Class05: Data Vis with ggplot

Melissa (PID: A16511023)

Graphics system in R

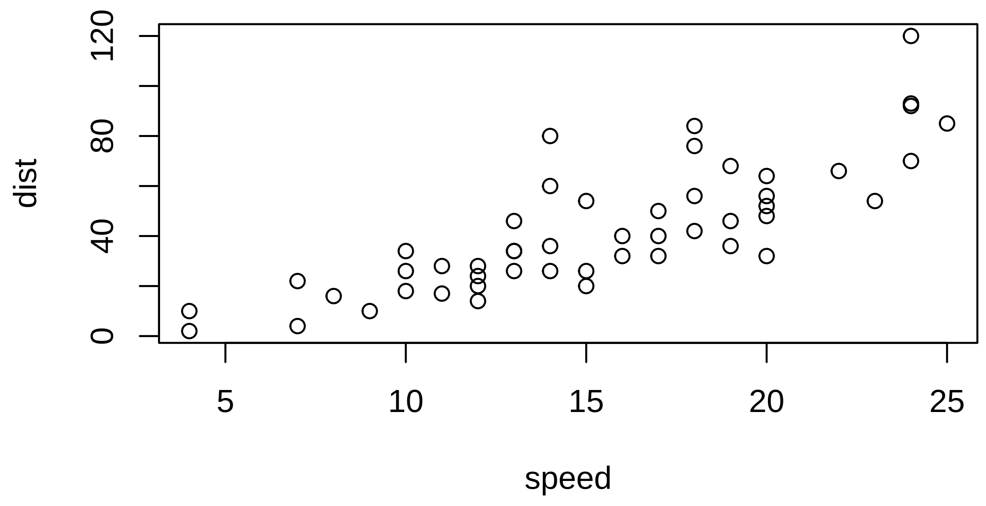
There are many graphics systems in R for making plots and figures.

We have already played a little with **"base R"** graphics and the `plot()` function. #back-tick is for code

Today we will start learning about a popular graphics package called `ggplot2()`.

This is an add on package - i.e. we need to install it. I install it (like I install any package) with the `install.package()` function.

```
plot(cars)
```

