

Stat 260, Lecture 3, Data Visualization II

Brad McNeney

Load packages

```
library(ggplot2)
data(diamonds)
library(dplyr)
library(gapminder)
data(gapminder)
gapminder <- mutate(gapminder,
                    log10Pop = log10(pop),
                    log10GdpPercap = log10(gdpPercap))
```

References

- ▶ Chapter 1 of printed text, Chapter 3 of online text.
- ▶ ggplot2 cheatsheet at [<https://www.rstudio.com/wp-content/uploads/2016/11/ggplot2-cheatsheet-2.1.pdf>]
- ▶ Wickham (2009) ggplot2: Elegant graphics for data analysis, Chapters 4 and 5.
- ▶ Chang (2012) R graphics cookbook. Available at [<http://www.cookbook-r.com/Graphs/>]

More ggplot Examples

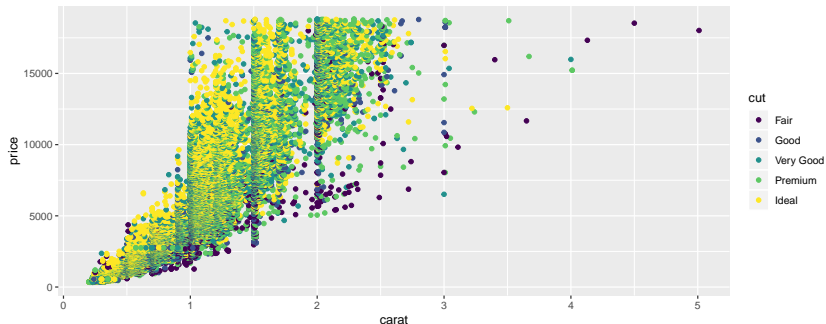
Best way to learn data visualization is look at examples, such as those in the references on the previous slide. In this lecture, more examples to illustrate working with ggplot layers: - data, - aesthetic mapping, - geom, - statistical transformation and - position adjustment

Diamonds dataset

```
p <- ggplot(diamonds,aes(x=carat,y=price,colour=cut)) +  
  geom_point()  
names(p)
```

```
## [1] "data"          "layers"        "scales"        "mapping"       "theme"  
## [6] "coordinates"  "facet"         "plot_env"      "labels"
```

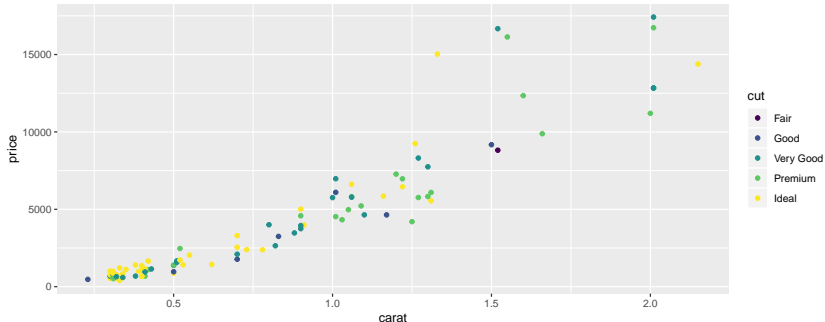
p



Data

- ▶ The data must be a data frame
- ▶ A **copy** of the data is stored in the plot object (changes to the source dataframe do not change plot)
- ▶ Possible to change the data of a plot object with `%>%`, as in

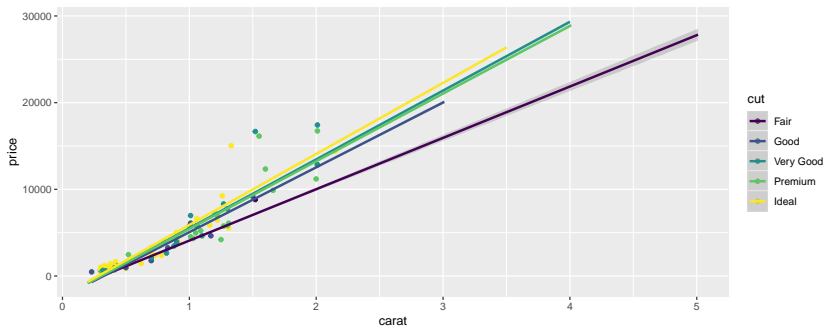
```
set.seed(123)
subdiamonds <- sample_n(diamonds,size=100)
p <- p %>% subdiamonds
p
```



Different data in different layers

- Can specify data for a layer to use.

