class 5: data viz with ggplot

Rachel Galleta (PID:16859649)

today we are exploring the **ggplot** package and how to make nice figures in R. there are a lot of ways to make Figures and plot in R. These include:

-so called "base" R -and add on packages like **ggplot2**

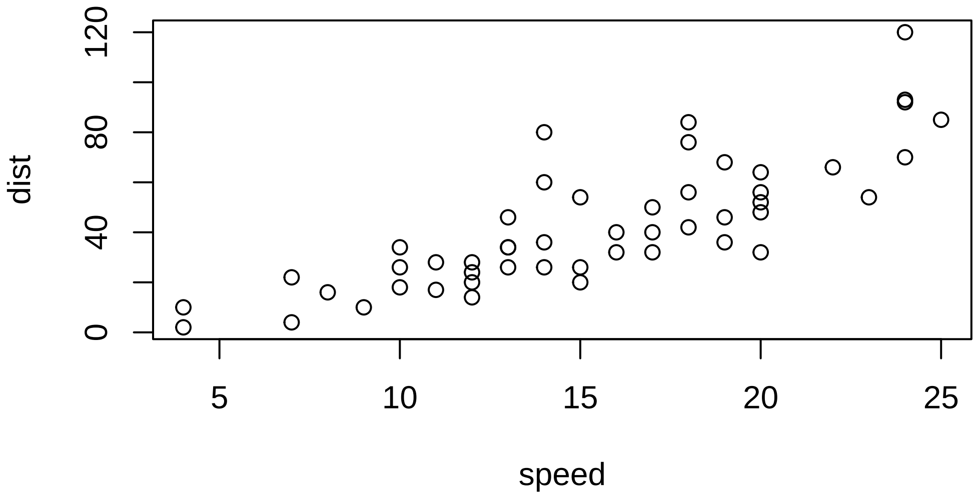
Here is a simple "base" R plot

```r
head(cars)
```

```
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```

we can simply pass to the 'plot()' function

