Introduction to ggplot2

Michael Friendly
Psych 6135
http://euclid.psych.yorku.ca/www/psy6135/

Resources: Books

  Complete ggplot2 documentation: http://docs.ggplot2.org/current/

- Kieran Healy, Data Visualization, a Practical Introduction
  A hands-on introduction to data visualization using ggplot2, with a wide range of topics. The online version: https://socviz.co/ is a great example of R bookdown publishing.

- Antony Unwin, Graphical Data Analysis with R
  A gentle introduction to doing visual data analysis, mainly with ggplot2.
  R code: http://www.gradaanwr.net/

- Winston Chang, R Graphics Cookbook: Practical Recipes for Visualizing Data
  Cookbook format, covering common graphing tasks; the main focus is on ggplot2
  R code from book: http://www.cookbook-r.com/Graphs/

Resources: Cheat sheets

- R Studio maintains a large number of cheat sheets, https://www.rstudio.com/resources/cheatsheets/

- Topics:
  - R Studio IDE, Data import, Data transformation (dplyr), Data visualization (ggplot2), R Markdown, ...
  - My collection: R Studio Cheat Sheets

What is ggplot2?

- ggplot2 is Hadley Wickham’s R package for producing “elegant graphics for data analysis”
  - An implementation of the ideas for graphics introduced in Lee Wilkinson’s Grammar of Graphics
  - These ideas and the syntax of ggplot2 help to think of graphs in a new and more general way
  - Produces pleasing plots, taking care of many of the fiddly details (legends, axes, colors, ...)
  - It is built upon the “grid” graphics system
  - It is open software, with a large number of gg_ extensions. See: https://exts.ggplot2.tidyverse.org/gallery/
Some things that should be simple are harder than you'd like in base graphics.

Here, I'm plotting gas mileage (mpg) vs. horsepower and want to use color and shape for different # of cylinders.

But I don't quite get it right!

```r
mtcars$cyl <- as.factor(mtcars$cyl)
plot(mpg ~ hp , data=mtcars,
     col=mtcars$cyl, pch=c(4,6,8)
     legend("topright", legend=levels(mtcars$cyl),
     pch = c(4,6,8),col=levels(mtcars$cyl))
```

colors and point symbols work differently in `plot()` and `legend()`

goal of ggplot2: this should "just work"

In ggplot2, just map the data variables to aesthetic attributes

```r
ggplot() takes care of the rest
```

```r
library(ggplot2)
ggplot(mtcars, aes(x=hp, y=mpg, color=cyl, shape=cyl)) + geom_point(size=3)
```


---

### Grammar of Graphics

- Every graph can be described as a combination of independent building blocks:
  - **data**: a data frame: quantitative, categorical; local or data base query
  - aesthetic mapping of variables into visual properties: size, color, x, y
  - geometric objects ("geom"): points, lines, areas, arrows, ...
  - coordinate system ("coord"): Cartesian, log, polar, map,
Wow! I can really see something there.

How can I enhance this visualization?

Easy: add a geom_smooth() to fit linear regressions for each level of cyl

It is clear that horsepower and # of cylinders are highly related (Duh!)

```r
ggplot(mtcars, aes(x=hp, y=mpg, color=cyl, shape=cyl)) +
  geom_point(size=3) +
  geom_smooth(method="lm", aes(fill=cyl))
```

Grammar of Graphics

- Other GoG building blocks:
  - **statistical transformations** ("stat") -- data summaries: mean, sd, binning & counting, ...
  - **scales**: legends, axes to allow reading data from a plot

A stat builds new variables to plot (e.g., count, prop).

---

**ggplot2: GoG -> graphic language**

- The implementation of GoG ideas in ggplot2 for R created a more expressive language for data graphs
  - **layers**: graph elements combined with "+" (read: "and")
  - **themes**: change graphic elements consistently
ggplot2: layers & aes()

Aesthetic attributes in the `ggplot()` call are passed to `geom_()` layers.

Other attributes can be passed as constants (size=3, color="black") or with `aes(color=, ...)` in different layers.

This plot adds an overall loess smooth to the previous plot. color="black" overrides the `aes(color=cyl)`.

```r
ggplot(mtcars, aes(x=hp, y=mpg)) +
  geom_point(size=3, aes(color=cyl, shape=cyl)) +
  geom_smooth(method="lm", aes(color=cyl, fill=cyl)) +
  geom_smooth(method="loess", color="black", se=FALSE)
```

ggplot2: themes

All the graphical attributes of ggplot2 are governed by themes – settings for all aspects of a plot.

A given plot can be rendered quite differently just by changing the theme.