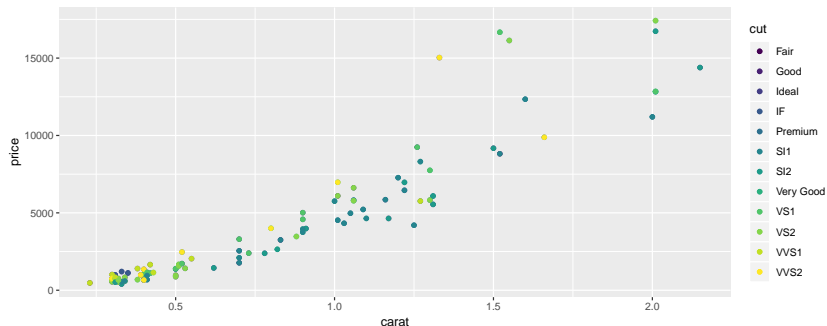
```
p + geom_smooth(data=diamonds, method="lm")
```



Aesthetic mappings

- ▶ We have seen that we can specify a default mapping in the initialization, or specific mappings in the layers
 - ▶ Warning: specifying a mapping twice can have unexpected consequences.

```
p + geom_point(aes(color=clarity))
```



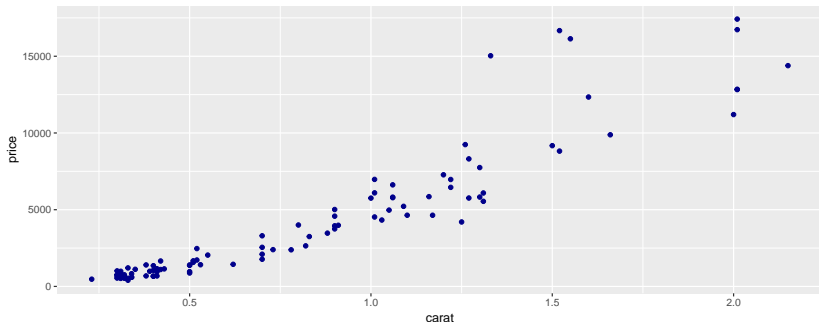
Exercise

Write R code to produce a scatterplot of price versus carat, with different colors for values of clarity, using the subdiamonds dataset.

Setting vs mapping

- ▶ An alternative to mapping aesthetics to variables is to set the aesthetic to a constant.
 - ▶ Set with the layer *parameter*, rather than mapping with `aes()`
 - ▶ E.G., set the color of points on a plot to dark blue

```
ggplot(subdiamonds, aes(x=carat, y=price)) + geom_point(color="darkblue")
```



- ▶ **Exercise:** Redo the above plot with the **mapping** `color="darkblue"` for the `geom_point()` rather than the **parameter** `color="darkblue"`. ""

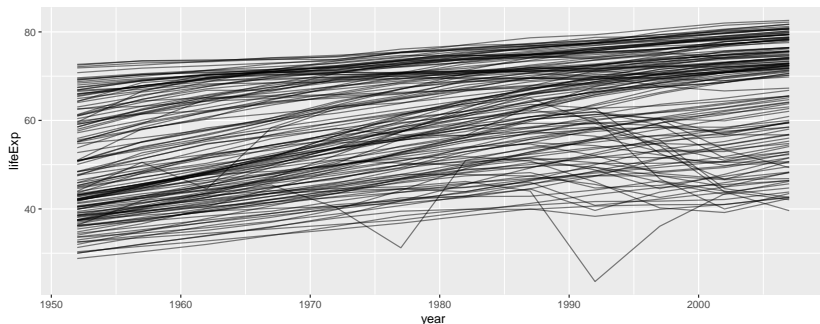
Grouping

- ▶ Many geoms in ggplot2 group observations (rows of the dataframe).
 - ▶ E.G., a boxplot of a quantitative variable by a categorical variable groups observations by the categorical variable.
- ▶ Default group is combinations (interaction) of all categorical variables in the aesthetic specification.
- ▶ If this is not what you want, or if there are no categorical variables, specify group yourself.

gapminder data again: Grouping to plot time series

- For plotting time series (multiple measurements on each observational unit) we want to group by observational unit.