```r
plot(cars)
```



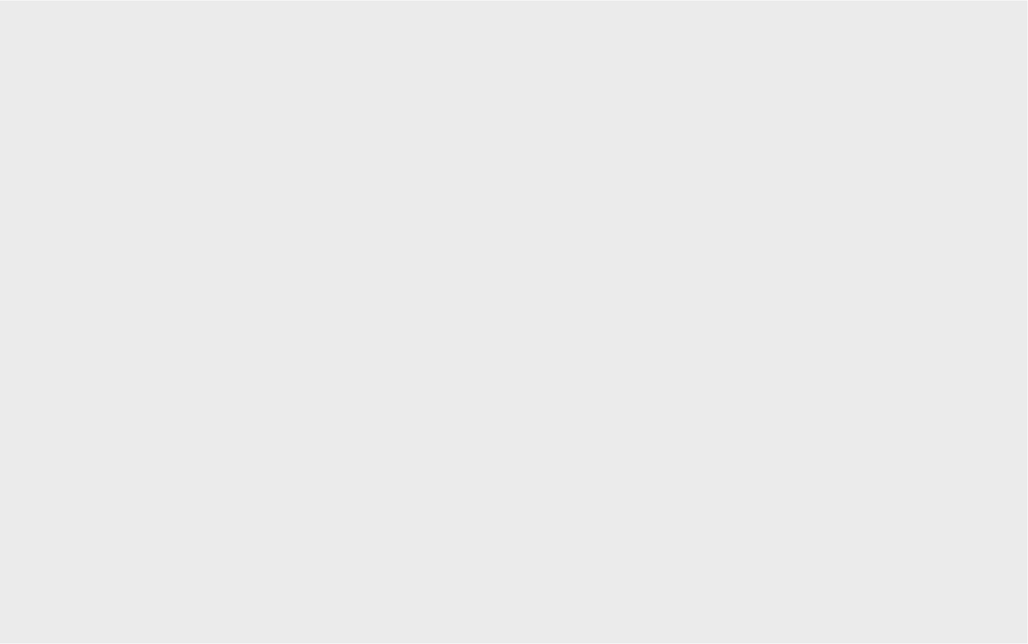
key-point:Base R is quick but not so nice and simple looking in some folks eyes.

lets see how we can plot this with **ggplot**…

1st i need to install this add-on package, for this we use the 'install.package()' function - ** WE DO THIS IN THE CONSOLE, NOT OUR REPORT** This is a one time only deal.

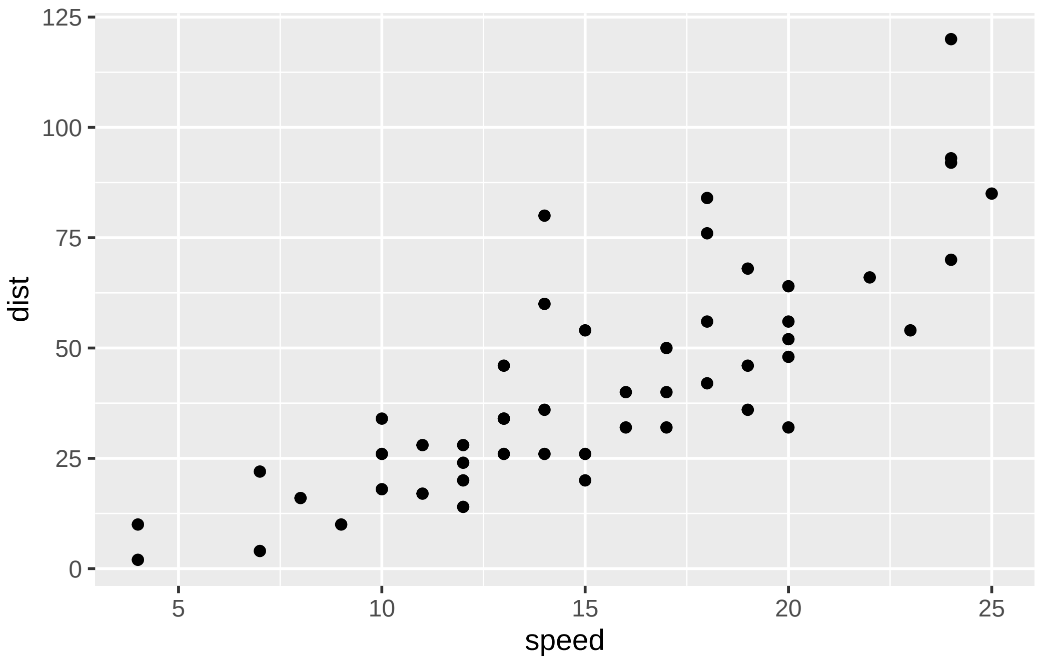
2nd we need to load the package with the 'library()' function every time we want we want to use it.

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

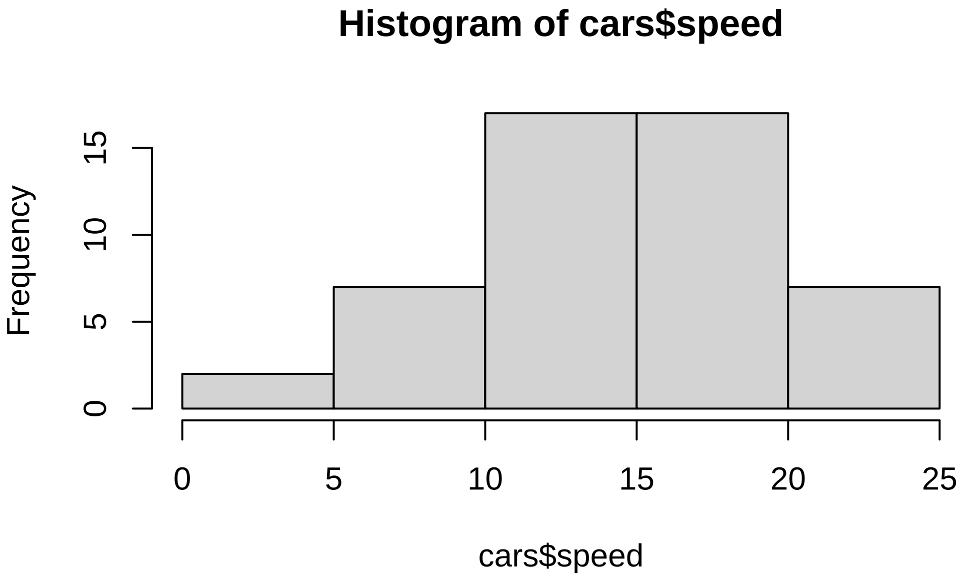


every ggplot is compose of at least three layers, -data(i.e data,frame with the things you want to plot), -aesthetics **aes()** that a map the columns of the data to you put plot features (ie aesthetics) -geom like **geom-point()** that sort how the plot appaers

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