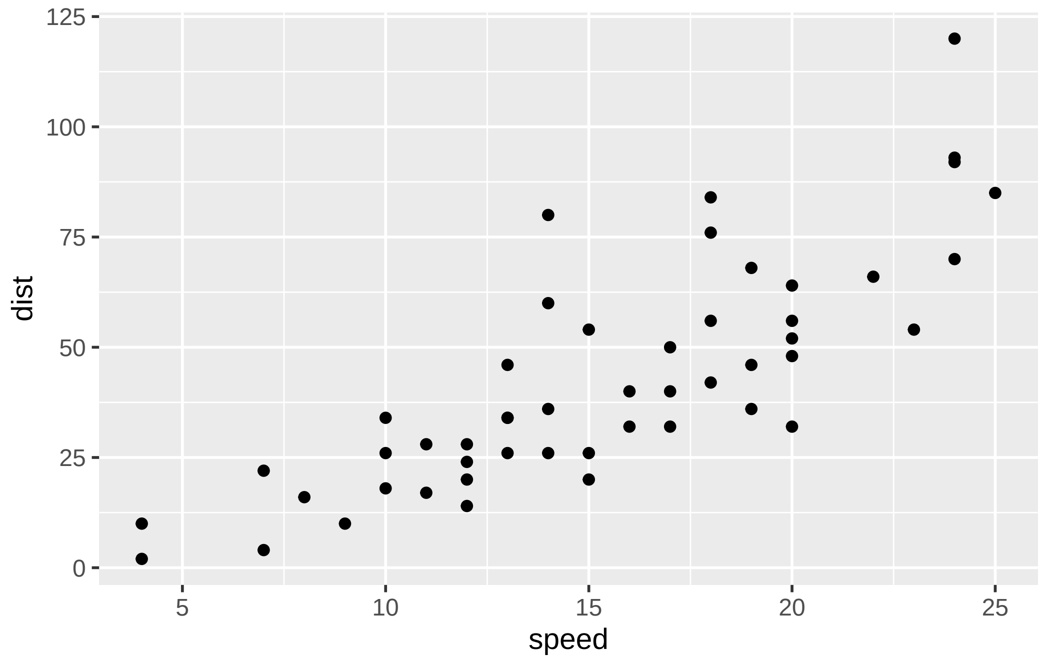


Before I can use the functions from a package I have to load up the package from my "library". We use the `library(ggplot2)` command to load it up.

```r
library(ggplot2)
ggplot(cars)
```

Every ggplot is made up of at least 3 things: - data (the numbers etc. that will go into your plot) - aes (how the columns of data map to the plot aesthetics) - geoms (how the plot actually looks, points, bars, lines, etc.)

```r
ggplot(cars) +
  aes(x=speed, y=dist) +
  geom_point()
```

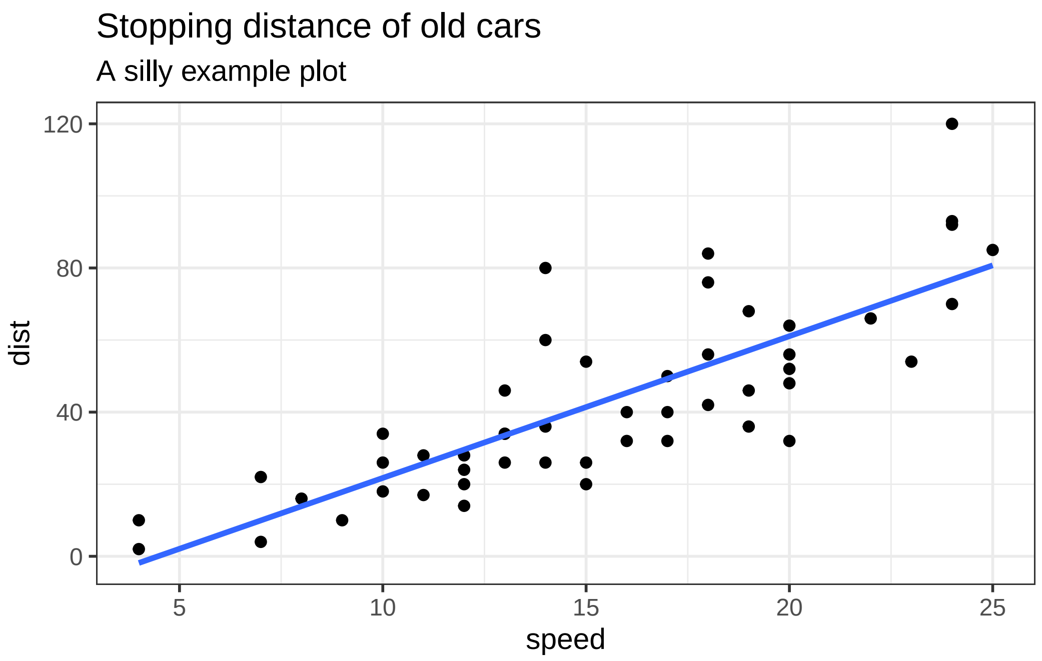


For simple plots ggplot is more verbose - it takes more code - than base R plot.

Add some more layers to our ggplot:

```
ggplot(cars) +
  aes(x=speed, y=dist) +
  geom_point() +
  geom_smooth(method=lm, se=FALSE) +
  labs(title="Stopping distance of old cars", subtitle="A silly example plot") +
  theme_bw()
```

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



#Section 6 of Lab

```
url <- "https://bioboot.github.io/bimm143_S20/class-material/up_down_expression.txt"
genes <- read.delim(url)
head(genes)
```

```
        Gene Condition1 Condition2      State
1      A4GNT -3.6808610 -3.4401355 unchanging
2       AAAS  4.5479580  4.3864126 unchanging
3      AASDH  3.7190695  3.4787276 unchanging
4       AATF  5.0784720  5.0151916 unchanging
5       AATK  0.4711421  0.5598642 unchanging
6 AB015752.4 -3.6808610 -3.5921390 unchanging
```

