R Markdown & ggplot2

Iegor Vyshnevskyi

2022-11-24

Objectives

- Learning R Markdown PowerPoint presentation builder
- Learning the basic usage of the `ggplot2` package
- Practicing creating some basic plots using `ggplot2`
- Practicing creating some presentation

Thanks

- Special thanks go to Dr. Jae Yeon Kim class on Data Visualization, Kieran Healy's book (2019) on data visualization and Hadley Wickham's book on ggplot2. I adopted their material.
- For more theoretical discussions - read The Grammar of Graphics by Leland Wilkinson.

Setup

- Check your `dplyr` package is up-to-date by typing `packageVersion("dplyr")`. If the current installed version is less than 1.0, then update by typing `update.packages("dplyr")`. You may need to restart R to make it work.

```
ifelse(packageVersion("dplyr") >= 1, # Condition
  "The installed version of dplyr package is greater than or equal to 1.0.0", # TRUE
  update.packages("dplyr") # FALSE
)
```

```
## [1] "The installed version of dplyr package is greater than or equal to 1.0.0"
```

```
if (!require("pacman")) install.packages("pacman")
```

```
## Loading required package: pacman
```

```
pacman::p_load(
  tidyverse, # the tidyverse framework
  here, # computational reproducibility
  gapminder, # toy data
  ggthemes, # additional themes
  ggrepel, # arranging ggplots
  patchwork, # arranging ggplots
  broom, # tidying model outputs
  gtsummary,
  ggfortify,
  rmarkdown
)
```

Toy data

```
library(gapminder)
```

```
gapminder
## # A tibble: 1,704 × 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## # … with 1,694 more rows
#head(gapminder,5)
#tail(gapminder,5)
```

Data exploration

There are so many different ways of looking at data in R. Can you discuss the pros and cons of each approach? Which one do you prefer and why?

Data exploration (approach 1)

```
str(gapminder)
## tibble [1,704 × 6] (S3: tbl_df/tbl/data.frame)
##  $ country  : Factor w/ 142 levels "Afghanistan",..: 1 1
1 1 1 1 1 1 1 1 ...
##  $ continent: Factor w/ 5 levels "Africa","Americas",..:
3 3 3 3 3 3 3 3 3 3 ...
##  $ year     : int [1:1704] 1952 1957 1962 1967 1972 1977
1982 1987 1992 1997 ...
##  $ lifeExp  : num [1:1704] 28.8 30.3 32 34 36.1 ...
##  $ pop      : int [1:1704] 8425333 9240934 10267083
11537966 13079460 14880372 12881816 13867957 16317921
22227415 ...
##  $ gdpPercap: num [1:1704] 779 821 853 836 740 ...
```

Data exploration (approach 2)

```
glimpse(gapminder) # similar to str() cleaner output
## Rows: 1,704
## Columns: 6
## $ country   <fct> "Afghanistan", "Afghanistan", "Afghanistan",
"Afghanistan", …
## $ continent <fct> Asia, Asia, Asia, Asia, Asia, Asia, Asia,
Asia, Asia, Asia, …
## $ year      <int> 1952, 1957, 1962, 1967, 1972, 1977, 1982,
1987, 1992, 1997, …
## $ lifeExp   <dbl> 28.801, 30.332, 31.997, 34.020, 36.088,
38.438, 39.854, 40.8…
## $ pop       <int> 8425333, 9240934, 10267083, 11537966,
13079460, 14880372, 12…
## $ gdpPercap <dbl> 779.4453, 820.8530, 853.1007, 836.1971,
739.9811, 786.1134, …
```

Data exploration (approach 3)

```
skimr::skim(gapminder) # like str()
+ summary() + more
```

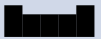
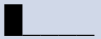
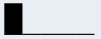
Name	gapminder
Number of rows	1704
Number of columns	6
_____	
Column type frequency:	
factor	2
numeric	4
_____	
Group variables	None

Data summary

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
country	0	1	FALSE	142	Afg: 12, Alb: 12, Alg: 12, Ang: 12
continent	0	1	FALSE	5	Afr: 624, Asi: 396, Eur: 360, Ame: 300

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
year	0	1	1979.50	17.27	1952.00	1965.75	1979.50	1993.25	2007.0	
lifeExp	0	1	59.47	12.92	23.60	48.20	60.71	70.85	82.6	
pop	0	1	29601212.32	106157896.74	60011.00	2793664.00	7023595.50	19585221.75	1318683096.0	
gdpPercap	0	1	7215.33	9857.45	241.17	1202.06	3531.85	9325.46	113523.1	

In-class activity: Group practice

Solve the following problems.

1. How many continents and countries are in the dataset?
2. How many years are observed in the dataset?
3. Identify grouping variables. For instance, if there's a school, a grouping variable is a class because students are nested in a class.

Visualizing (ggplot2)

The grammar of graphics

- the grammar of graphics
  - data
  - aesthetic attributes (color, shape, size)
  - geometric objects (points, lines, bars)
  - stats (summary stats)
  - scales (map values in the data space)
  - coord (data coordinates)
  - facet (facetting specifications)

No worries about new terms. We're going to learn them by actually plotting.

- Workflow:
  1. Tidy data (what data) : `ggplot(data = )`
  2. Mapping (what relationships) : `aes(x = , y= )`
  3. Geom (how) : `geom_()`
  4. Cooordinates and scales (how to see)
  5. Labels and guides (how to guide) : `labs()`, `guides()`
  6. Themes (how to theme)
  7. Save files

mapping and geom

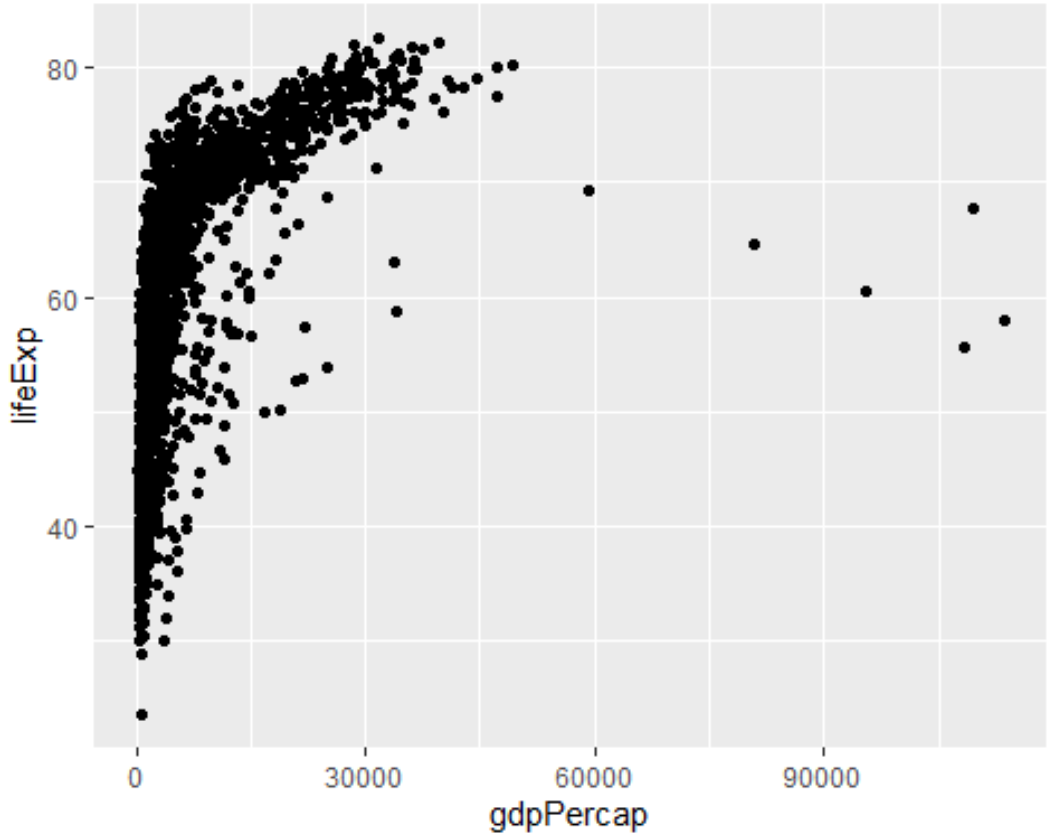
- `aes` (aesthetic mappings or aesthetics) tells which variables (x, y) in your data should be represented by which visual elements (color, shape, size) in the plot.
- `geom_` tells the type of plot you are going to use

Toy example

```
gapminder
## # A tibble: 1,704 × 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## # … with 1,694 more rows
```

Toy example: ggplot

```
p <- ggplot(
  data = gapminder,
  mapping = aes(x = gdpPercap, y =
lifeExp)
) # ggplot or R in general takes
positional arguments too. So, you
don't need to name data, mapping
each time you use ggplot2.

p + geom_point()
```



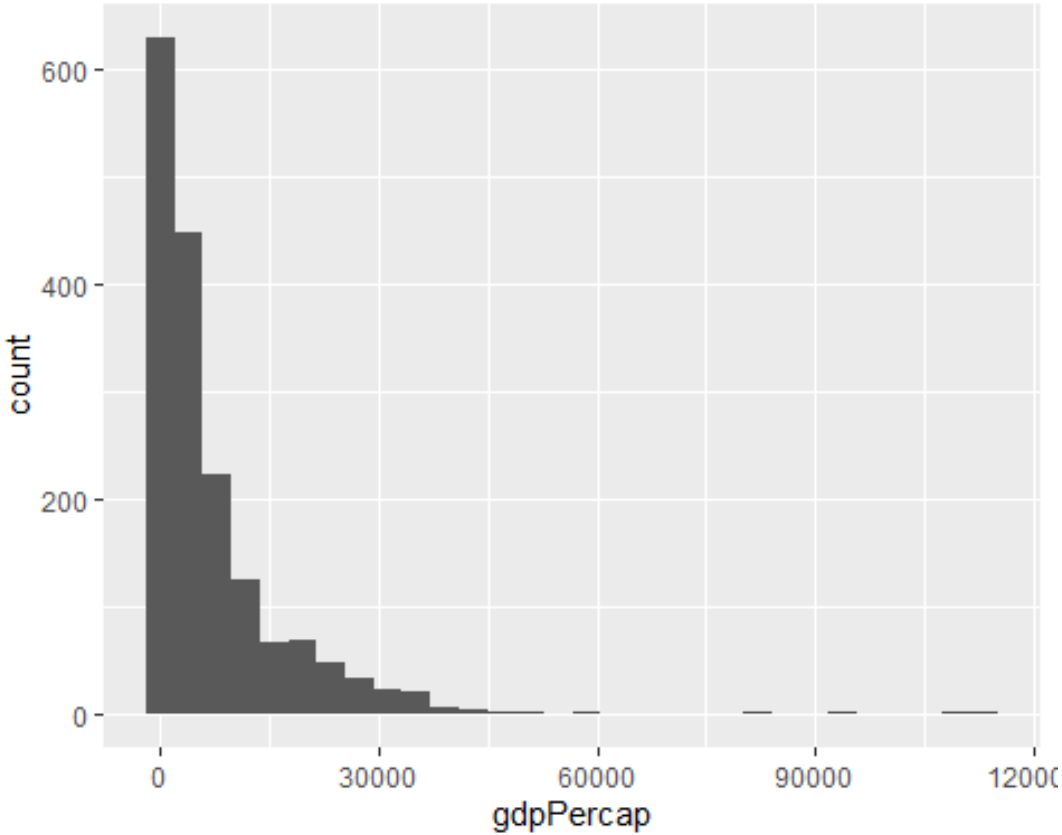
Univariate distribution

- Types of univariate plots (Nolan and Stoudt 2021: 72):
  - Quantitative: rug plot, histogram, density curve, box-and-whisker plot, violin plot, normal quantile plot
  - Qualitative: bar plot, dot chart, line plot, pie chart
- `geom_histogram()`: For the probability distribution of a continuous variable. Bins divide the entire range of values into a series of intervals (see the Wiki entry).
- `geom_density()`: Also for the probability distribution of a continuous variable. It calculates a kernel density estimate of the underlying distribution.