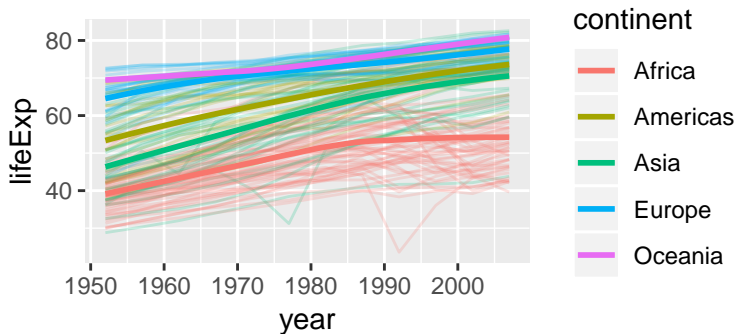
```
ggplot(gapminder,aes(x=year,y=lifeExp,group=country)) + geom_line(alpha=.5)
```



Different groups on different layers

- To add summaries by continent, we need to specify different groups on different layers.

```
ggplot(gapminder, aes(x=year, y=lifeExp, color=continent)) +  
  geom_line(aes(group=country), alpha=0.2) +  
  geom_smooth(aes(group=continent), se=FALSE)
```

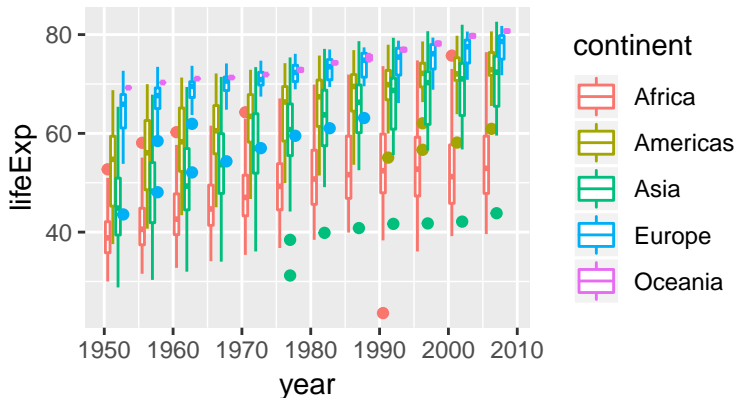


- **Exercise** Redo the above, but remove the mapping in `geom_line()` and specify `group=country` in the default mapping. What grouping variable is used by `geom_line()`? What grouping variable is used by `geom_smooth()`?

Using `interaction()` to specify groups

- Could do boxplots of life expectancy by year **and** continent. (Not recommended – too busy – just using for illustration.)

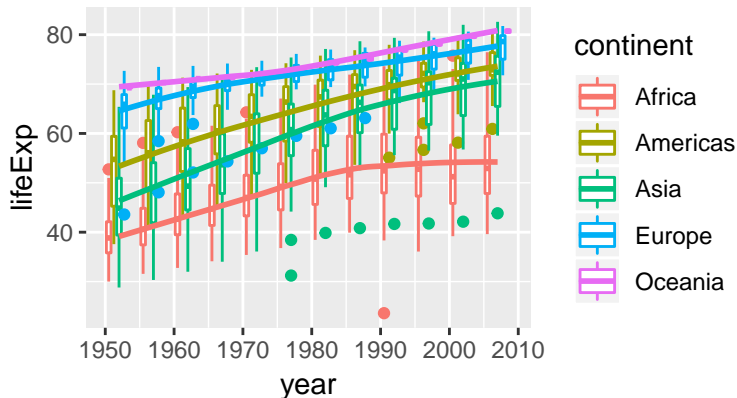
```
ggplot(gapminder, aes(x=year, y=lifeExp,  
  group=interaction(year, continent), color=continent)) +  
  geom_boxplot()
```