Q. Use the nrow() function to find out how many genes are in this dataset. What is your answer?

```
nrow(genes)
```

```
[1] 5196
```

Q. Use the colnames() function and the ncol() function on the genes data frame to find out what the column names are (we will need these later) and how many columns there are. How many columns did you find?

```
colnames(genes)
```

```
[1] "Gene"       "Condition1" "Condition2" "State"
```

```
ncol(genes)
```

```
[1] 4
```

Q. Use the table() function on the State column of this data.frame to find out how many 'up' regulated genes there are. What is your answer?

```
table(genes$State)
```

```

      down unchanging         up 
        72       4997        127 
```

Q. Using your values above and 2 significant figures. What fraction of total genes is up-regulated in this dataset?

```
round(table(genes$State) / nrow(genes) *100, 2)
```

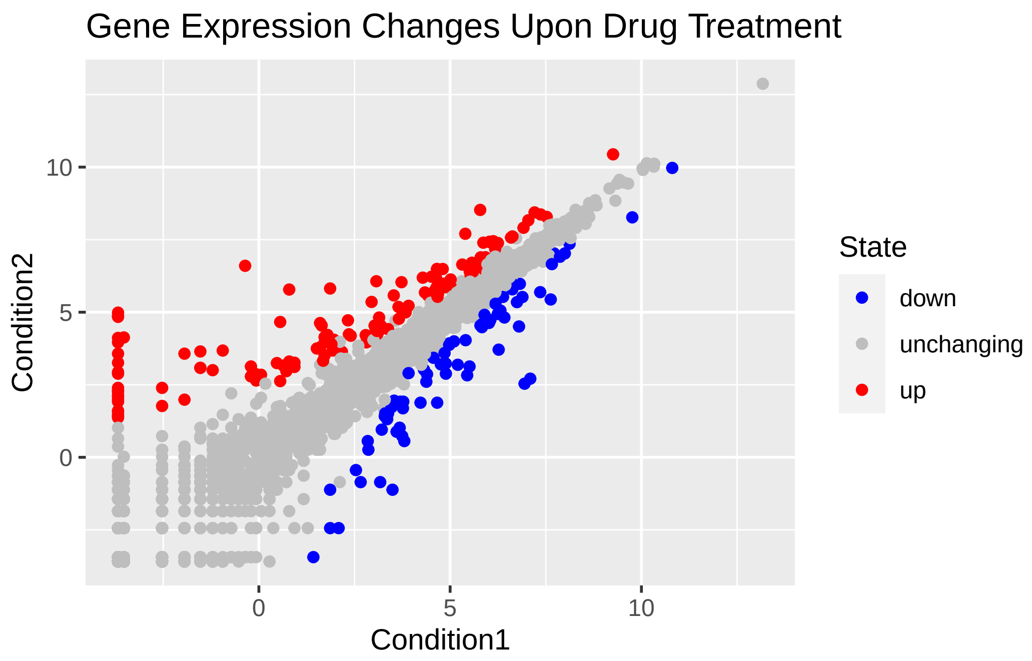
```

      down unchanging         up 
      1.39      96.17       2.44 
```

```
(127/5196) *100
```

```
[1] 2.444188
```

Making graph for Section 6.

```r
p <- ggplot(genes) +
  aes(x=Condition1, y=Condition2, col=State) +
  geom_point() +
  scale_color_manual(values=c("blue", "gray", "red")) + labs(title="Gene Expression Change
p
```



Section 7 of lab.

```r
url <- "https://raw.githubusercontent.com/jennybc/gapminder/master/inst/extdata/gapminder.

gapminder <- read.delim(url)
```

```r
# install.packages("dplyr")  ## un-comment to install if needed
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag
```