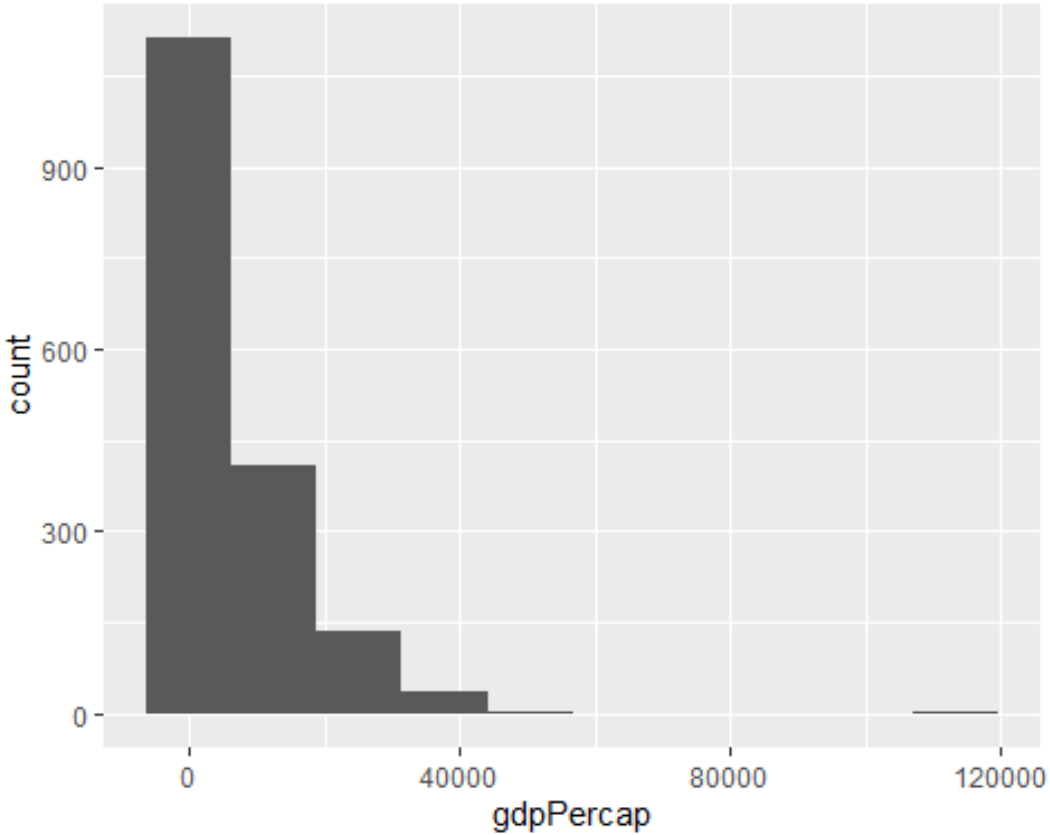
Histogram

```
gapminder %>%
  ggplot(aes(x = gdpPercap)) +
  geom_histogram() # stat_bin argument picks up 30 bins (or "bucket") by default. = statistical transformation
```

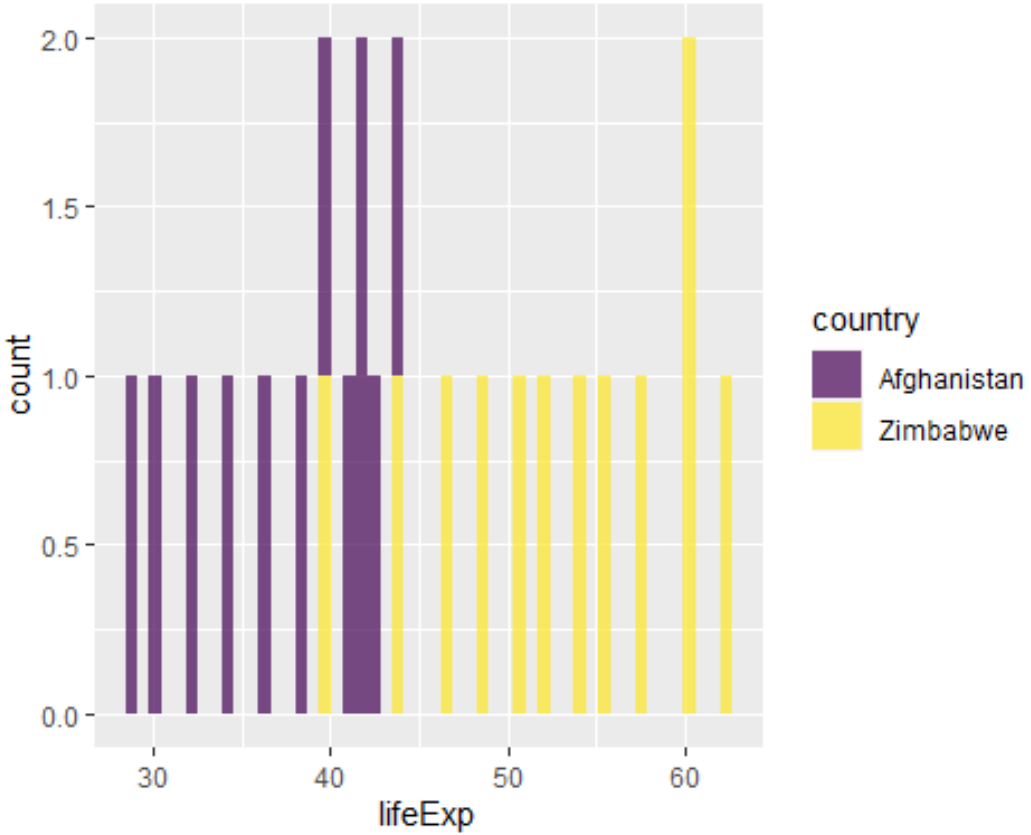
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```r
gapminder %>%
  ggplot(aes(x = gdpPercap)) +
  geom_histogram(bins = 10) # only
10 bins.
```

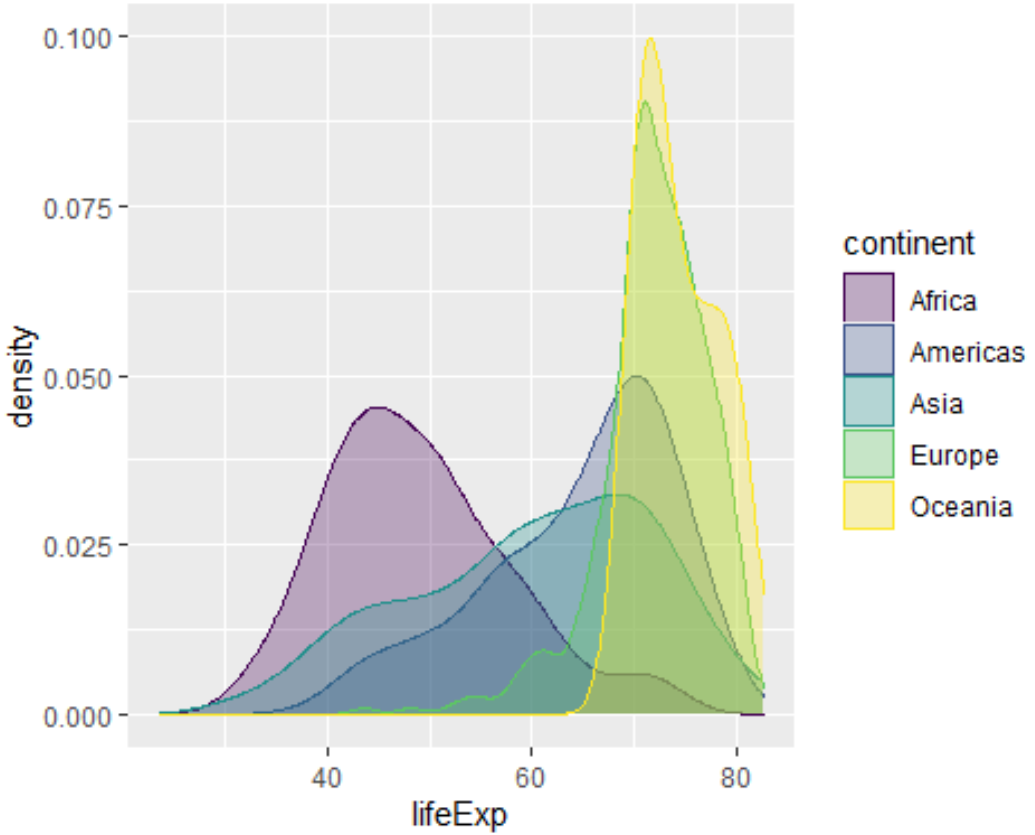


```
ggplot(
  data = subset(gapminder, country
%in% c("Afghanistan", "Zimbabwe")),
  mapping = aes(x = lifeExp, fill =
country)
) +
  geom_histogram(bins = 50, alpha =
0.7) +
  scale_fill_viridis_d()
```



Density