If you haven't saved the `ggplot` object, `last_plot()` gives you something to work with further.

```r
last_plot() + theme_bw()
```

ggplot2: facets

Facets divide a plot into separate subplots based on one or more discrete variables.

```r
plt <- ggplot(mtcars, aes(x=hp, y=mpg, color=cyl, shape=cyl)) +
  geom_point(size=3) +
  geom_smooth(method="lm", aes(fill=cyl)) +
  geom_smooth(method="loess", color="black", se=FALSE)
plt + facet_wrap(~cyl)
```

labeling points: geom_text()

Sometimes it is useful to label points to show their identities. `geom_text()` usually gives messy, overlapping text.

```r
plt2 <- ggplot(mtcars, aes(x=wt, y=mpg)) +
  geom_point(color = 'red', size=2) +
  geom_smooth(method="loess") +
  labs(y="Miles per gallon", x="Weight (1000 lbs.)") +
  theme_classic(base_size = 16)
plt2 + geom_text(aes(label = rownames(mtcars)))
```

Note the use of `theme_classic()` and better axis labels.

But this is still messy: wouldn’t want to publish this.
labeling points: geom_text_repel()

```r
library(ggrepel)
plt2 + geom_text_repel(aes(label = rownames(mtcars)))
```

`geom_text_repel()` in the ggrepel package assigns repulsive forces among points and labels to assure no overlap. Some lines are drawn to make the assignment clearer.

labeling points: selection

```r
mod <- loess(mpg ~ wt, data = mtcars)
resids <- residuals(mod)
mtcars$label <- ifelse(abs(resids) > 2.5, rownames(mtcars), "")
plt2 + geom_text_repel(aes(label = mtcars$label))
```

It is easy to label points selectively, using some criterion to assign labels to points.

Here, I:
1. fit the smoothed loess curve,
2. extract residuals, $r_i$
3. assign labels where $|r_i| > 2.5$
4. add the text layer

ggplot2: coords

Coordinate systems, `coord_*()` functions, handle conversion from geometric objects to what you see on a 2D plot:
- A simple bar chart, standard coordinates
- A pie chart is just a bar chart in polar coordinates!

```r
p <- ggplot(df, aes(x = "", y = value, fill = group)) + geom_bar(stat = "identity")
p + coord_polar("y", start = 0)
```

Anatomy of a ggplot

Other details of ggplot concern `scales` You can control everything...
ggplot objects

Traditional R graphics just produce graphical output on a device. However, `ggplot()` produces a “ggplot” object, a list of elements.

```r
> names(plt)
[1] "data" "layers" "scales" "mapping" "theme" "coordinates"
[7] "facet" "plot_env" "labels"
```

What methods are available?

```r
> methods(class="gg")
[1] +
> methods(class="ggplot")
[1] grid.draw plot print summary
```

The “gg” class provides the “+” method.

The “ggplot” class provides other, standard methods.

Playfair: Balance of trade charts

In the *Commercial and Political Atlas*, William Playfair used charts of imports and exports from England to its trading partners to ask “How are we doing?”

Here is a re-creation of one example, using ggplot2. How was it done?

```r
> data(EastIndiesTrade, package="GDAdata")
> head(EastIndiesTrade)
      Year Exports Imports
1 1700     180     460
2 1701     170     480
3 1702     160     490
4 1703     150     500
5 1704     145     510
6 1705     140     525
```

ggplot thinking

I want to plot two time series, & fill the area between them.

Start with a line plot of Exports vs. Year: `geom_line()`

Add a layer for the line plot of Imports vs. Year:

```r
c1 <- ggplot(EastIndiesTrade, aes(x=Year, y=Exports)) +
ylim(0,2000) + geom_line(colour="black", size=2) + geom_line(aes(x=Year, y=Imports), colour="red", size=2)
```

Fill the area between the curves: `geom_ribbon()`

```r
c1 <- c1 + geom_ribbon(aes(ymin=Exports, ymax=Imports), fill="pink") + ylab("Exports and Imports")
```

That looks pretty good. Add some text labels using `annotate()`

```r
c1 <- c1 +
annotate("text", x = 1710, y = 0, label = "Exports", size=4) +
annotate("text", x = 1770, y = 1620, label = "Imports", color="red", size=4) +
annotate("text", x = 1732, y = 1950, label = "Balance of Trade to the East Indies", color="black", size=5)
```

Finally, change the theme to b/w:

```r
c1 <- c1 + theme_bw()
```
Plot what you want to show

Playfair’s goal was to show the balance of trade with different countries. Why not plot Exports – Imports directly?

c2 <- ggplot(EastIndiesTrade, aes(x = Year, y = Exports - Imports)) + geom_line(colour="red", size=2) + ylab("Balance = Exports - Imports") + geom_ribbon(aes(ymin=Exports-Imports, ymax=0), fill="pink", alpha=0.5) + annotate("text", x = 1710, y = -30, label = "Our Deficit", color="black", size=5) + theme_bw()

aes(x=, y=) can use expressions calculated from data variables

Composing several plots

ggplot objects use grid graphics for rendering

The gridExtra package has functions for combining or manipulating grid-based graphs

library(gridExtra)
ggrid.arrange(c1, c2, nrow=1)

Saving plots: ggsave()

- If the plot is on the screen
  ggsave("path/filename.png") # height=, width=
- If you have a plot object
  ggsave(myplot, file="path/filename.png")
- Specify size:
  ggsave(myplot, "path/filename.png", width=6, height=4)
- any plot format (pdf, png, eps, svg, jpg, …)
  ggsave(myplot, file="path/filename.jpg")
ggsave(myplot, file="path/filename.pdf")

Building a custom graph

Custom graphs can be constructed by adding graphical elements (points, lines, text, arrows, etc.) to a basic ggplot()

John Arbuthnot: data on male/female sex ratios:

```
> data(Arthurbothn, package="HistData")
> head(Arthurbothn[,c(1:3,6,7)])

  Year Males Females Ratio Total
1 1629  5218    4683 1.114 9.901
2 1630  4858    4457 1.090 9.315
3 1631  4422    4102 1.078 8.524
4 1632  4994    4590 1.088 9.584
5 1633  5158    4839 1.066 9.997
6 1634  5035    4820 1.045 9.855
```

Arbuthnot didn’t make a graph. He simply calculated the probability that in 81 years from 1629–1710, the sex ratio would always be > 1. The first significance test!
Building a custom graph

```r
ggplot(Arbuthnot, aes(x=Year, y=Ratio)) +
  ylim(1, 1.20) +
  ylab("Sex Ratio (M/F)") +
  geom_point(pch=16, size=2)
```

Start with a basic scatterplot, Ratio vs. Year

An R script for this example is available at:

Building a custom graph

```r
ggplot(Arbuthnot, aes(x=Year, y=Ratio)) +
  ylim(1, 1.20) +
  ylab("Sex Ratio (M/F)") +
  geom_point(pch=16, size=2) +
  geom_line(color="gray")
```

Connect points with a line

Building a custom graph

```r
ggplot(Arbuthnot, aes(x=Year, y=Ratio)) +
  ylim(1, 1.20) +
  ylab("Sex Ratio (M/F)") +
  geom_point(pch=16, size=2) +
  geom_line(color="gray") +
  geom_smooth(method="loess", color="blue",
              fill="blue", alpha=0.2) +
  geom_smooth(method="lm", color="darkgreen",
              se=FALSE)
```

Add smooths:
- loess curve
- linear regression line

```r
arbuth <- last_plot()
```

# save what we have so far
arbuth <- last_plot()

Building a custom graph

```r
arbuth +
  geom_hline(yintercept=1, color="red", size=2) +
  annotate("text", x=1645, y=1.01, label="Males = Females", color="red", size=5)
```

Add horizontal reference line & text label
**Building a custom graph**

```
arbuth +
  geom_hline(yintercept=1, color="red", size=2) +
  annotate("text", x=1645, y=1.01, label="Males = Females", color="red", size=5) +
  label="Arbuthnot's data on the Male / Female Sex Ratio", size=5.5)
```

Add figure title

![Graph showing Arbuthnot's data on the Male/Female Sex Ratio](image)

**Change the theme and font size**

```
arbuth +
  geom_hline(yintercept=1, color="red", size=2) +
  annotate("text", x=1645, y=1.01, label="Males = Females", color="red", size=5) +
  annotate("text", x=1680, y=1.19,
              label="Arbuthnot's data on the Male / Female Sex Ratio", size=5.5) +
  theme_bw() + theme(text = element_text(size = 16))
```

Consulting for Guerry

Guerry (1833) made shaded maps of France to determine if crime was related to literacy & other factors

Guerry: Mes cartes sont très jolies, non? But how can I go further?
MF: Make scatterplots! Add smooths & data ellipses. See you next week at Café Lillas
Guerry: Les boissons sont sur moi!
Building Guerry’s plots

```r
ggplot(aes(x=Literacy, y=Crime_pers/1000), data=Guerry) + geom_point(size=2)
```

Start with a basic scatterplot

Add data ellipses to show correlation
- 68% ~ mean ± 1 sd
- 95% ~ mean ± 2 sd

Guerry’s plots: Add smooths

```r
ggplot(aes(x=Literacy, y=Crime_pers/1000), data=Guerry) + geom_point(size=2) +
stat_ellipse(level=0.68, color="blue", size=1.2) +
stat_ellipse(level=0.95, color="gray", size=1, linetype=2) +
geom_smooth(method="lm", formula=y~x, fill="lightblue") +
geom_smooth(method="loess", formula=y~x, color="red", se=FALSE)
```

Add lm() and loess() smooths
- lm shows regression slope
- loess diagnoses possible non-linearity

Coffee break: save the current plot object

```r
gplot <- last.plot()
```

Guerry’s plots: Styling

```r
gplot + theme_bw() +
theme(text = element_text(size=18))
```

Guerry: I want to publish this! But need to make axis labels larger

MF:
- Change the basic theme to theme_bw()
- Increase the font size for all text
- You can change the style of anything you want
Guerry’s plots: Labeling

Guerry: OK, but I see some unusual points. What are they?
MF: Need to calculate “unusualness” – Mahalanobis $D^2$ squared distance from centroid

```r
library(ggrepel)
gplot +
  theme_bw() +
  theme(text = element_text(size=18)) +
  geom_label_repel(aes(label=Department),
    data = gdf[gdf$dsq > 4.6,])
```

$$D^2 = (x - \bar{x})' S^{-1} (x - \bar{x})'$$