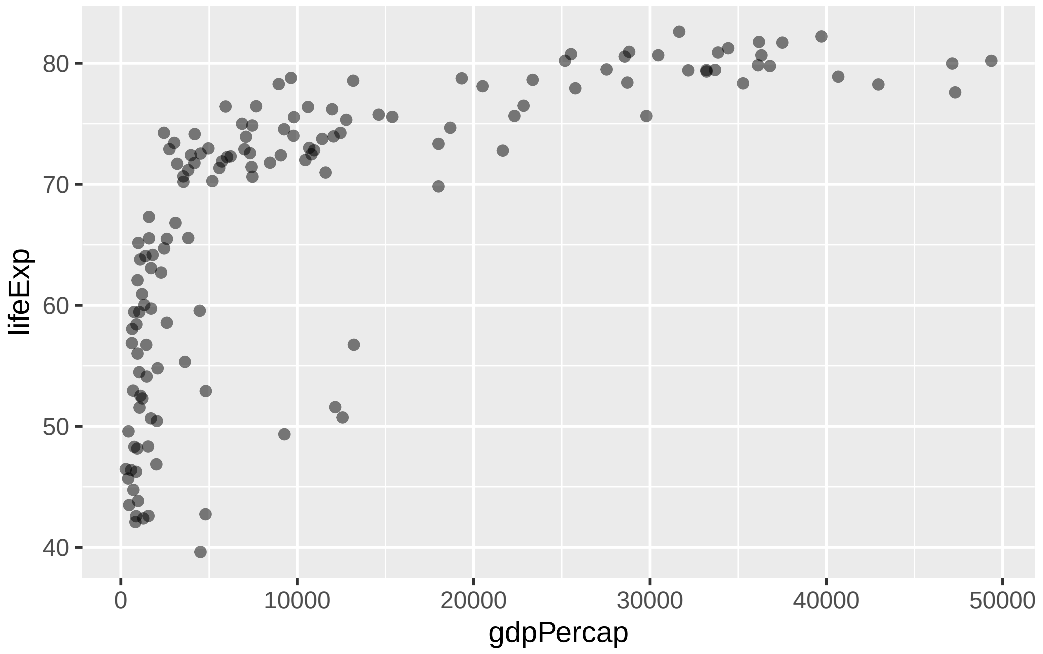
```
The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```r
gapminder_2007 <- gapminder %>% filter(year==2007)
```

Let's consider the gapminder_2007 dataset which contains the variables GDP per capita gdpPercap and life expectancy lifeExp for 142 countries in the year 2007

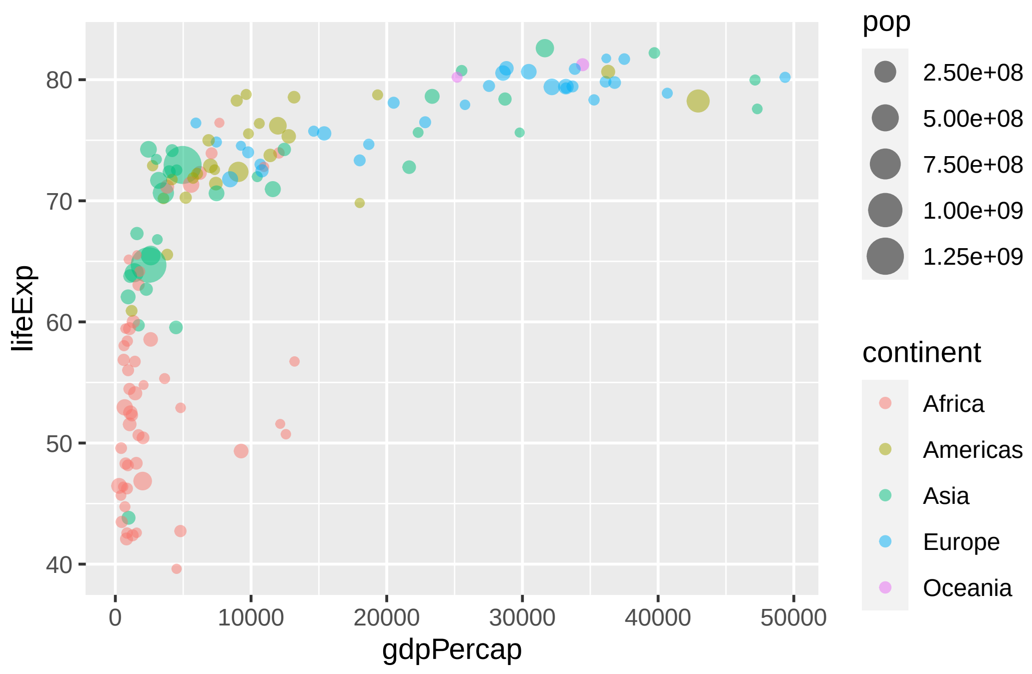
Q. Complete the code below to produce a first basic scater plot of this gapminder_2007 dataset:

```r
ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp) +
  geom_point(alpha=0.5)
```



