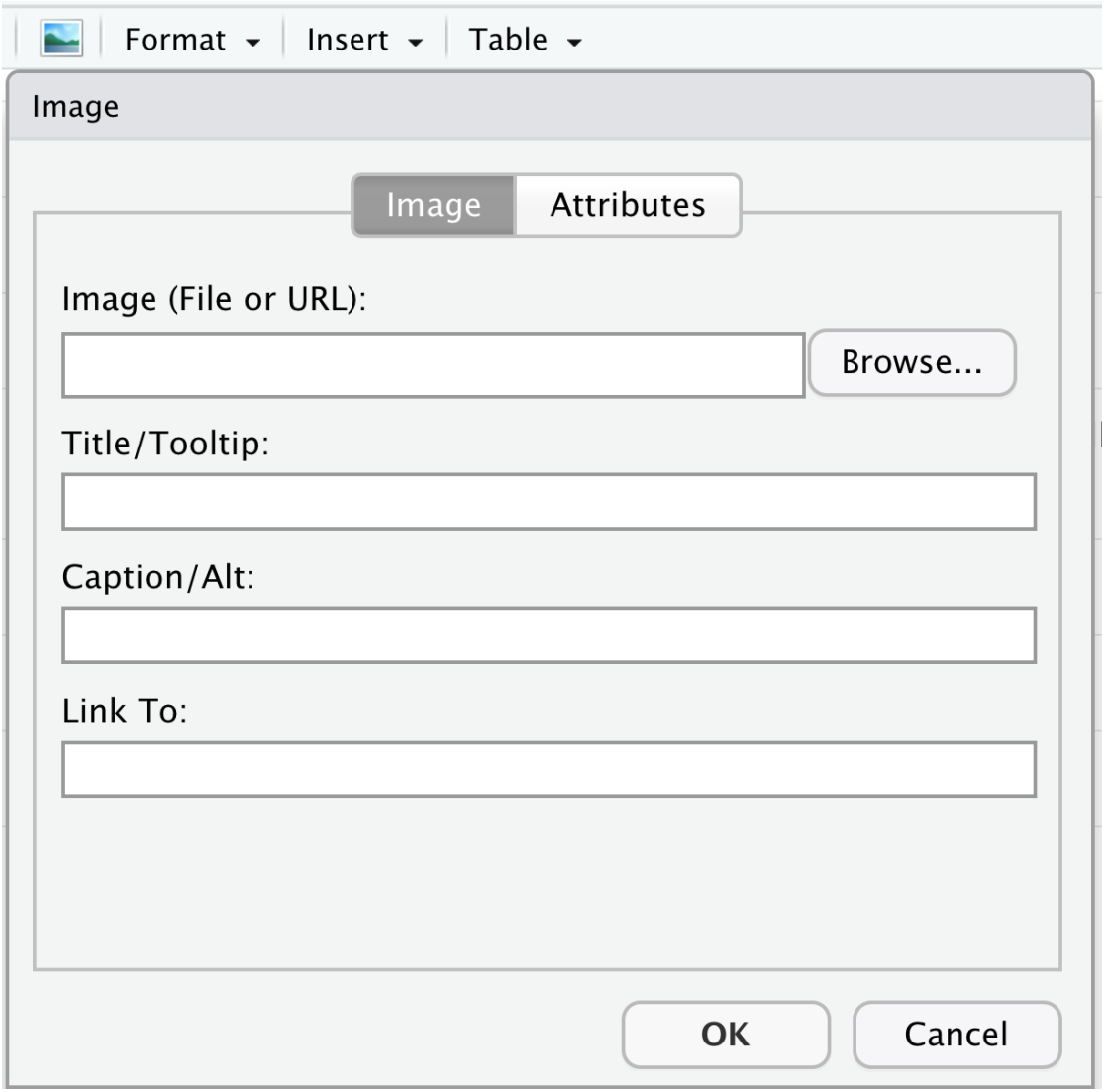
Cross-reference one of the figures above.

Including external images

```
knitr::include_graphics("path/to/image")
+out.width
```

`include_graphics()` also accepts URLs

Including external images




```
![Alt text](path/to/image)
```

Your Turn 4

Include `external_img/r_rollercoaster.png` in the code chunk below.

Render

Let's change a few chunk options: 1) Add a chunk name 2) Set `fig.alt` describing the image 3) Modify `out.width` to use a different percentage than the default.

Render again

Resources

Quarto: Figures: Quarto documentation figures

Quarto: Article Layout: Quarto documentation article layout

Taking Control of Plot Scaling: A detailed blog on understanding scaling