```
gapminder %>%
  ggplot(aes(x = lifeExp, fill =
continent, color = continent)) +
  geom_density(alpha = 0.3) +
  scale_color_viridis_d() +
  scale_fill_viridis_d()
```

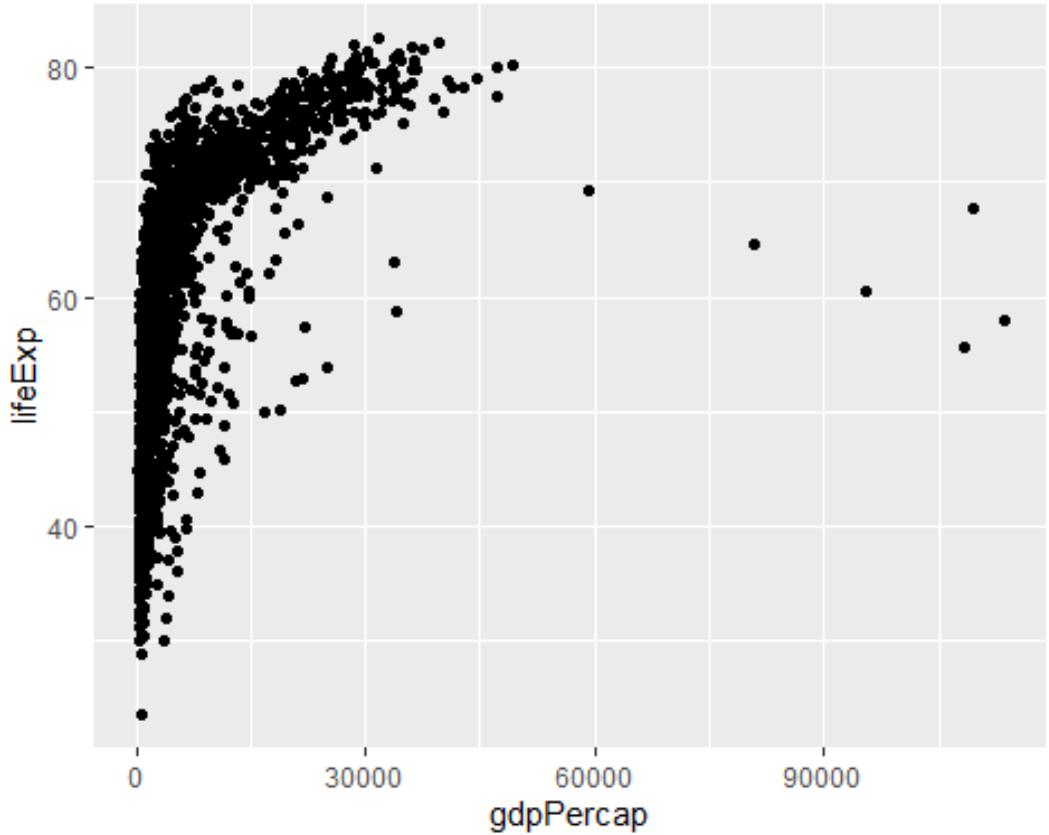


Bivariate distributions

Types of bivariate plots (Nolan and Stoudt 2021: 72):

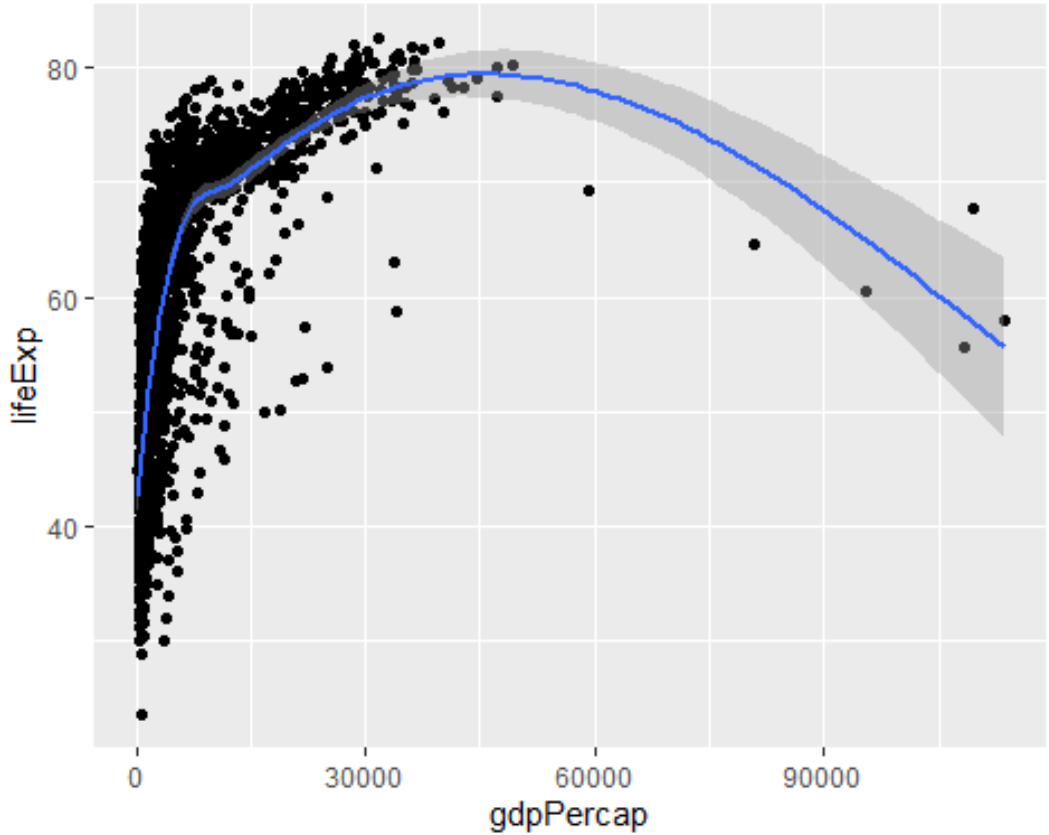
- Both quantitative: scatter plot, smooth curve, contour plot, heat map
- Both qualitative: side-by-side bar plots, mosaic plot, overlaid lines
- Quantitative/Qualitative: overlaid density curves, side-by-side box-and-whisker plots, overlaid smooth curves, quantile-quantile plot

```
p <- ggplot(data = gapminder,
            aes(x = gdpPercap,
                y = lifeExp))

# Scatter plot
p + geom_point()
```

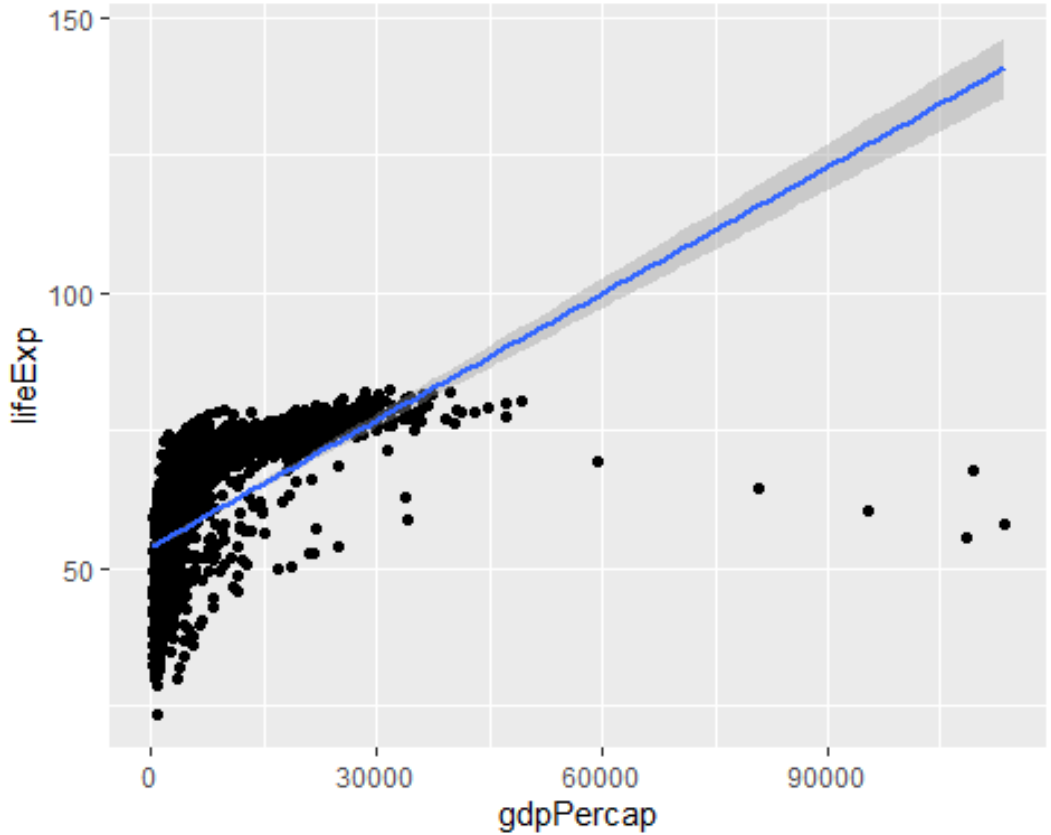


Smoothing helps to clarify the trend(s).

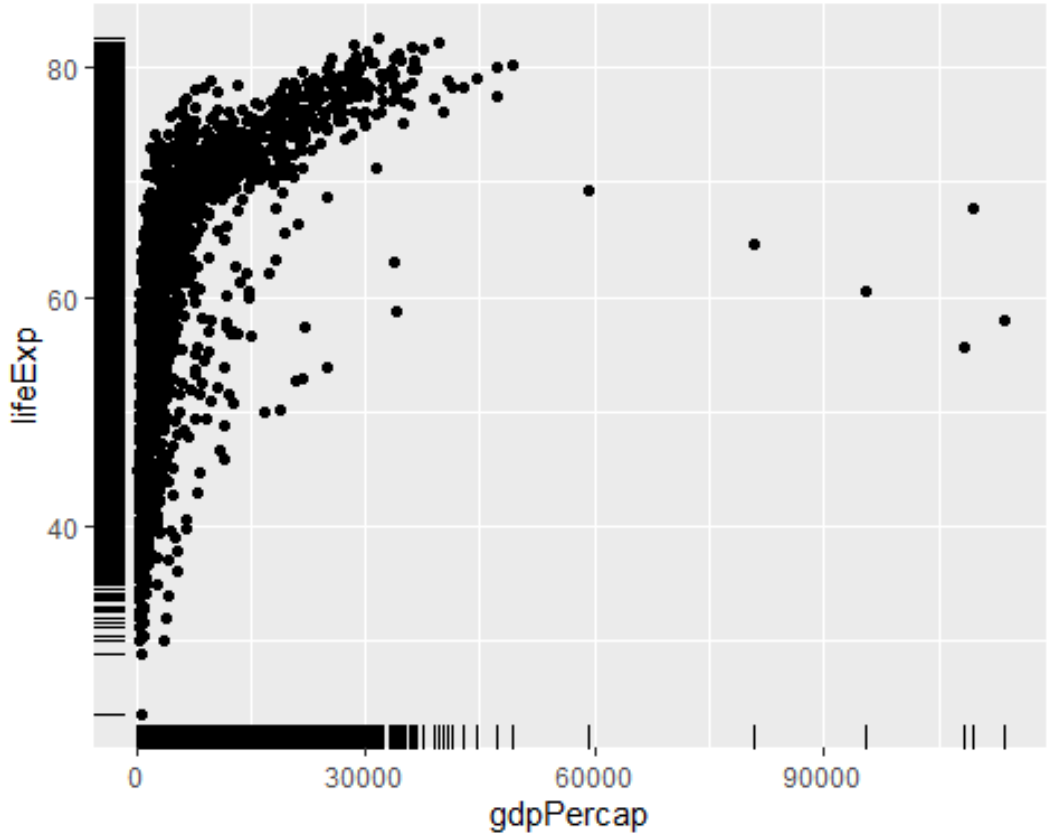
```
# Adding a smoothed line

p + geom_point() +
  geom_smooth()
## `geom_smooth()` using method =
'gam' and formula 'y ~ s(x, bs =
"cs")'
```