```r
hist(cars$speed)
```



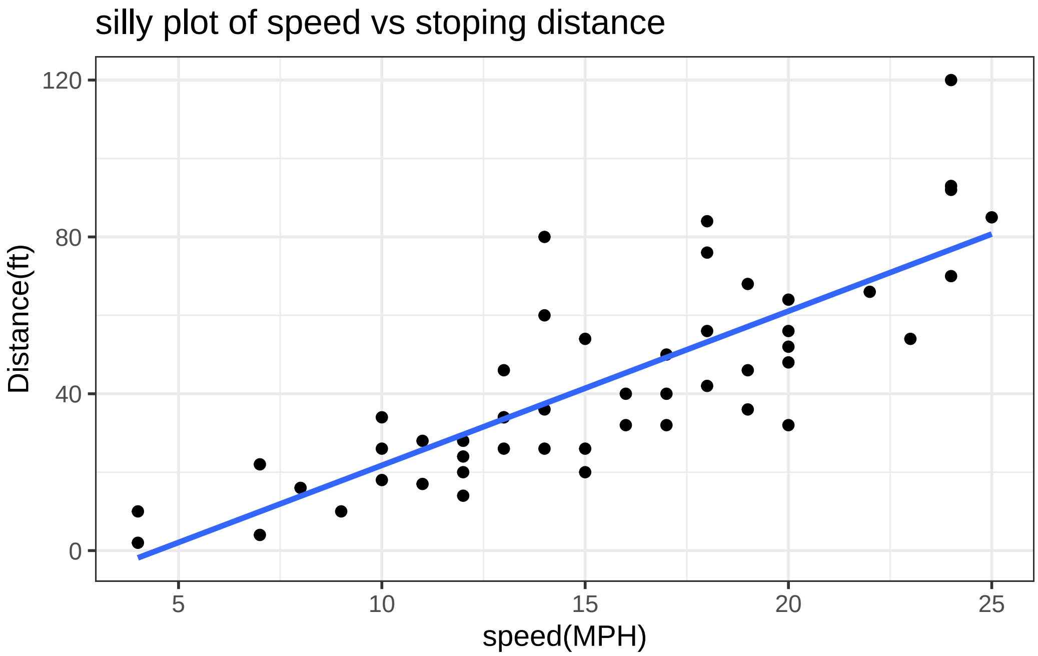
for simple "canned" graphs base is quicker but as things are more custom and elaborate the ggplot wins out..

lets add more layers to our ggplot:

add a line showing the relationship between x and y add a tittle add customs axis label “speed(MPH)” and “distance(ft)” change the theme…

```
ggplot(cars)+
  aes(x=speed,y=dist)+
  geom_point()+
  geom_smooth(method='lm', se=FALSE)+
  labs(title="silly plot of speed vs stoping distance",
       x= "speed(MPH)",
       y= "Distance(ft)") +
  theme_bw()
```

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



read some genes expression data

```
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
```

> Q1. how many genes are in this wee dataset Q2. how many "up" regulated genes are there?

