Data Visualization using R

C. Tobin Magle, PhD

Based on http://www.datacarpentry.org/R-ecology-lesson/
Outline

• Why ggplot2?

• **Basic elements** (data, aesthetics, geoms)

• **Modifications** (transparency, color, grouping, themes)

• **Exporting plots** (ggsave)
Prerequisites

• Install R and R studio
  http://www.datacarpentry.org/R-ecology-lesson/index.html#setup_instructions

• See the Basic Analysis with R lesson if you’re unfamiliar with R or R studio
  http://libguides.colostate.edu/data-and-donuts/r-analysis
Why ggplot2?

- Reproducibility
- Part of the tidyverse
- Pretty by default
- Customizable

http://varianceexplained.org/r/why-i-use-ggplot2/
Data set: survey of small animals

- Stored in a data frame

- **Rows**: observations of individual animals

- **Columns**: Variables that describe the animals
  - Species, sex, date, location, etc
Setup project


• Unzip the download file

• Drag “uw-r-workshop-master” folder to the documents folder

• Open the .Rproj file
Setup environment

• Check if tidyverse is installed
  • Will automatically load if it’s installed

• If not, install and load it

• Read the data into an object called `surveys_c`

```r
require("tidyverse")
install.packages("tidyverse")
library("tidyverse")
surveys_c <- read_csv(
  file = "complete_surveys.csv")
```
**ggplot2 makes modular plots**

geoms + Aesthetic mappings + Plot annotations = Full plot

[Diagram showing modular components of a plot]

[Website link:](https://vita.had.co.nz/papers/layered-grammar.html)
“Grammar” of ggplot2 graphics

• **data**: the data you want to plot

• **Aesthetic mappings**: mapping variables to axes

• **geoms**: how you want the data to be drawn
  - Ex: points, lines, bars
ggplot2 functions

- **ggplot()**: creates a ggplot object
- **aes()**: draws axes based on arguments
- **geom_XXX()**: draws points/lines etc.
- **+**: adds components to plot
  - Modular structure
Plotting data

- **ggplot()**:  
  - *data* table to plot

- **mapping = aes()** function  
  - *x* axis variable  
  - *y* axis variable

- **geom()** - what to draw data as

```r
ggplot(data = surveys_complete, 
       aes(x = weight, 
            y = hindfoot_length)) + 
geom_point()
```
Specify data

- **ggplot()**: Creates a plot object
- **Input**: data table
- **Output**: blank plot

```
ggplot(data = surveys_complete)
```
Aesthetic mappings

• Draw the axes

• Input:
  • `mapping` argument
  • `aes()` function
    • `x = x axis variable`
    • `y = y axis variable`

• Output: plot with axes, no data
Geometries

- **geom_point()**: draws data points

- **Input**: `geom_` function
  - requires `+` operator
  - Whitespace matters

- **Output**: scatterplot

```
ggplot(data = surveys_complete, 
      aes(x = weight, 
          y = hindfoot_length)) + geom_point()
```
Modifying ggplots

• Color and transparency
• Other geoms
• Themes
Add transparency

- Add argument to `geom_point()`

- Reduce transparency of points

- **Input**: `alpha = 0.1`
  - 1/10 opacity
  - Range: 0-1

```r
ggplot(surveys_complete, 
aes(x = weight, 
y = hindfoot_length)) + 
geom_point(alpha = 0.1)
```
Add color

• Change point colors to blue

• **Input:** color argument

• **Output:** blue points

```r
ggplot(surveys_complete, aes(x = weight, 
y = hindfoot_length)) + 
geom_point(alpha = 0.1, 
color = "blue")
```
Add color by species

• Color points by variable

• **Input:** color = <factor variable>

• **Must be inside** `aes()`
  • Anything that references the dataset
  • Like x and y variables

• **Output:** a colored plot

```r
ggplot(data = surveys_complete, 
  aes(x = weight, 
      y = hindfoot_length)) + 
geom_point(alpha = 0.1, 
  aes(color = species_id))
```
Exercise 1

• Use the previous example as a starting point.