```
p + geom_point() +
  geom_smooth(method = "lm")
## `geom_smooth()` using formula 'y
~ x'
```



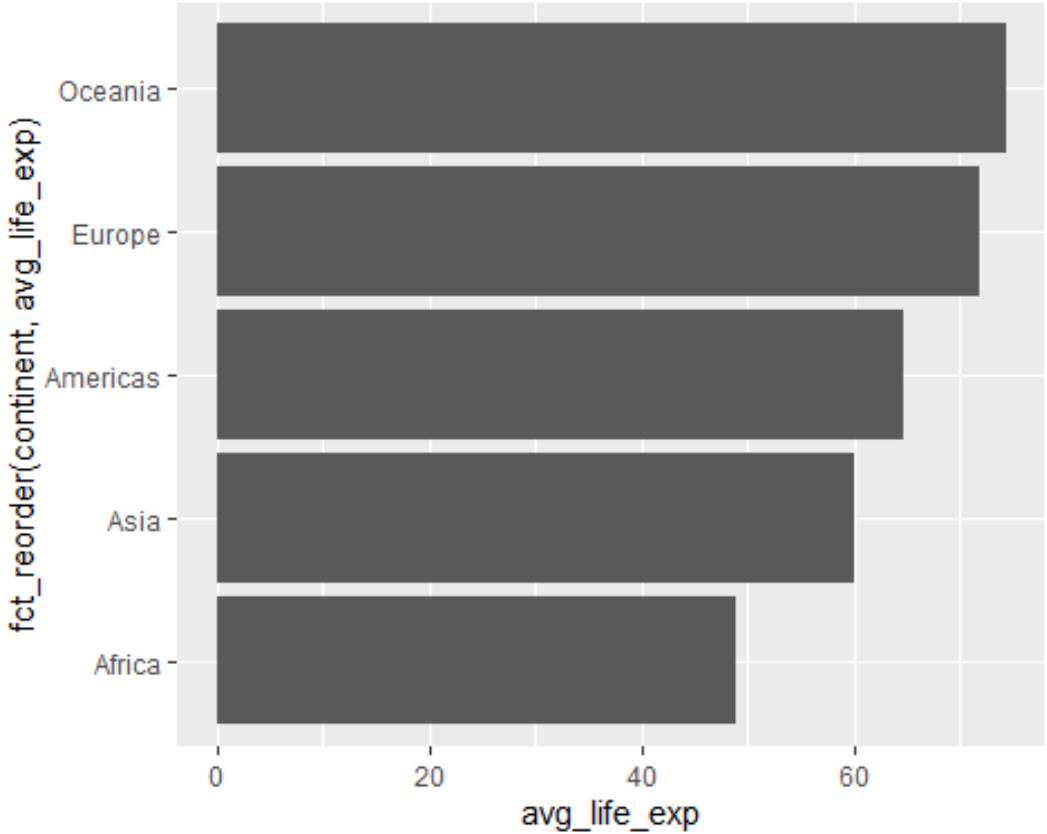
```r
# rug plot
p + geom_point() +
  geom_rug()
```



If observations are too few: consider using scatter plots without a smoothed line.

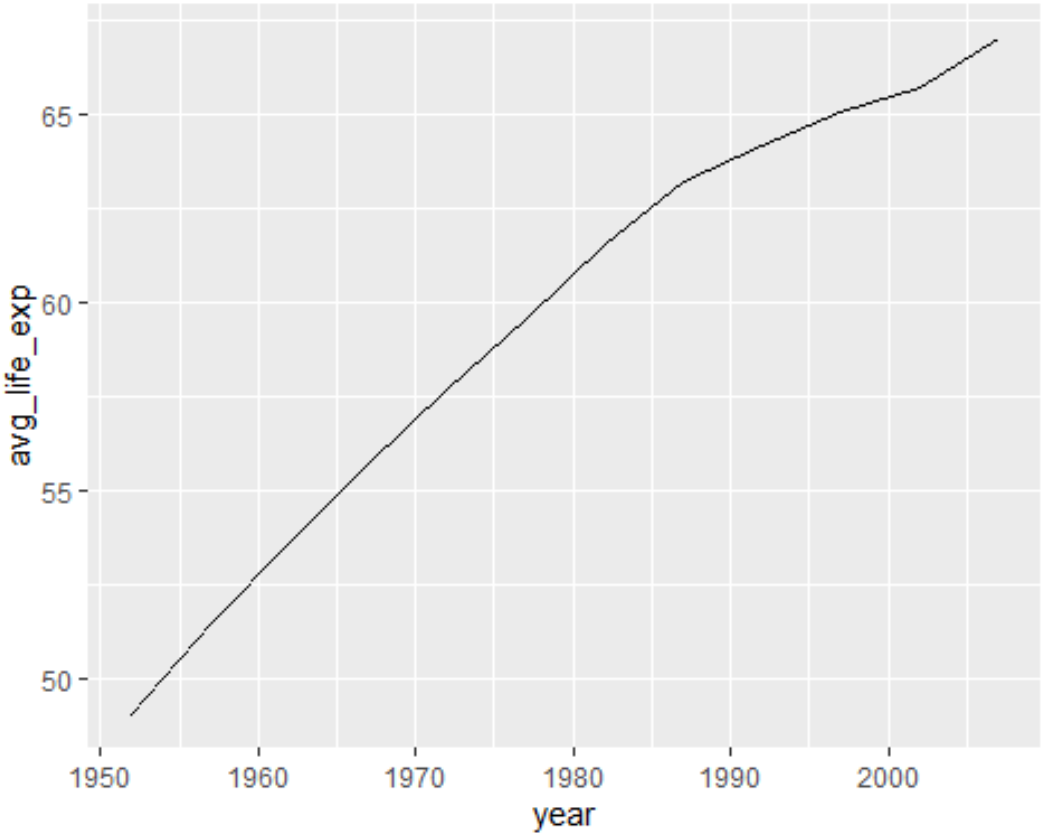
Discrete comparison

```
# Bar plot
gapminder %>%
  group_by(continent) %>%
  summarize(avg_life_exp =
mean(lifeExp)) %>%
  ggplot(aes(x =
fct_reorder(continent,
avg_life_exp), y = avg_life_exp)) +
  geom_col() +
  coord_flip()
```

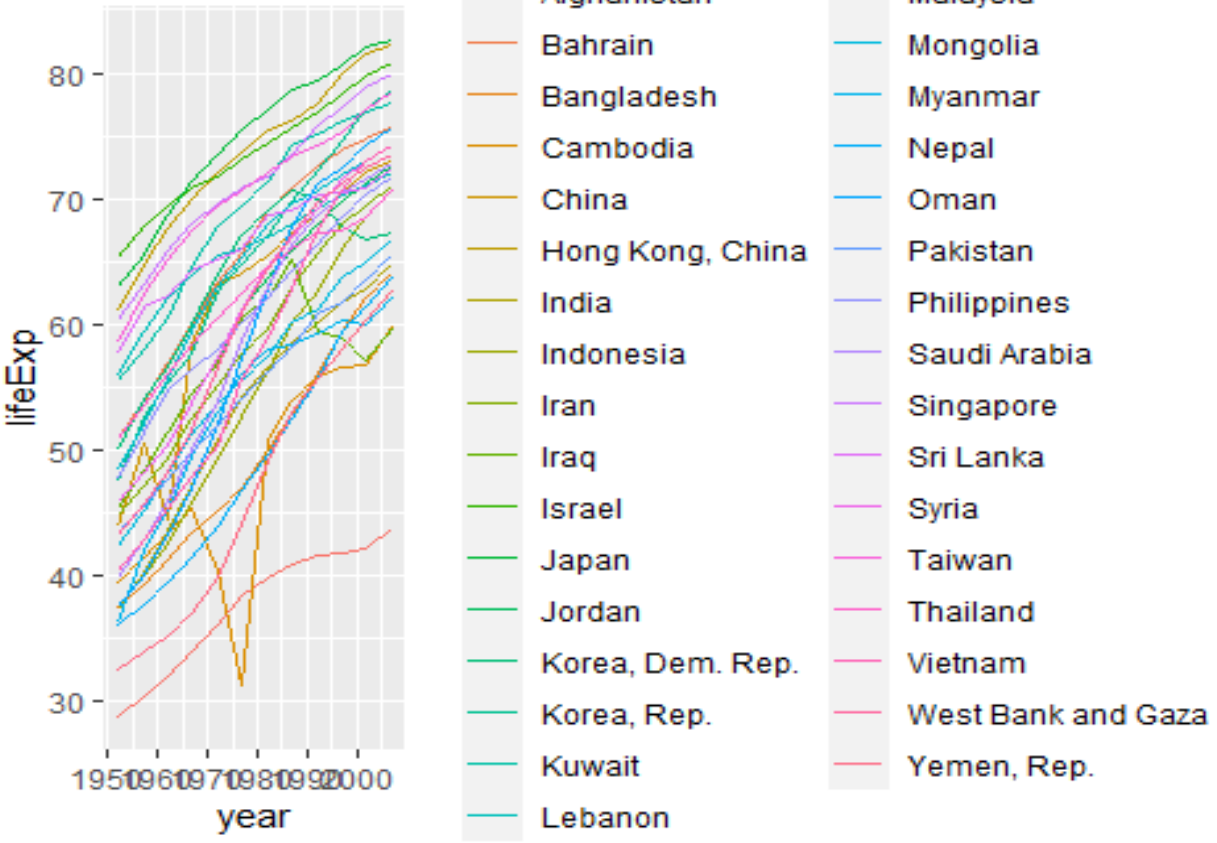


Temporal trends

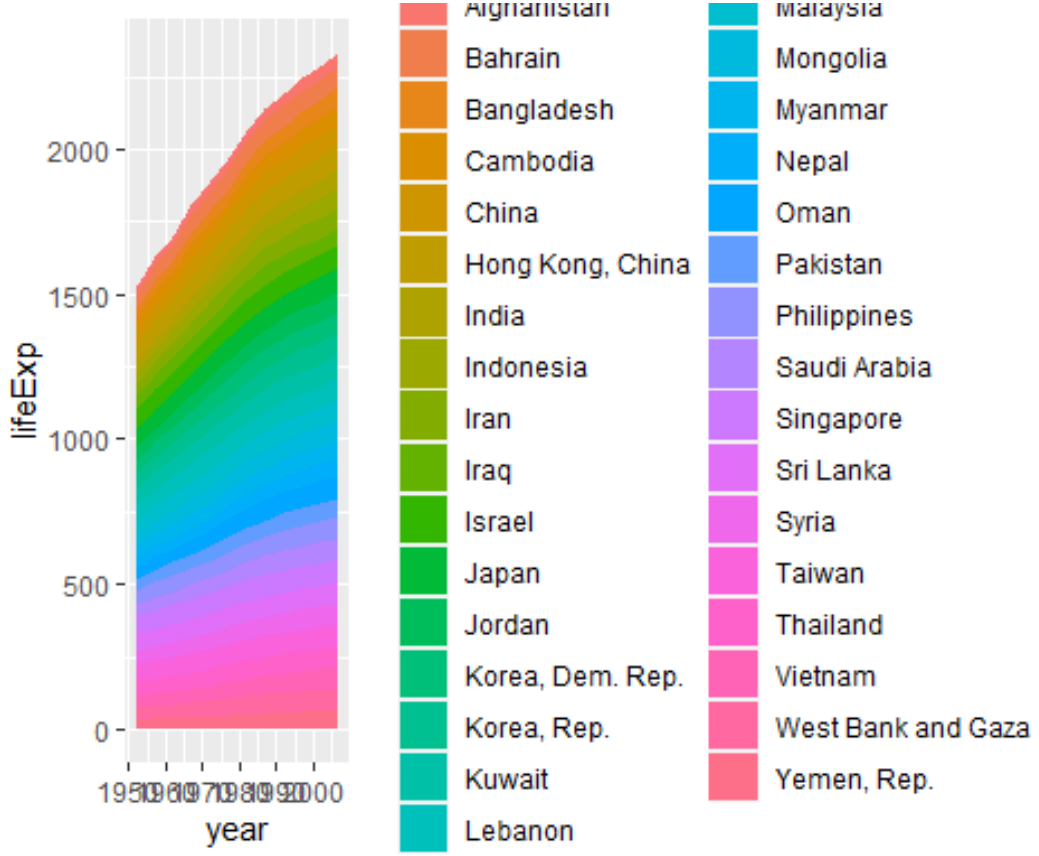
```
# Line plot using a summary
gapminder %>%
  group_by(year) %>%
  summarize(avg_life_exp =
mean(lifeExp)) %>%
  ggplot(aes(x = year, y =
avg_life_exp)) +
  geom_line()
```