```
sum(genes$State=="up")
```

```
[1] 127
```

a useful function for counting up accurances of things in a vector in the 'table()' functions.

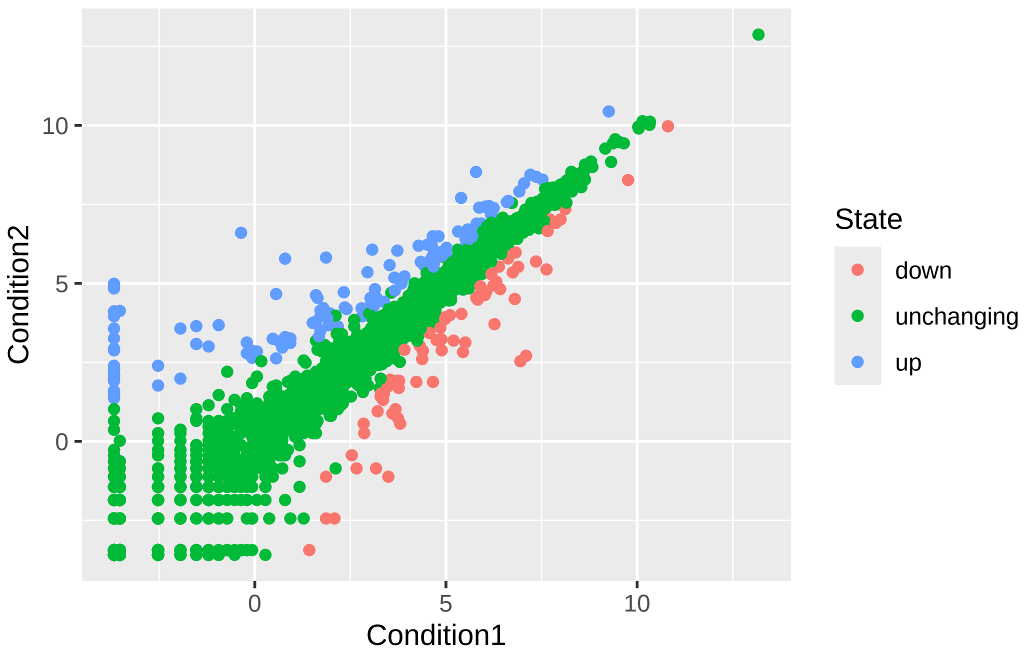
```
table(genes$State)
```

```

      down unchanging         up 
        72       4997        127 
```

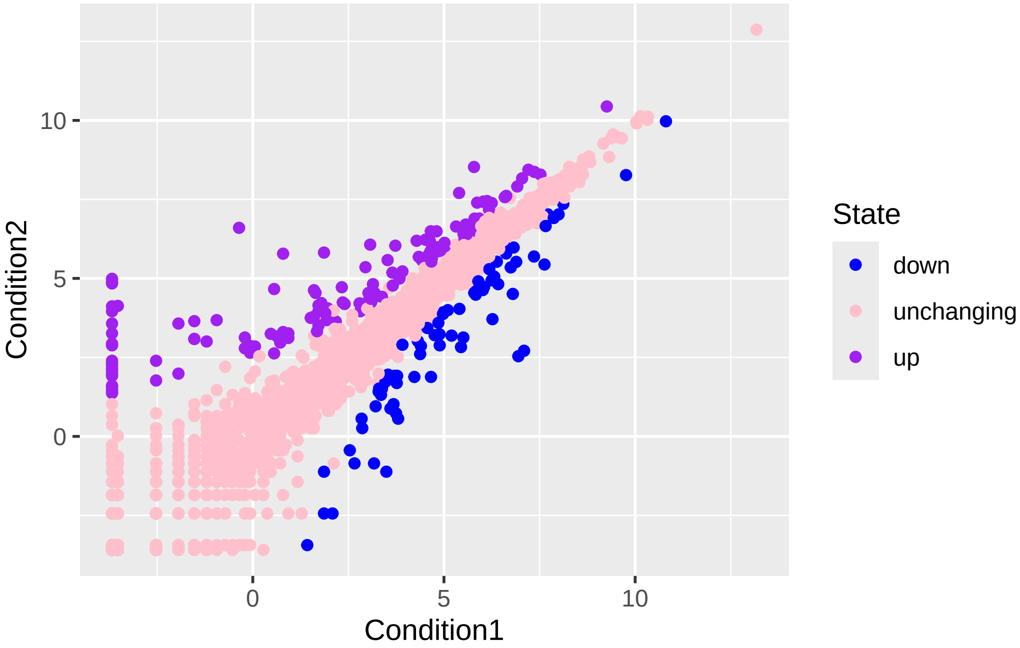
make a v1 figure