Overriding group on a layer

- Boxplots of life expectancy by year and continent on one layer, smoother by continent alone on another.

```
ggplot(gapminder, aes(x=year, y=lifeExp, group=interaction(year, continent),  
                      color=continent)) + geom_boxplot() +  
geom_smooth(aes(group=continent), se=FALSE)
```



Geoms

- ▶ These are the shapes you want on the plot.
 - ▶ See the list of geoms on the cheatsheet.
- ▶ Each has a set of aesthetics that are required for drawing and a set of aesthetics that it understands.
 - ▶ E.G., `geom_point()` requires x and y position, and understands color, size and shape.
- ▶ Aesthetics can also be passed to geoms as parameters.
 - ▶ Recall difference between `geom_point(color="darkblue")` and `geom_point(aes(color="darkblue"))`
- ▶ Each geom also has a default statistic (stat) and positional adjustment.
 - ▶ More on these next.

Stats

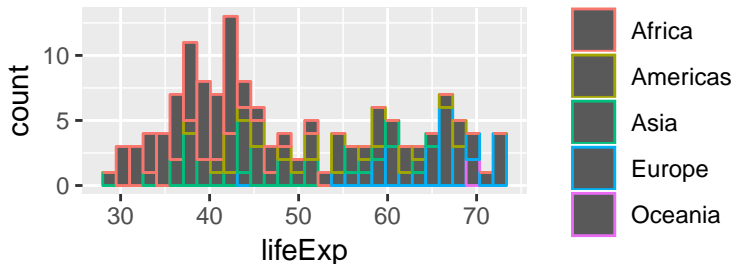
- ▶ Stats are statistical summaries of groups of data points. E.G.,
 - ▶ `stat_smoother()` is a moving average of the y positions over windows of x positions.
 - ▶ `stat_bin()` is a binning of a quantitative variable into bins of a given width
 - ▶ See the cheatsheet for a list.
- ▶ Stats create new variables, and these variables can be mapped to aesthetics.
 - ▶ E.G., `stat_bin()` creates the variables `count`, `density` and `x`
 - ▶ Enclose derived variable name with `..` to use.

```
p <- ggplot(gapminder, aes(x=lifeExp)) +  
  geom_histogram(aes(y= ..density..))
```

Position adjustment

- ▶ See cheatsheet or “Position Adjustments” section of text for more information.
- ▶ Default for most geoms is no adjustment (“identity”)
- ▶ Adjustment to x and/or y position, such as “jitter” can reduce overplotting.
- ▶ Boxplots in recent plot of gapminder data were “dodge”d.
- ▶ Histograms of a continuous variable by a categorical grouping variable are “stack”ed by default.

```
gdat <- filter(gapminder, year==1952)
ggplot(gdat, aes(x=lifeExp, color=continent)) + geom_histogram()
```



Tools for working with layers

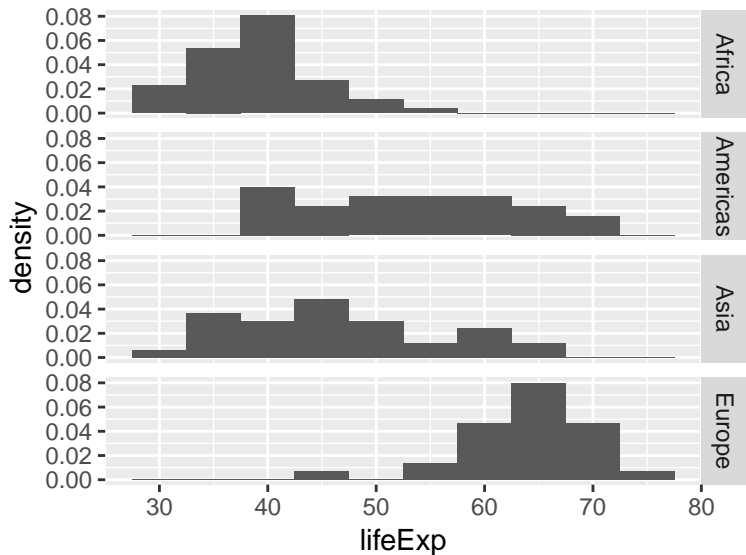
- ▶ Many possible topics
- ▶ In the interest of time, focus on a few
 - ▶ displaying distributions
 - ▶ adding error bars and other measures of uncertainty
 - ▶ annotating a plot

Displaying distributions

- ▶ The standard histogram is `geom_histogram()`.
 - ▶ Displays counts by default, but we have seen how to display as density
 - ▶ Density is better for comparing the shape of distributions
- ▶ To include group information, can stack bars (see previous example), use faceting to produce separate histograms, or superpose density histograms.