```r
# individual comparison
gapminder %>%
  filter(continent == "Asia
  ggplot(aes(x = year, y =
                col = country)
  geom_line()
```



```r
# overall trend
gapminder %>%
  filter(continent == "Asia") %>%
  ggplot(aes(x = year, y = lifeExp,
             fill = country)) +
  geom_area()
```

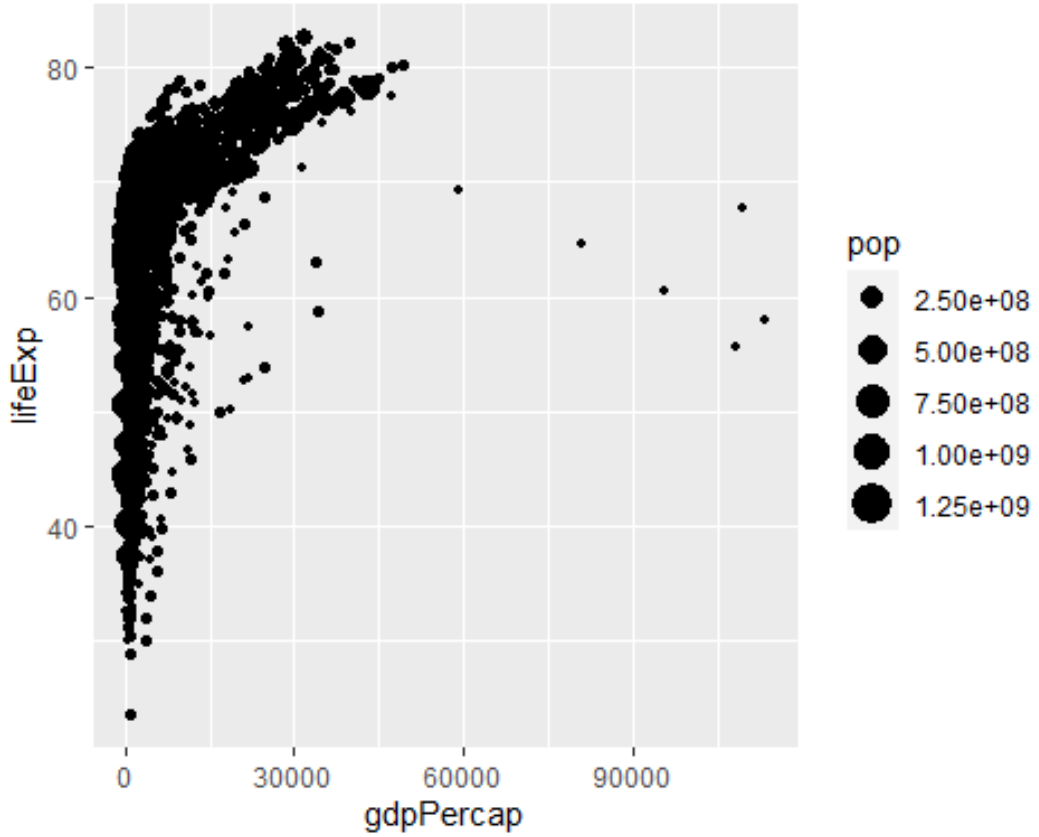


Advanced aes (size, color)

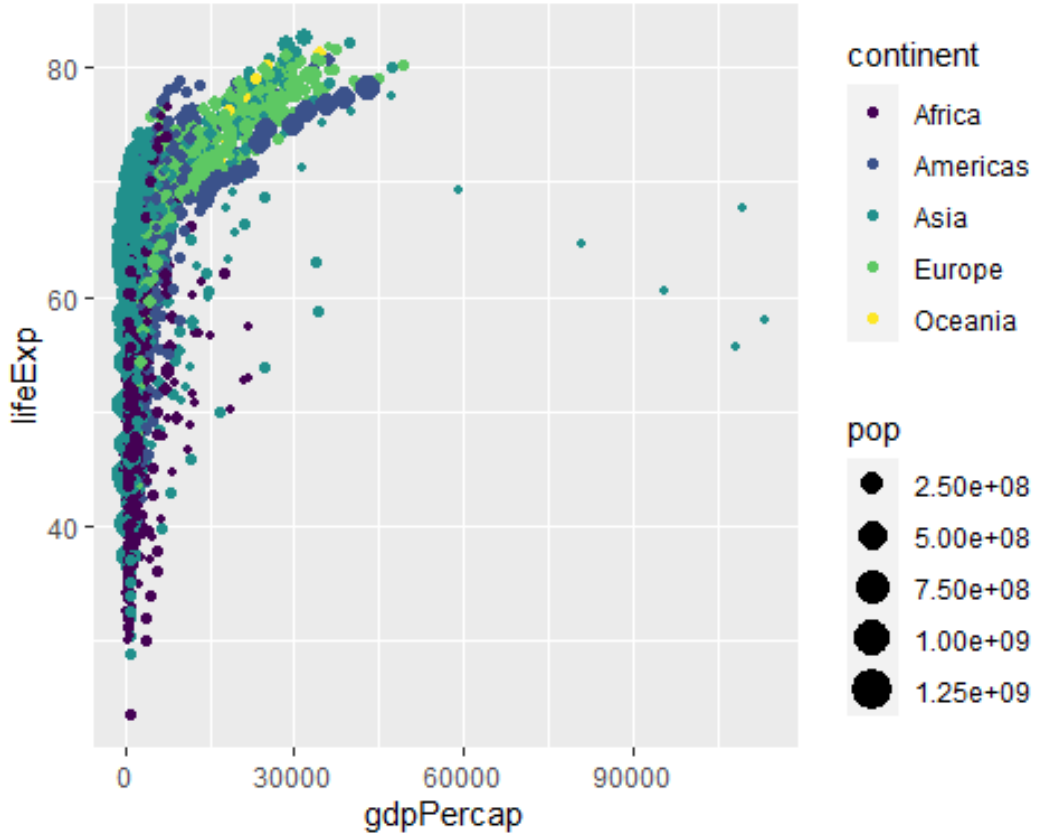
There's also `fill` argument (mostly used in `geom_bar()`). Color `aes` affects the appearance of lines and points, fill is for the filled areas of bars, polygons, and in some cases, the interior of a smoother's standard error ribbon.

The property size/color/fill represents…

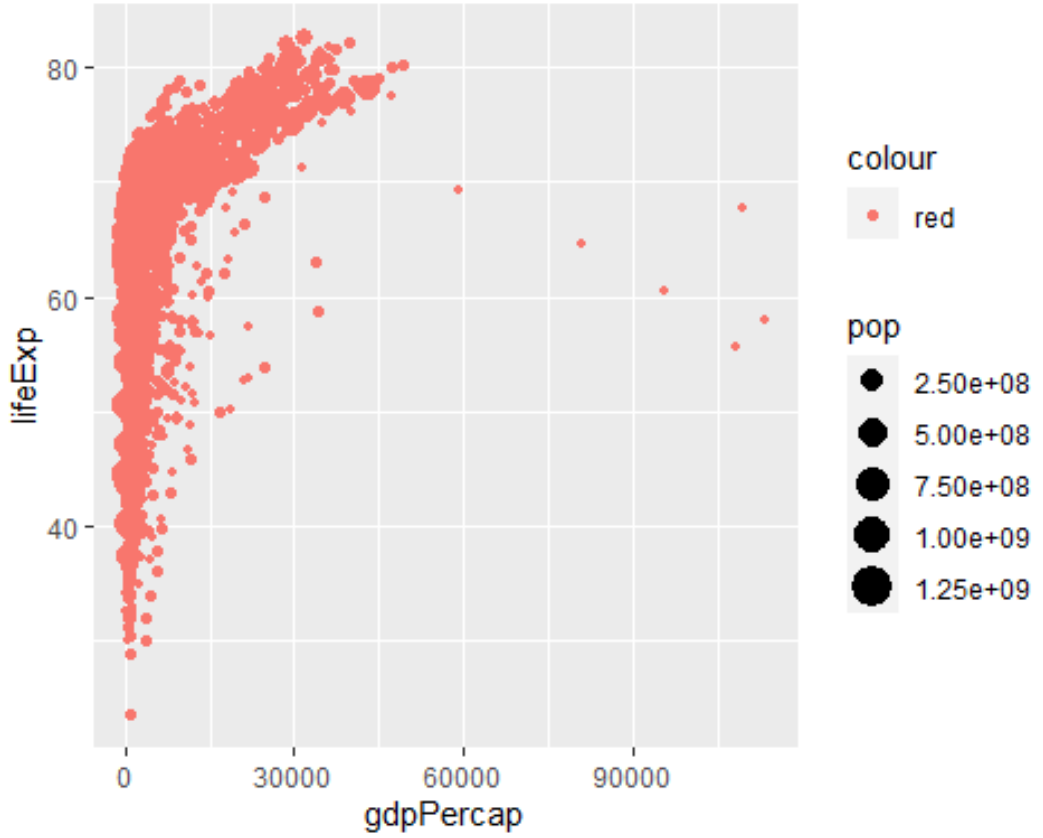
```
ggplot(
  data = gapminder,
  mapping = aes(
    x = gdpPercap, y = lifeExp,
    size = pop
  )
) +
  geom_point()
```