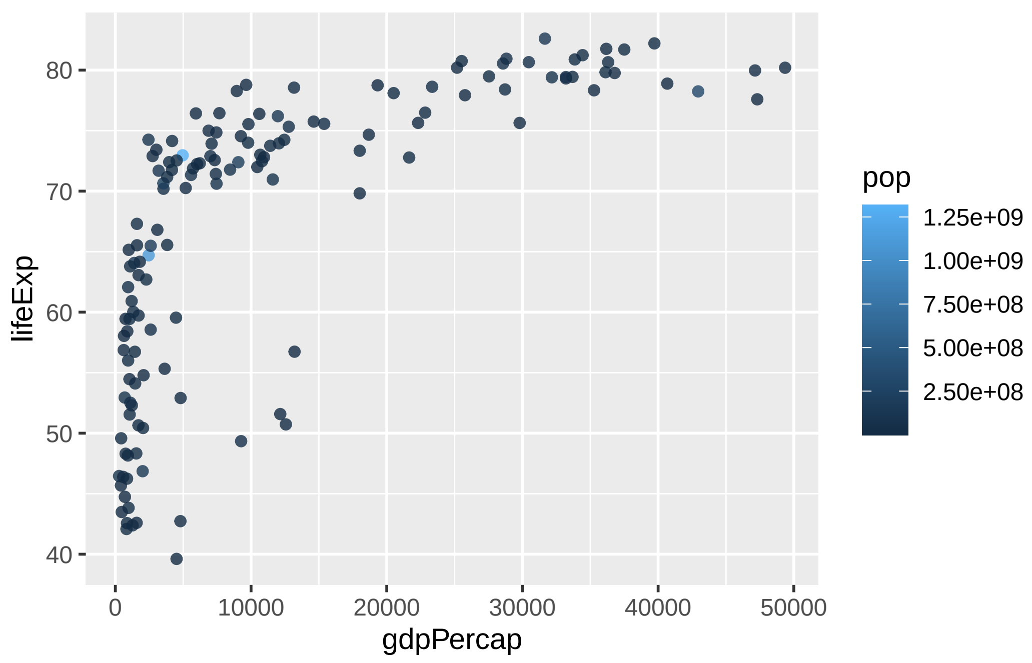
Adding more variables to `aes()`. By mapping the continent variable to the point color aesthetic and the population pop (in millions) through the point size argument to aes() we can obtain a much richer plot that now includes 4 different variables from the data set:

```r
ggplot(gapminder_2007) +
  aes(x=gdpPercap, y=lifeExp, color=continent, size=pop) +
  geom_point(alpha=0.5)
```