```
ggplot(genes) +
    aes(x=Condition1, y=Condition2, col=State) +
    geom_point()
```



```
p <- ggplot(genes) +
    aes(x=Condition1,
        y=Condition2,
        col=State) +
    geom_point()
```

```
p + scale_colour_manual( values=c("blue","pink","purple"))
```



##more plotting

read gapminder

```
# File location online
url <- "https://raw.githubusercontent.com/jennybc/gapminder/master/inst/extdata/gapminder.tsv

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

lets have a wee peak

```
head(gapminder,3)
```

```
      country continent year lifeExp      pop gdpPercap
1 Afghanistan      Asia 1952  28.801  8425333  779.4453
2 Afghanistan      Asia 1957  30.332  9240934  820.8530
3 Afghanistan      Asia 1962  31.997 10267083  853.1007
```

```
tail(gapminder,3)
```

```
      country continent year lifeExp      pop gdpPercap
1702 Zimbabwe    Africa 1997  46.809 11404948  792.4500
```

```
1703 Zimbabwe    Africa 2002  39.989 11926563  672.0386
1704 Zimbabwe    Africa 2007  43.487 12311143  469.7093
```

Q4. how many different country values are in this data set?

```
nrow(gapminder)
```

```
[1] 1704
```

```
length(table(gapminder$country))
```