```r
ggplot(
  data = gapminder,
  mapping = aes(
    x = gdpPercap, y = lifeExp,
    size = pop,
    color = continent
  )
) +
  geom_point() +
  scale_color_viridis_d()
```



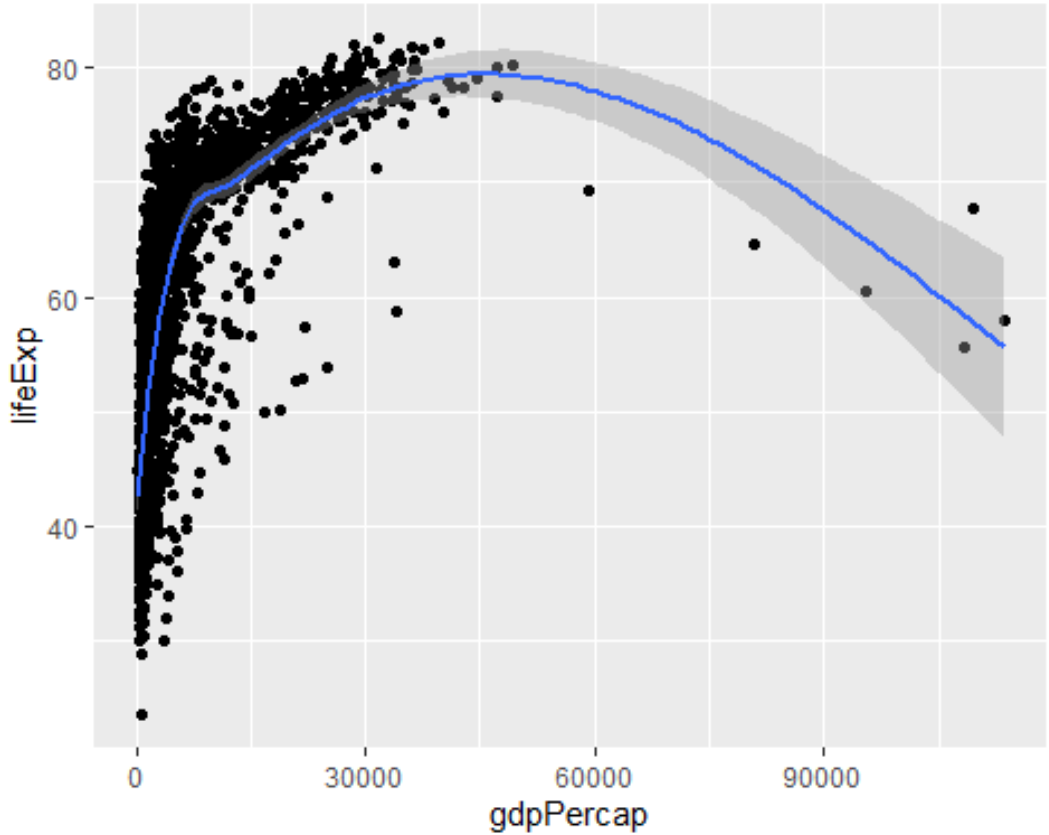
```r
# try red instead of "red"
ggplot(
  data = gapminder,
  mapping = aes(
    x = gdpPercap, y = lifeExp,
    size = pop,
    color = "red"
  )
) +
  geom_point()
```



Aesthetics also can be mapped per Geom.

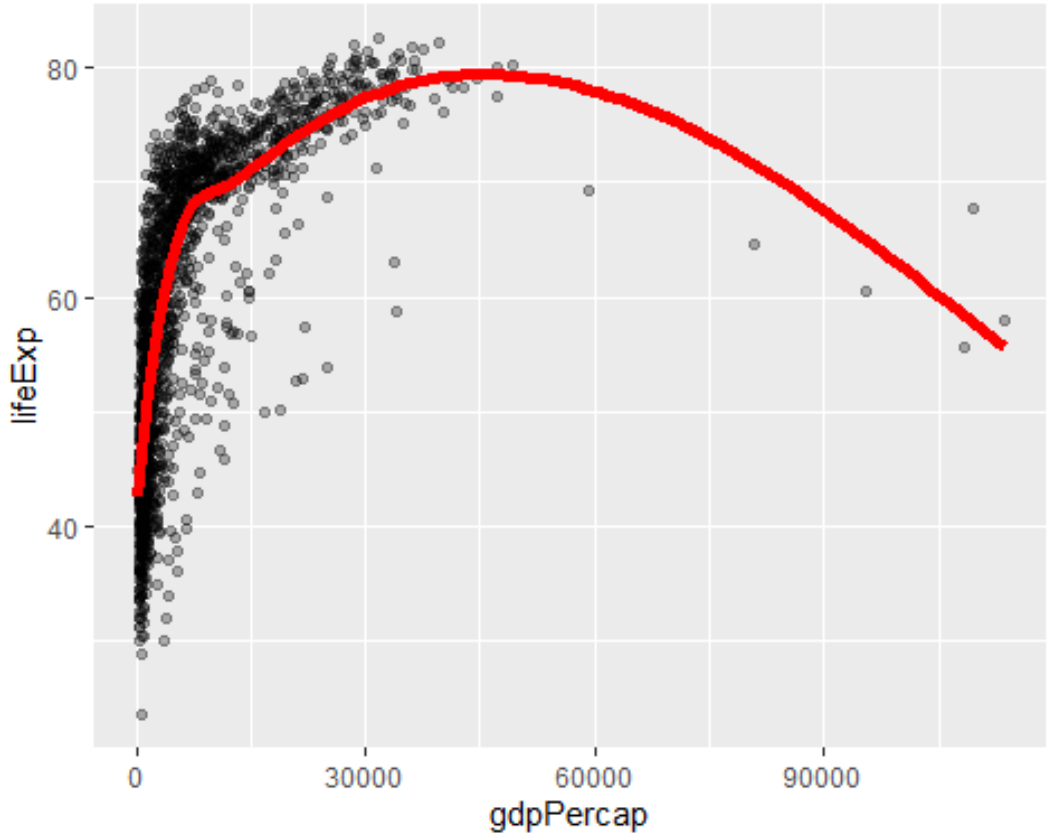
```
p + geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method =
'gam' and formula 'y ~ s(x, bs =
"cs")'
```

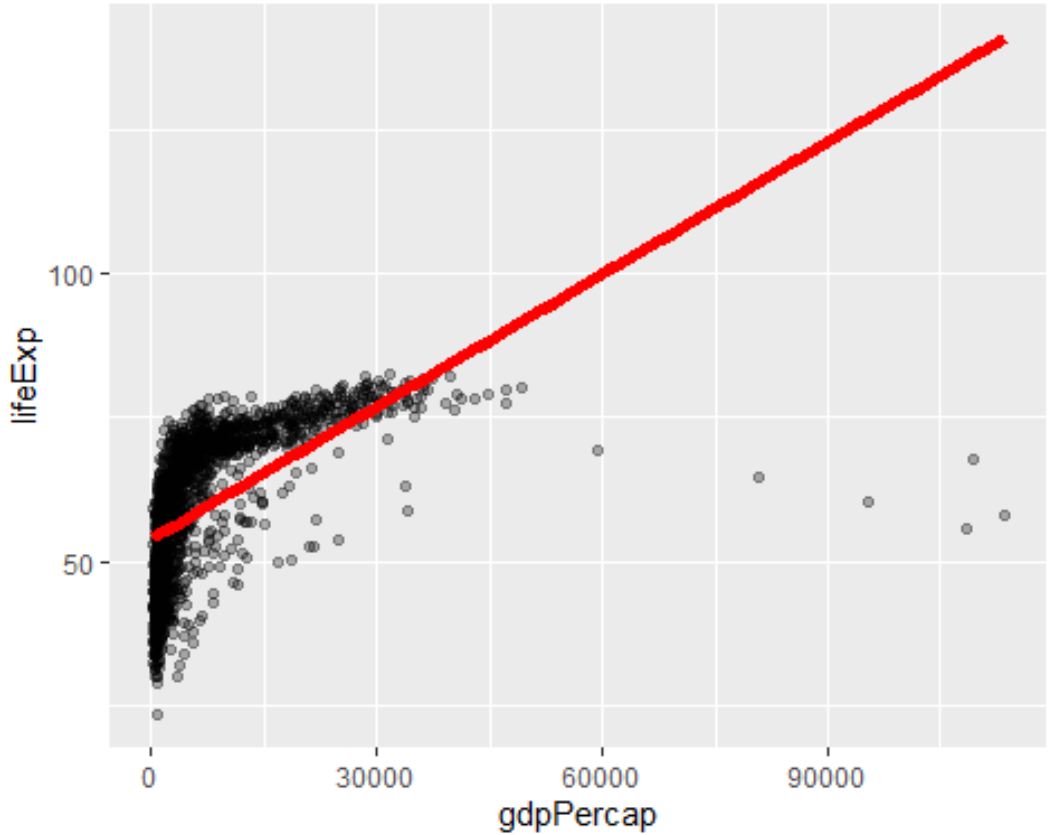


```
p + geom_point(alpha = 0.3) + # alpha controls transparency
  geom_smooth(color = "red", se = FALSE, size = 2)
```

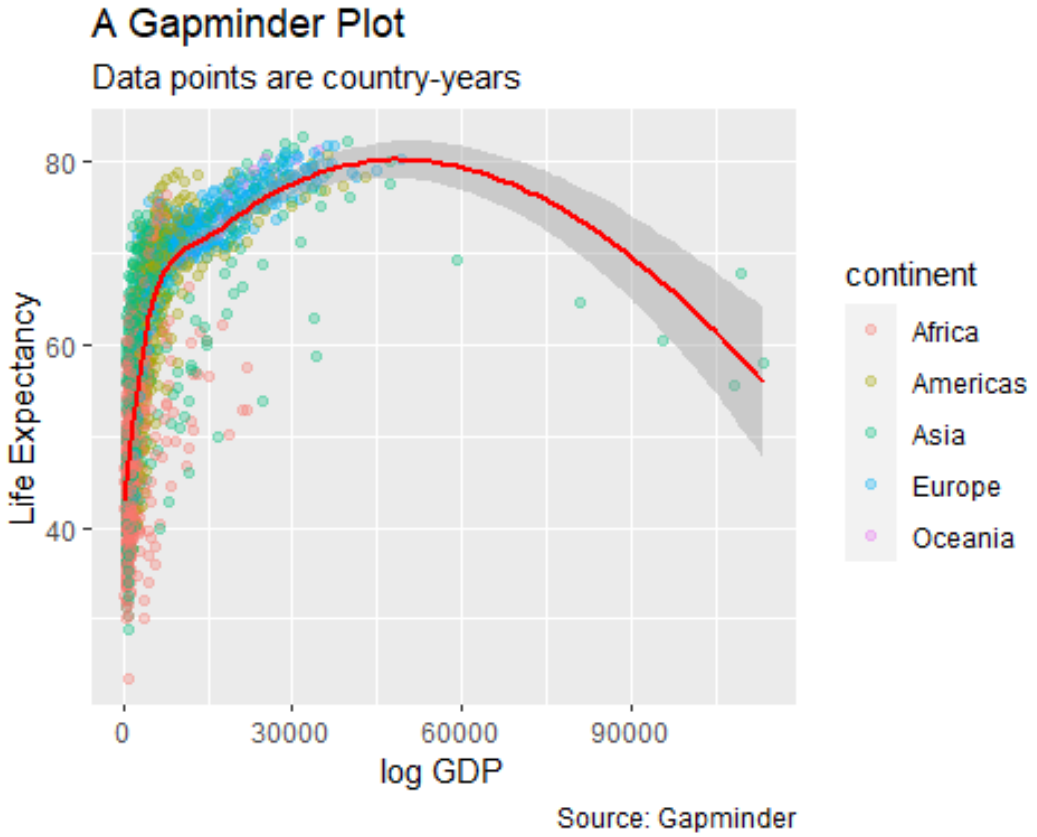
```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



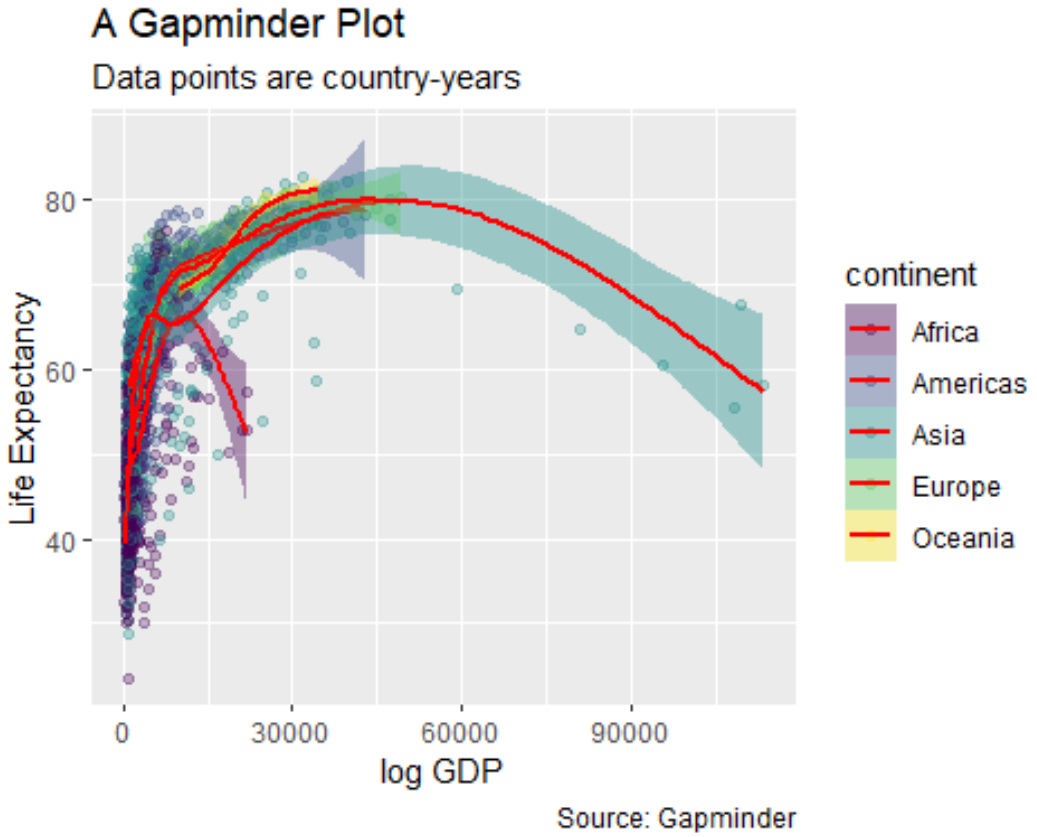
```
p + geom_point(alpha = 0.3) + # alpha controls transparency
  geom_smooth(color = "red", se = FALSE, size = 2, method = "lm")
## `geom_smooth()` using formula 'y ~ x'
```



```
ggplot(
  data = gapminder,
  mapping = aes(
    x = gdpPercap, y = lifeExp,
    color = continent
  )
) +
  geom_point(alpha = 0.3) +
  geom_smooth(method = "loess",
color = "red") +
  labs(
    x = "log GDP",
    y = "Life Expectancy",
    title = "A Gapminder Plot",
    subtitle = "Data points are
country-years",
    caption = "Source: Gapminder"
  )
## `geom_smooth()` using formula 'y
~ x'
```

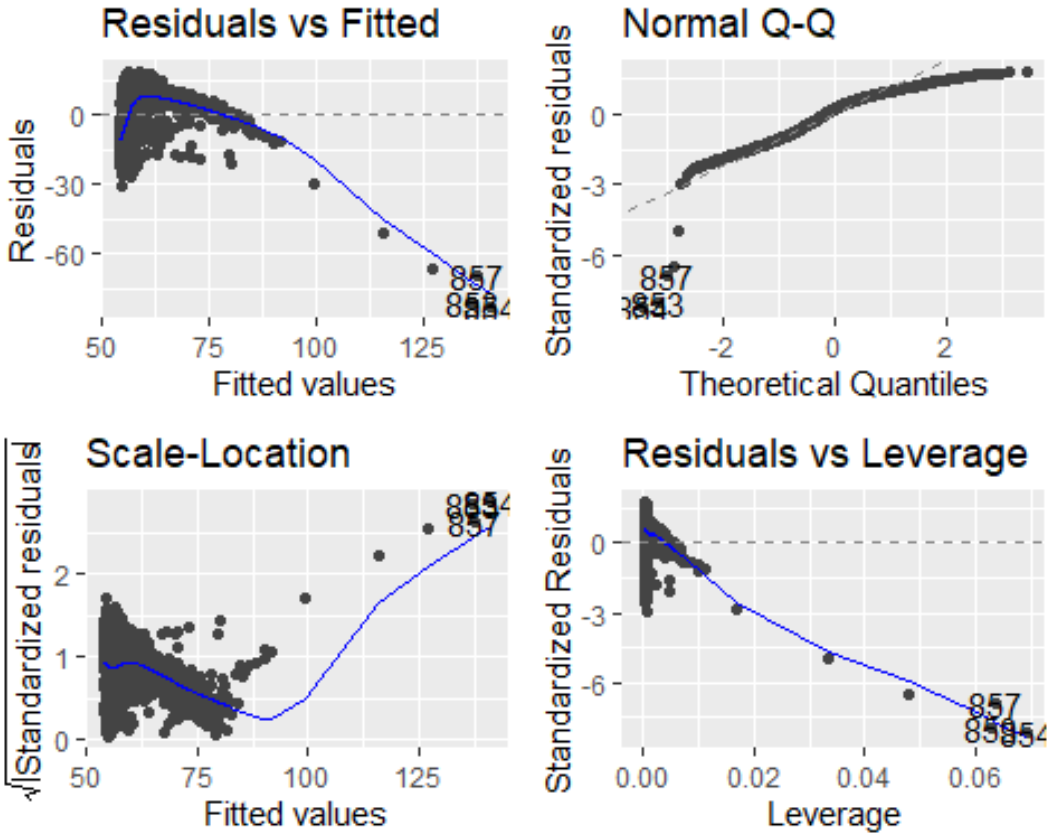


```
ggplot(
  data = gapminder,
  mapping = aes(
    x = gdpPercap, y = lifeExp,
    color = continent,
    fill = continent
  )
) +
  geom_point(alpha = 0.3) +
  geom_smooth(method = "loess", color
= "red") +
  labs(
    x = "log GDP",
    y = "Life Expectancy",
    title = "A Gapminder Plot",
    subtitle = "Data points are
country-years",
    caption = "Source: Gapminder"
  ) +
  scale_color_viridis_d() +
  scale_fill_viridis_d()
## `geom_smooth()` using formula 'y ~
x'
```

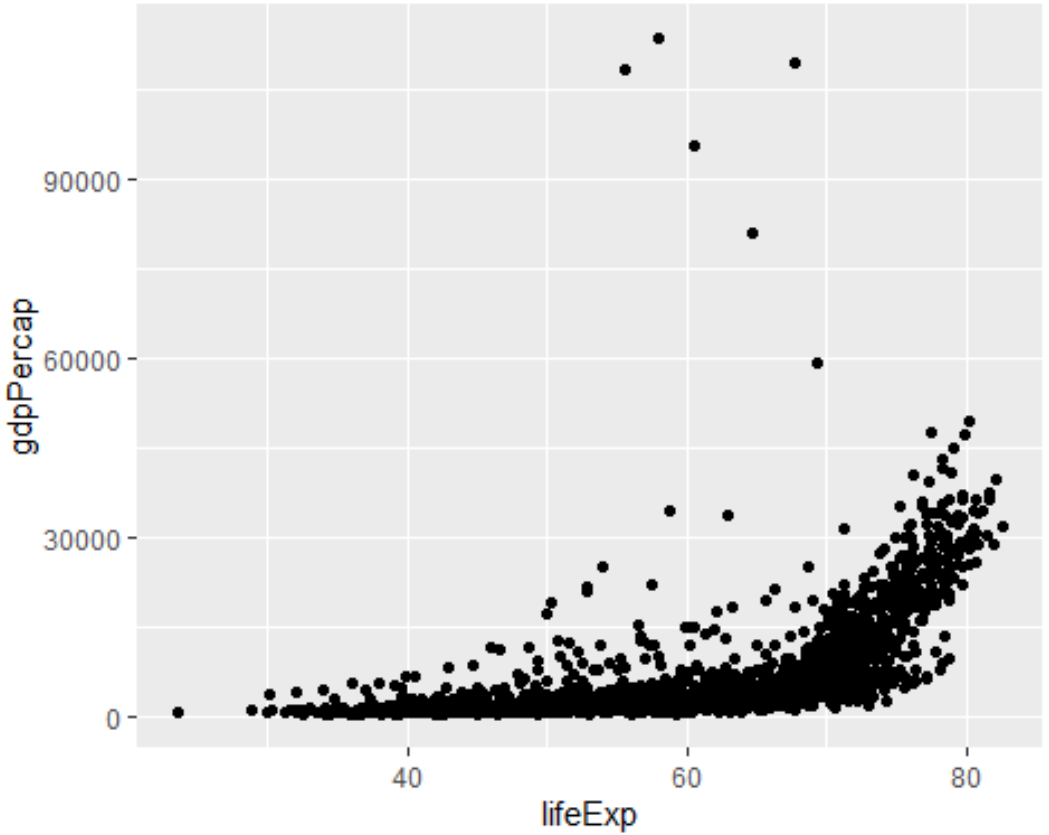


Coordinates and scales

```
# checking the data distribution is
important to evaluate a model

lm.out <- lm(lifeExp ~ gdpPercap,
data = gapminder)

autoplot(lm.out)
```

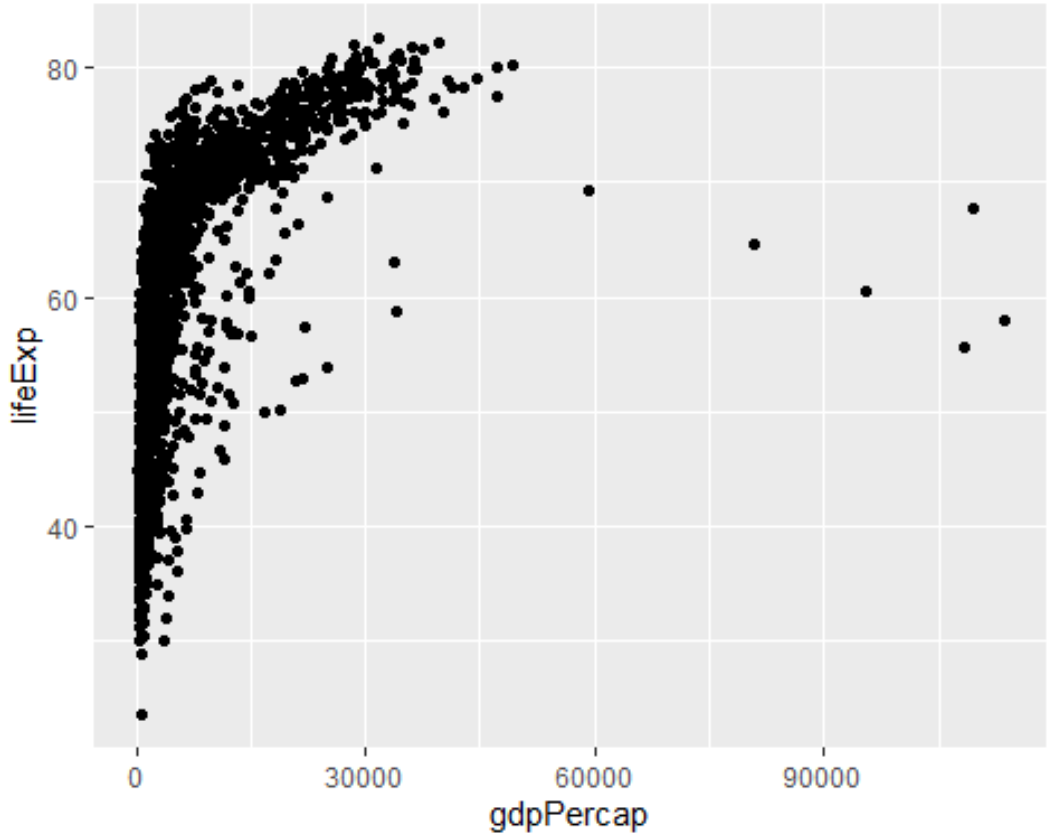


```
p + geom_point() +
  coord_flip() # coord_type
```