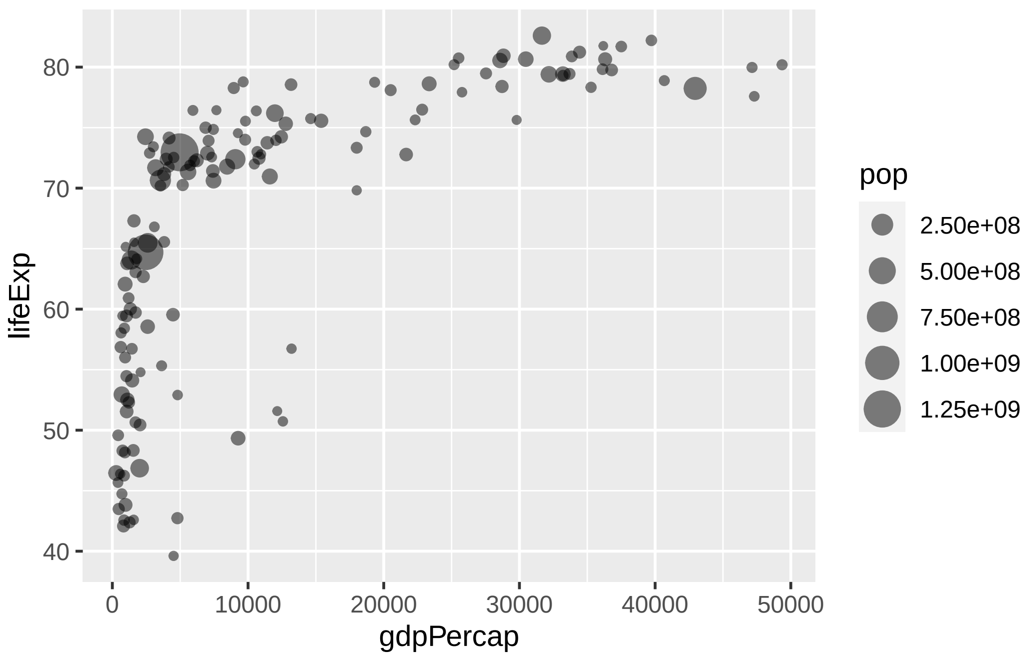
By contrast, let's see how the plot looks like if we color the points by the numeric variable population pop:

```r
ggplot(gapminder_2007) +
  aes(x = gdpPercap, y = lifeExp, color = pop) +
  geom_point(alpha=0.8)
```



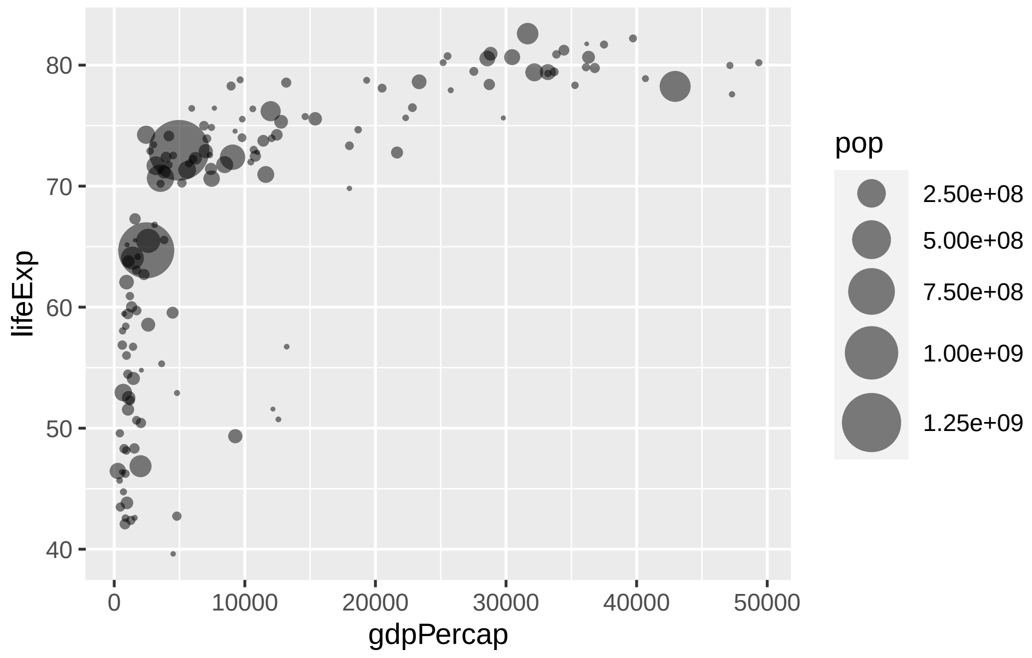
For the gapminder_2007 dataset we can plot the GDP per capita (x=gdpPercap) vs. the life expectancy (y=lifeExp) and set the point size based on the population (size=pop) of each country we can use:

```
ggplot(gapminder_2007) +
  aes(x = gdpPercap, y = lifeExp, size = pop) +
  geom_point(alpha=0.5)
```