• Add color to the data points according to the plot from which the sample was taken (plot_id).

• Hint: Check the class for plot_id. Consider changing the class of plot_id from integer to factor. Why does this change how R makes the graph?
Modifying ggplots

- Color and transparency

- Other geoms
  - For categorical data
  - For time series data

- Themes
Plotting categories

- geom_point doesn’t work for everything
  - Overplotted
  - How can we spread out the data?
- Use a different geom:
  - geom_jitter()
Geom jitter

- `geom_jitter()` - spreads out the points
- Better but still overplotted
- What kind of plot does this look like?

```r
ggplot(surveys_complete, aes(x = species_id, y = hindfoot_length)) + geom_jitter(alpha = 0.1)
```
Box plots

- Categorical vs. numeric

- `geom_boxplot()` - draws a boxplot
  - Median - middle line
  - Quartiles - edge of box
  - Outliers - points outside the box

- Shows the spread of the data better

```r
ggplot(surveys_complete, aes(x = species_id, y = hindfoot_length)) + geom_boxplot()
```
Layering geoms

- Use the `+` operator to overlay geoms

```r
ggplot(surveys_complete, 
       aes(x = species_id, 
           y = hindfoot_length)) + geom_boxplot() + geom_jitter(alpha = 0.1, 
                                                                 color = "tomato")```
Order matters

- Reverse order of the geoms

```r
ggplot(surveys_complete, 
  aes(x = species_id, 
      y = hindfoot_length)) + 
geom_jitter(alpha = 0.3, 
            color = "tomato") + 
geom_boxplot()
```
Exercise 2: Violin plot

• Plot the same data as in the previous example, but as a Violin plot
  • Hint: see `geom_violin()`.

• What information does this give you about the data that a box plot does?
Plotting data over time

• Visualize trends over time
  • Usually a summary statistic

• Likely need to reshape data

• Consider which variables you need to build the plot
Time series data

• Create a new table with necessary data

• For # observed in each species over time:
  • Year
  • Species
  • n (number of each species in each year)

• Syntax:

```r
yearly_counts <- surveys_complete %>%
  count(year, species_id)
```
Time series data

• Use `geom_line()`

• **Input** to `ggplot()`
  • `data = yearly_counts`
  • `x = year`
  • `y = n` (# observations)

```r
ggplot(data = yearly_counts, 
aes(x = year,
    y = n)) +
geom_line()
```
group by species

• Create a line for each species id

• Input:
  • `aes()` argument: `group`

• But what line is what species?

```r
ggplot(yearly_counts,
       aes(x = year, y = n,
           group = species_id)) +
geom_line()
```
Create a legend

• Add color to autogenerate a legend
• \texttt{aes()} argument \texttt{color}
• Can use \texttt{group\_by} and \texttt{color} separately

\begin{verbatim}
ggplot(yearly_counts, 
aes(x = year, y = n, 
    color = species_id)) + 
geom_line()
\end{verbatim}
Save ggplots to a variable

- Saves ggplot as a list

- To render a saved plot, send the name of the plot to the console

- Can add to the plot or use it to export the plot to a file

```r
lineplot <- ggplot(data = yearly_counts, aes(x = year, 
    y = n)) + 
    geom_line(aes(color = species_id))
```
Exercise

• Using colors, create a plot that depicts how the average weight of each species changes through the years.

• Save it to an object called mean_weight_sp

• Hint: reshape the data using the following code

```r
yearly_weight <- surveys_complete %>%
  group_by(year, species_id) %>%
  summarize(avg_weight = mean(weight))
```
Creating publication quality graphs

lineplot

lineplot + theme_xx()
Applying pre-made themes

- Premade
  - Ex: `theme_bw()`
  - List of themes in help (`?theme`)

- Apply theme using `+` operator

[Link to ggplot2 documentation](https://ggplot2.tidyverse.org/reference/ggtheme.html)
Customize axis labels

- **Default**: variable names as axis labels

- **labs()**: specify labels
  - **title**: plot title
  - **x**: x axis label
  - **y**: y axis label

- Provides more descriptive text

```r
line_bw_lab <- line_bw + labs(title = 'Observed Species in time',
                              x = 'Year of observation',
                              y = 'Count')
```

Observed species in time

![Graph showing observed species over time with customizable axis labels](image-url)
Customize font size

- **theme()** - fine adjustments to the theme
  - Used with theme elements
  - Ex: text size

- **element_text()**
  - **size** - text size
  - **family** - font

- **?margin** for more ggplot theme elements

```r
line_bw_labs_font <- line_bw_labs + theme(text=element_text(size=16, family="Arial"))
```

**Observed species in time**
Create your own theme

- Save your theme to a list object
- Can be applied to any plot
- Make sure to include theme elements only, not geoms etc.

```r
arial_theme <- theme_bw() +
  theme(text = element_text(size=16,
                             family="Arial"))
```
Apply your theme

ggplot(surveys_complete,
aes(x = species_id,
    y = hindfoot_length)) +
geom_boxplot() +
arial_theme
Arranging plots

• No function in ggplot2

• Use the gridExtra library

• `grid.arrange()`
  • Plots to arrange
  • `Ncols` - number of columns
  • `Widths` -

```r
grid.arrange(spp_weight_boxplot, spp_count_plot,
              ncol = 2,
              widths = c(4, 6))
```
Save your plot

• Plot window > Export

• `ggsave()`: saves plot to a file
  • `filename` - name of new file
  • `plot` - plot to be saved
  • `device` - output format
  • `units` - units for output dimensions
  • `width` - width of output
  • `height` - height of output

```r
#last plot by default
plot = line_bw_labs_font,
device = "png",    #default
units = "in",      #default
width = 15,
height = 10)
```
Final Exercise:

Take five minutes to:

• improve one of the plots generated in this exercise or
• create a beautiful graph of your own.
• Cheatsheet:  

Here are some ideas:

• Try changing line thickness
• Change the name of the legend? What about its labels?
• Try using a different color palette:  
  http://www.cookbook-r.com/Graphs/Colors_(ggplot2)/
Faceting

- Split one plot into many based on a factor

- `facet_XXX()` - specify layout of panels

- `~` - formula
  - `x ~ y`
Facet wrap

- Facet on one variable

- `facet_wrap()` - wraps panels horizontally, Fits to the screen

- Argument: `~ y`

```r
ggplot(data = yearly_counts,
       aes(x = year, y = n)) +
geom_line() +
facet_wrap(~ species_id)
```
**Color by sex**

- Adds dimensionality to the plot

```r
ggplot(data = yearly_sex_counts, 
      aes(x = year, 
          y = n, 
          color = sex)) + 
  geom_line() + 
  facet_wrap(~ species_id)
```
Facet grid

- Specify your layout

```r
ggplot(data = yearly_sex_weight, 
      aes(x = year, 
          y = avg_weight, 
          color = species_id)) +
  geom_line() +
  facet_grid(sex ~ .)
```
Facet grid

- Change orientation

```r
ggplot(data = yearly_sex_weight,
       aes(x = year,
           y = avg_weight,
           color = species_id)) +
geom_line() +
facet_grid(.~sex)
```
Separate into plots instead of color

```r
ggplot(data = yearly_sex_weight,
    aes(x = year,
        y = avg_weight,
        color = species_id)) +
geom_line() +
facet_grid(sex ~ species_id)
```
Faceting exercise

Using **facets** and **themes**, create a plot that depicts how the **average weight** of each **species** changes through the **years**.
Need help?

• Email: tobin.magle@wisc.edu

• UW-Research Data Services: http://researchdata.wisc.edu/

• Data Carpentry lesson

• ggplot2 Cheat Sheet:
Take the survey!