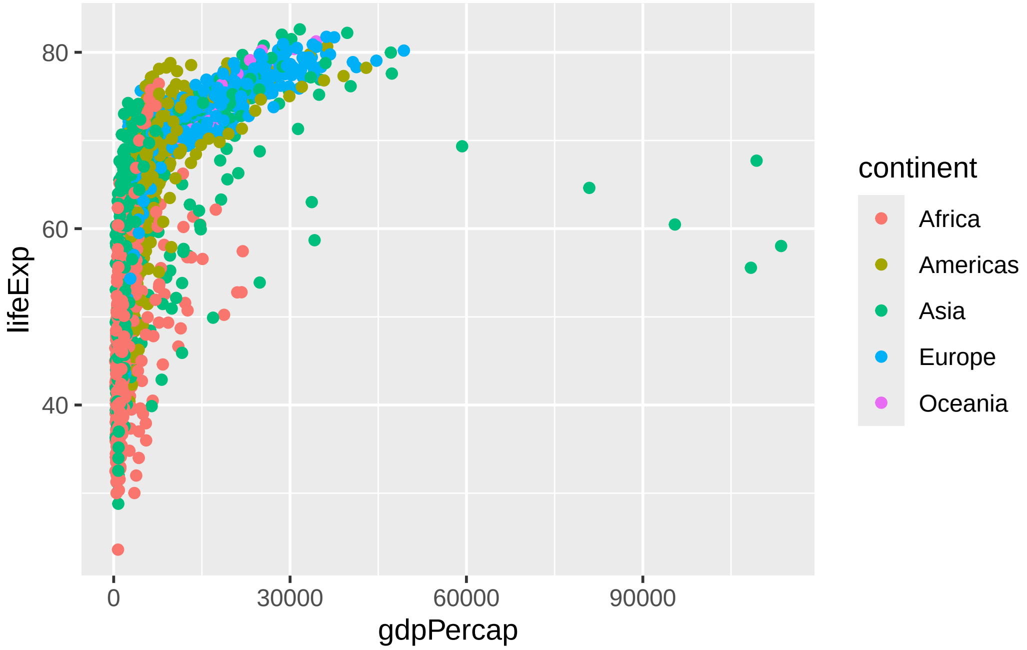
```
[1] 142
```

Q5. How many different continents value are in this data set.

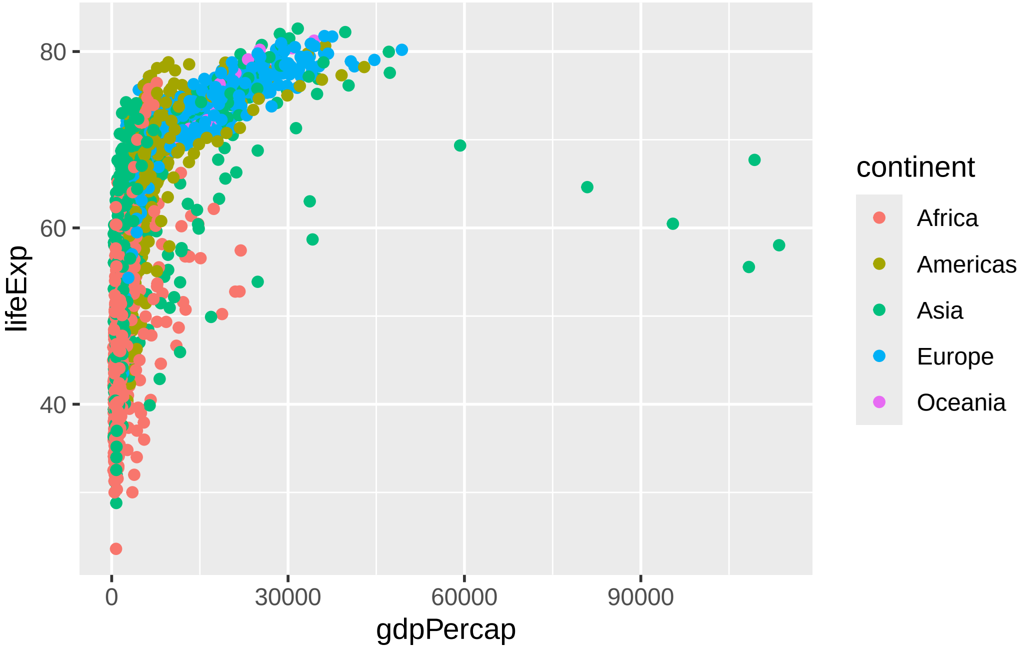
```
unique(gapminder$continent)
```

```
[1] "Asia"     "Europe"   "Africa"   "Americas" "Oceania"
```

```
ggplot(gapminder)+
  aes(gdpPercap,lifeExp,col=continent)+
  geom_point()
```



```
ggplot(gapminder)+
  aes(gdpPercap,lifeExp,col=continent, label=country)+
  geom_point()
```



I can use **ggrepel** package to make more sensible lables here

```
library(ggrepel)
```

i want separate pannel per continent

```
ggplot(gapminder)+
  aes(gdpPercap,lifeExp,col=continent,label=country)+
  geom_point()+
  geom_text_repel()+
  facet_wrap(~continent)
```

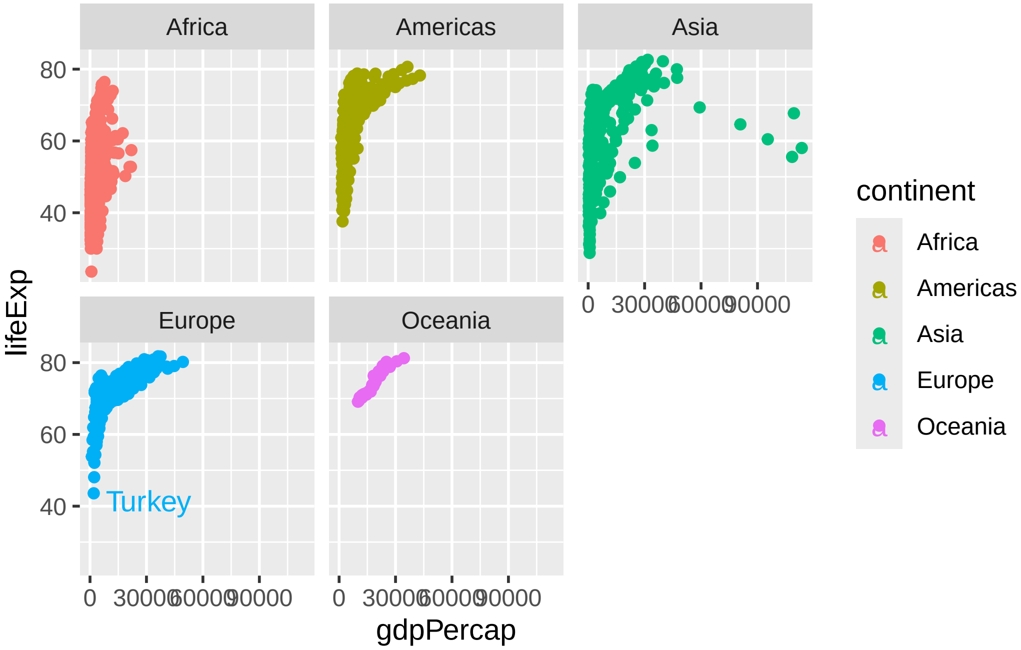
```
Warning: ggrepel: 624 unlabeled data points (too many overlaps). Consider
increasing max.overlaps
```

```
Warning: ggrepel: 359 unlabeled data points (too many overlaps). Consider
increasing max.overlaps
```

```
Warning: ggrepel: 300 unlabeled data points (too many overlaps). Consider
increasing max.overlaps

Warning: ggrepel: 24 unlabeled data points (too many overlaps). Consider
increasing max.overlaps

Warning: ggrepel: 396 unlabeled data points (too many overlaps). Consider
increasing max.overlaps
```



what are the advantages of ggplot over base R: ggplot2 has several advantages over base R graphics:

1. **Layered Grammar**: ggplot uses a consistent, layered approach. You build plots by adding layers (data, aesthetics, geoms) with the + operator, making complex plots easier to construct and modify [1], [3], [2], [4], [5].
2. **Declarative Syntax**: You specify *what* you want to plot, not *how* to draw it. This makes code more readable and easier to maintain [1], [3], [2], [4], [5].
3. **Beautiful Defaults**: ggplot produces attractive, publication-quality figures with sensible defaults, reducing the need for manual tweaking [1], [3], [2], [4].
4. **Customization**: While base R gives pixel-level control, ggplot makes it much easier to customize and add features (like color, shape, themes, legends) by simply adding layers [1], [3], [2], [4], [5].

5. **Consistency**: The same grammar applies to all plot types, so once you learn the system, you can quickly create many kinds of visualizations [1], [3], [2], [4], [5].
6. **Data Mapping**: ggplot excels at mapping data columns to visual features (aesthetics), making it powerful for exploring relationships in data [1], [3], [2], [4], [5].

Base R is fast for simple, exploratory plots, but ggplot is superior for complex, polished, and reproducible graphics. Do you have questions about any of these points?