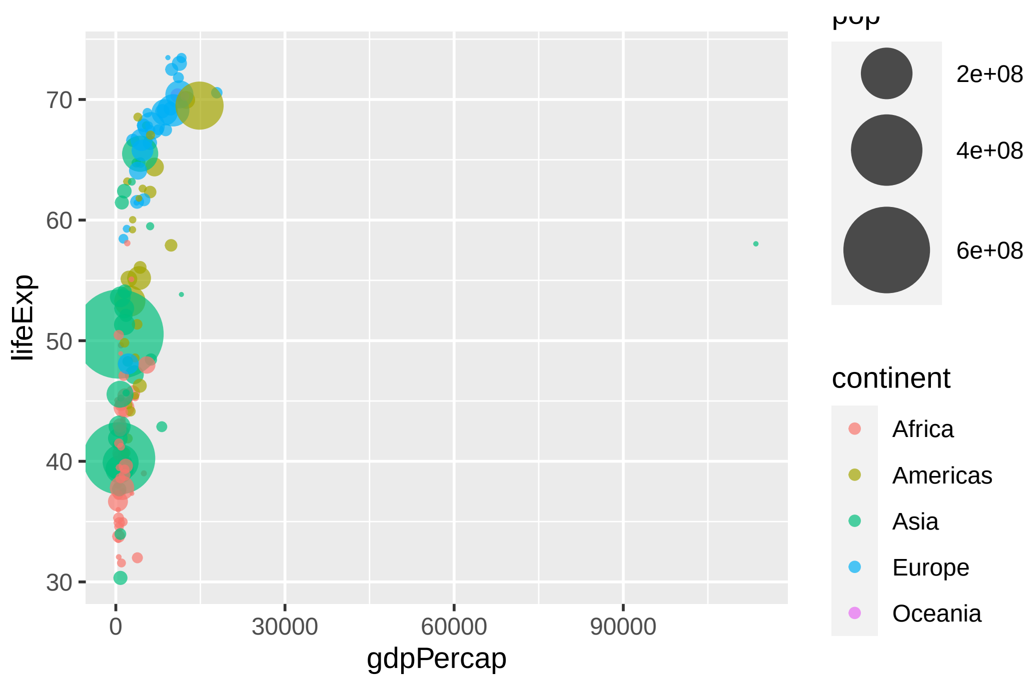
However, if you look closely we see that the point sizes in the plot above do not clearly reflect the population differences in each country. If we compare the point size representing a population of 250 million people with the one displaying 750 million, we can see, that their sizes are not proportional. Instead, the point sizes are binned by default. To reflect the actual population differences by the point size we can use the scale_size_area() function instead. The scaling information can be added like any other ggplot object with the + operator:

```
ggplot(gapminder_2007) +
  geom_point(aes(x = gdpPercap, y = lifeExp,
                 size = pop), alpha=0.5) +
  scale_size_area(max_size = 10)
```



Q. Can you adapt the code you have learned thus far to reproduce our gapminder scatter plot for the year 1957? What do you notice about this plot is it easy to compare with the one for 2007?

```
gapminder_1957 <- gapminder %>% filter(year==1957)
ggplot(gapminder_1957) +
  geom_point(aes(x = gdpPercap, y = lifeExp,
                 size = pop, color=continent), alpha=0.7) +
  scale_size_area(max_size = 15)
```



Q. Do the same steps above but include 1957 and 2007 in your input dataset for ggplot(). You should now include the layer facet_wrap(~year) to produce the following plot: