Class05: Data Vis with ggplot **AUTHOR** Lisa Chen (PID: A17082974) **Graphics systems 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. 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.packages() function. plot(cars) 120 0 100 0 0 80 0 0 9 20 0 0 10 15 20 5 25 speed 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. 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.) ggplot(cars) + aes(x=speed, y=dist) + geom\_point() 100 -75 dist 50 **-**25 **-**20 10 15 25 speed 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() + labs(title="stopping distance of old cars", subtitle = "a silly example plot") + theme\_bw() `geom\_smooth()` using method = 'loess' and formula = 'y  $\sim$  x' stopping distance of old cars a silly example plot 120 80 dist 40 20 15 10 speed url <- "https://bioboot.github.io/bimm143\_S20/class-material/up\_down\_expression.txt"</pre> genes <- read.delim(url)</pre> head(genes) Gene Condition1 Condition2 State A4GNT -3.6808610 -3.4401355 unchanging 1 AAAS 4.5479580 4.3864126 unchanging AASDH 3.7190695 3.4787276 unchanging AATF 5.0784720 5.0151916 unchanging 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 1.39 96.17 2.44 Q. Complete the code below to produce the following plot ggplot(genes) + aes(x=Condition1, y=Condition2) + geom\_point() 10 -Condition2 10 Condition1 p <- ggplot(genes) +</pre> aes(x=Condition1, y=Condition2, col=State) + geom\_point() p + scale\_colour\_manual( values=c("blue","gray","red") ) 10 -State Condition2 down unchanging 0 -10 Condition1 Q. Nice, now add some plot annotations to the p object with the labs() function so your plot looks like the following: p + scale\_colour\_manual(values=c("blue","gray","red")) + labs(title="Gene Expresion Changes Upon Drug Treatment", x="Control (no drug) ", y="Drug Treatment") Gene Expresion Changes Upon Drug Treatment 10 **-Drug Treatment** State down unchanging 10 Control (no drug) url <- "https://raw.githubusercontent.com/jennybc/gapminder/master/inst/extdata/gapmind</pre> gapminder <- read.delim(url)</pre> 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 gapminder\_2007 <- gapminder %>% filter(year==2007) Q. Complete the code below to produce a first basic scater plot of this gapminder\_2007 dataset: ggplot(gapminder\_2007) + aes(x=gdpPercap, y=lifeExp, color=continent, size=pop) + geom\_point(alpha=0.5) 80 pop 2.50e+08 5.00e+08 70 **-**7.50e+08 1.00e+09 1.25e+09 lifeExp continent Africa Americas Asia 50 **-**Europe Oceania 40 -10000 50000 40000 20000 30000 gdpPercap By contrast, let's see how the plot looks like if we color the points by the numeric variable population pop: ggplot(gapminder\_2007) + aes(x = gdpPercap, y = lifeExp, color = pop) +geom\_point(alpha=0.8) 80 -70 **-**

40 -10000 20000 40000 30000 50000 0 gdpPercap adjusting point size: ggplot(gapminder\_2007) + aes(x = gdpPercap, y = lifeExp, size = pop) +geom\_point(alpha=0.5) 80 -70 pop 2.50e+08 5.00e+08 7.50e+08 1.00e+09 1.25e+09 40 -10000 20000 40000 30000 50000 gdpPercap ggplot(gapminder\_2007) +

lifeExp

50 **-**

pop

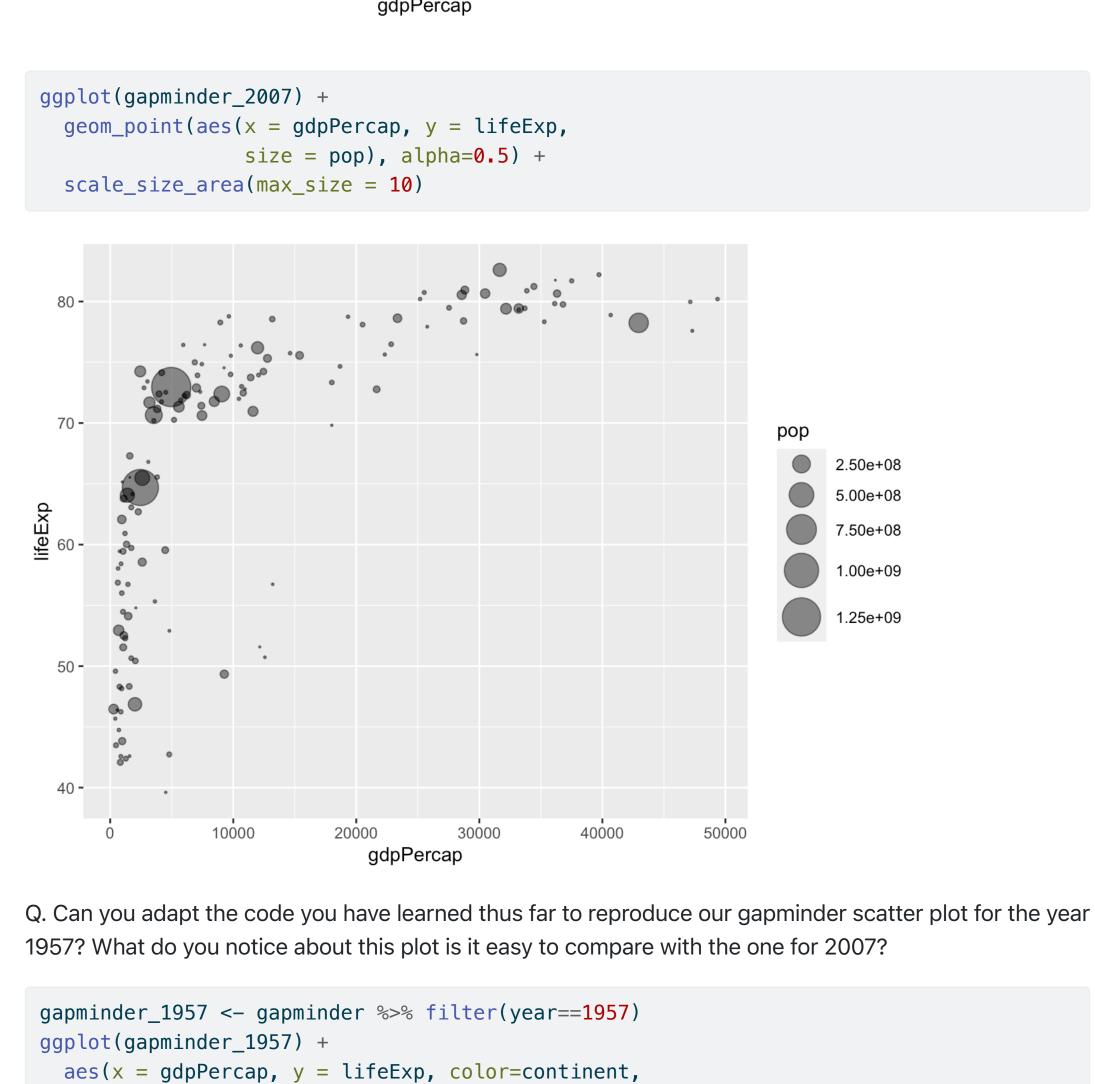
1.25e+09

1.00e+09

7.50e+08

5.00e+08

2.50e+08



size = pop) +

geom\_point(alpha=0.7) +

70 **-**

30 **-**

50 **-**

30 **-**

60000

30000

90000

gdpPercap

scale\_size\_area(max\_size = 15)

30000

60000

gdpPercap

Continent

Africa

Americas

90000

Q. Do the same steps above but include 1957 and 2007 in your input data set for ggplot(). You should

pop

2e+08

4e+08

Asia

continent

Africa

Asia

Americas

Europe

Oceania

Europe

Oceania

```
now include the layer facet_wrap(~year) to produce the following plot:
   gapminder_1957 <- gapminder %>% filter(year==1957 | year==2007)
   ggplot(gapminder_1957) +
     aes(x = gdpPercap, y = lifeExp, color=continent, size = pop) +
     geom_point(alpha=0.7) +
     scale_size_area(max_size = 15) +
     facet_wrap(~year)
                                                 2007
                    1957
                                                                    pop
    80 -
                                                                            2.50e+08
                                                                            5.00e+08
    70 -
                                                                            7.50e+08
                                                                            1.00e+09
lifeExp
                                                                            1.25e+09
```

30000

60000

90000