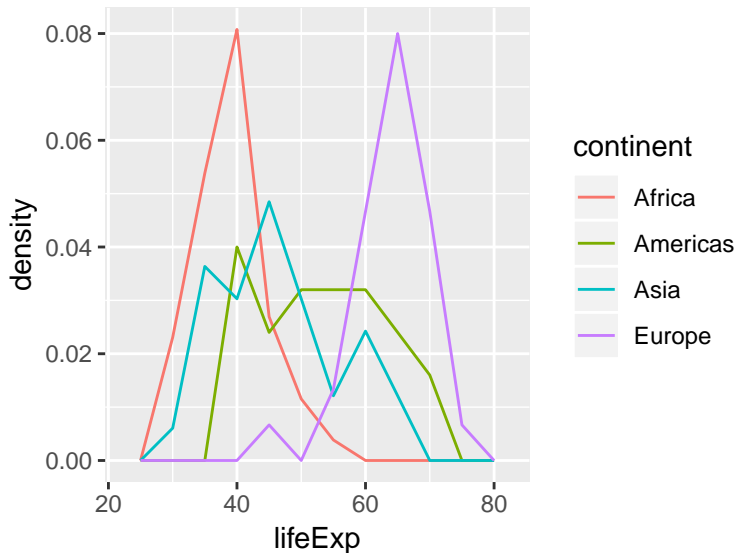
Histograms with faceting

```
gdat <- filter(gdat, continent != "Oceania")  
h <- ggplot(gdat, aes(x=lifeExp))  
h + geom_histogram(aes(y= ..density..), binwidth=5) + facet_grid(continent ~ .)
```



Histograms superposed

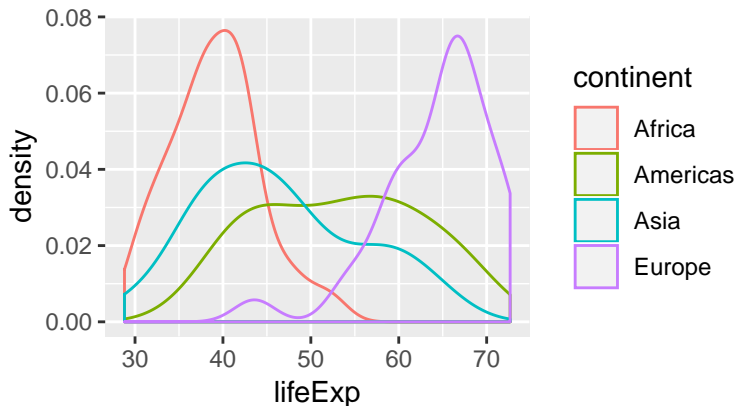
```
h + geom_freqpoly(aes(y=..density..,color=continent), binwidth=5)
```



Density estimation

- ▶ `geom_density()` plots a density estimate
 - ▶ Think of adding up small normal densities centred at each observed datapoint, with the width of each distribution a user-defined parameter
- ▶ Can superpose multiple density plots.

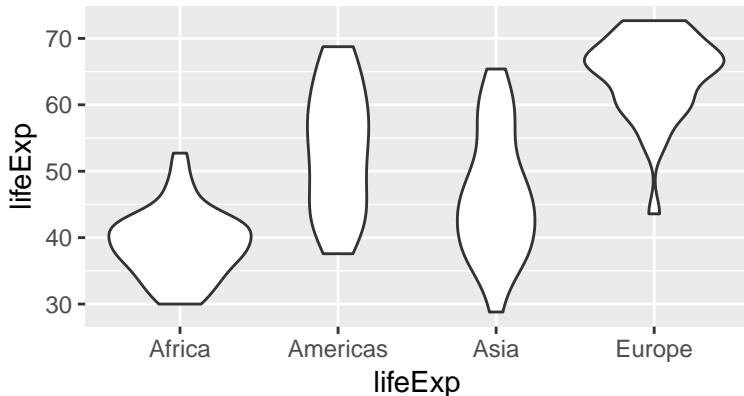
```
h + geom_density(aes(color=continent))
```



Violin plots

- ▶ Instead of superposing, dodge density estimates with a violin plot.
 - ▶ Violins are density estimate and its mirror image displayed vertically

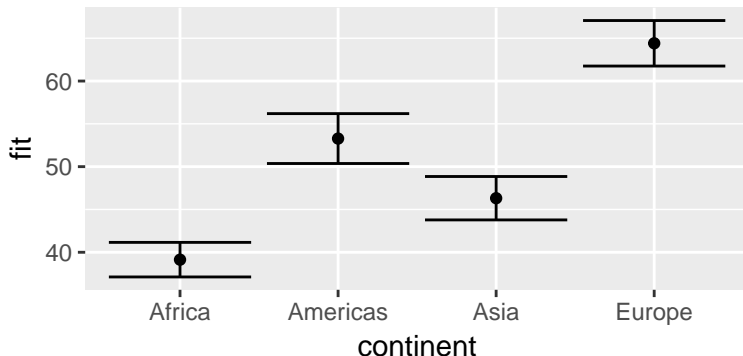
```
h + geom_violin(aes(x=continent,y=lifeExp))
```



Adding measures of uncertainty

- ▶ `geom_smooth()` includes pointwise error bands by default.
- ▶ For factors can add error bars.

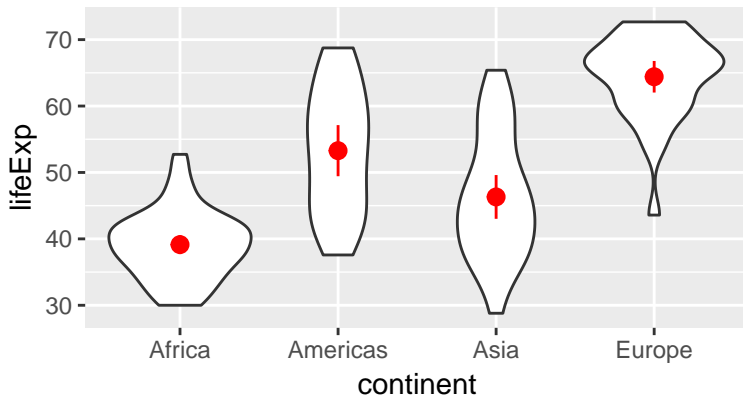
```
gfit <- lm(lifeExp ~ continent, data=gdat)
newdat <- data.frame(continent=c("Africa", "Americas", "Asia", "Europe"))
mm <- data.frame(newdat, predict(gfit, newdata=newdat, interval="confidence"))
ggplot(mm, aes(x=continent, y=fit)) +
  geom_point() + geom_errorbar(aes(ymin=lwr, ymax=upr))
```



Measures of uncertainty with stat summaries

- ▶ Variety of built-in summaries, or can write your own (not covered).
- ▶ Summarize y for different values of x or bins of x values.

```
ggplot(gdat, aes(x=continent, y=lifeExp)) +  
  geom_violin() + # superpose over violin plot  
  stat_summary(fun.data="mean_cl_normal", color="red")
```



Annotating a plot

- ▶ Basic tools for annotating are
 - ▶ `geom_text()` and `geom_label()` to add text
 - ▶ Geoms such as `geom_abline()` to add line segments (see cheatsheet)
 - ▶ `labs()` for adding axis labels, titles, and captions
 - ▶ `annotate()` to add annotations using aesthetics passed as vectors to the function, rather than mapped from a dataframe.
- ▶ Can add annotations one at a time or many at a time
 - ▶ to add many at a time, create a data frame

Many annotations

```
gm07 <- filter(gapminder, year == 2007)
topOilbyGDP <- c("Kuwait", "Guinea", "Norway", "Saudi Arabia")
gdpOil <- filter(gm07, country %in% topOilbyGDP)
ggplot(gm07, aes(x=gdpPerCap, y=lifeExp)) + geom_point() +
  geom_text(data=gdpOil, aes(label=country))
```