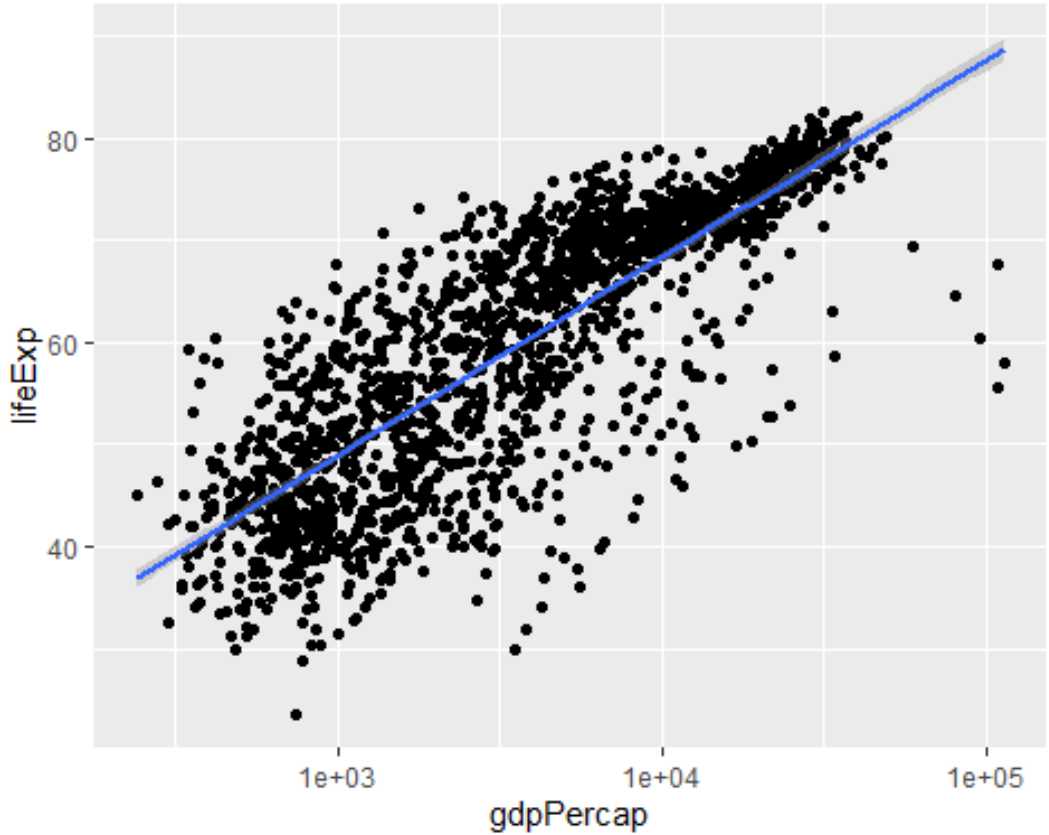
The data is heavily bunched up against the left side.

```
p + geom_point() # without scaling
```



```
p + geom_point() +
  scale_x_log10() # scales the axis
of a plot to a log 10 basis
```

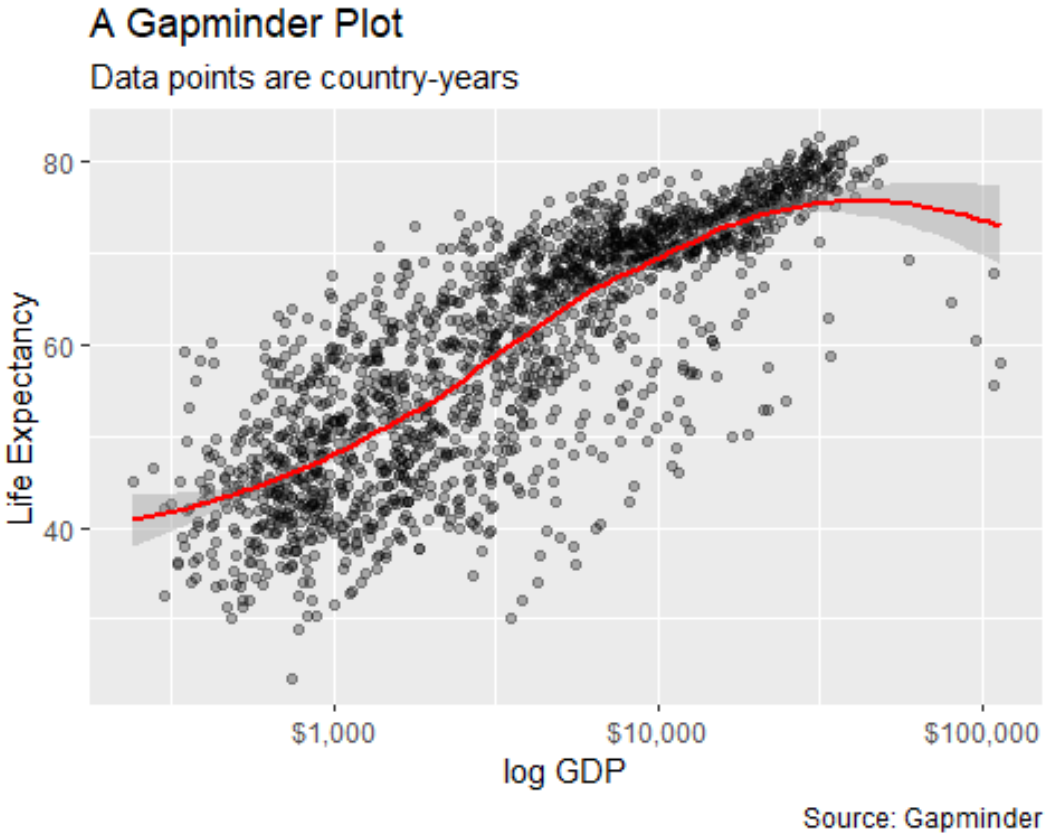
```
p + geom_point() +
  geom_smooth(method = "lm") +
  scale_x_log10()
## `geom_smooth()` using formula 'y
~ x'
```



Labels and guides

`scales` package has some useful premade formatting functions. You can either load scales or just grab the function you need from the library using `scales::`

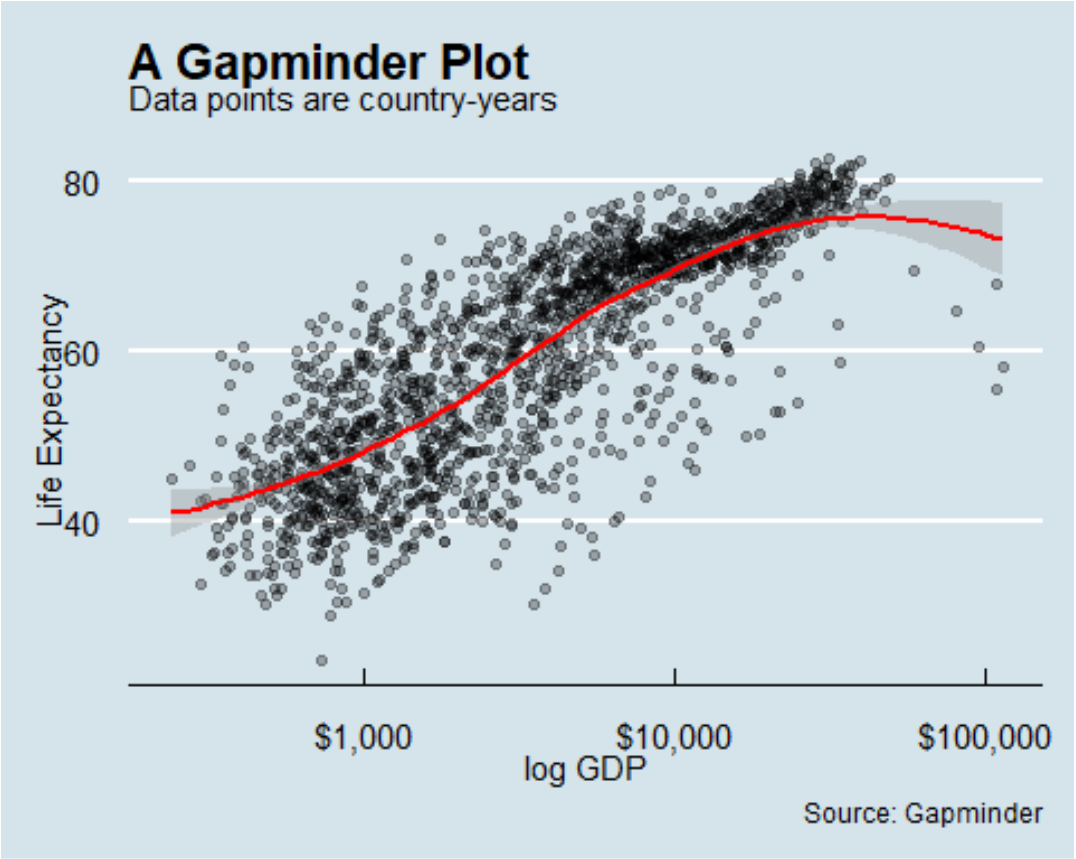
```
p + geom_point(alpha = 0.3) +
  geom_smooth(method = "loess",
color = "red") +
  scale_x_log10(labels =
scales::dollar) +
  labs(
    x = "log GDP",
    y = "Life Expectancy",
    title = "A Gapminder Plot",
    subtitle = "Data points are
country-years",
    caption = "Source: Gapminder"
  )
## `geom_smooth()` using formula 'y
~ x'
```



Themes

```
p + geom_point(alpha = 0.3) +
  geom_smooth(method = "loess",
color = "red") +
  scale_x_log10(labels =
scales::dollar) +
  labs(
    x = "log GDP",
    y = "Life Expectancy",
    title = "A Gapminder Plot",
    subtitle = "Data points are
country-years",
    caption = "Source: Gapminder"
  ) +
  theme_economist()
```

```
## `geom_smooth()` using formula 'y
~ x'
```



ggsave

```
figure_example <- p + geom_point(alpha = 0.3) +
  geom_smooth(method = "gam", color = "red") +
  scale_x_log10(labels = scales::dollar) +
  labs(
    x = "log GDP",
    y = "Life Expectancy",
    title = "A Gapminder Plot",
    subtitle = "Data points are country-years",
    caption = "Source: Gapminder"
  ) +
  theme_economist()
```

```
dir.create(here("outputs"))
## Warning in dir.create(here("outputs")): 'C:\Users\Vera\OneDrive -
## kdis.ac.kr\Woosong_2022\Work\2022_fall\DAfM\outputs' already exists
ggsave(here("outputs", "figure_example.png"))
## Saving 5 x 4 in image
## `geom_smooth()` using formula 'y ~ s(x, bs = "cs")'
```