Economics of Financial Crisis
Yaprak Tavman
Best Practices for Data Visualization

Visualizing with ggplot2

ggplot2 is an open-source data visualization package meant for use with the programming language R. The software is a part of the tidyverse package for R, and is free to download and install on personal computers.

R has a wide array of functions that each perform specific tasks. Where data visualization is concerned, **ggplot() is the specific function** used in RStudio to turn data into compelling and persuasive visualizations.

The ggplot() function gives users the ability to visualize their data using line graphs, bar charts, scatterplots, geographic maps, heatmaps, histograms, and more.

Questions to consider:

1. **In what ways have you manipulated your data?** Data visualizations are subjective and open to manipulation just as much as any other data. It's important to account for how human error and human judgment impact data outcomes. In short, just because a data visualization exists, doesn't mean that it's telling the whole truth.
2. **What aspects of your dataset are best suited to visualization?** Consider what information is most necessary for your audience. If an unfamiliar audience first encounters your project, what visuals would help them familiarize themselves quickly with the data? What visualizations can draw out the conclusions you are trying to support with your research?
3. **What type of plot is best suited to visualize your data?** ggplot2 has a host of available visuals and chart options. Choose the plot that works best for your project.

Best practices for data visualization

- Some good **general principles f**or R include:
    - keeping an eye on file paths
    - remembering to check the working directory
    - and verifying which project space you're working in
- **Clean your data.** Good data visuals start from well organized data (usually in tabular format). Be sure to choose clear, concise names for your columns that are friendly for

Developed by: Tieanna Graphenreed, DITI Fellow and Colleen Nugent, DITI Fellow
Taught by: Tieanna Graphenreed, DITI Fellow and Colleen Nugent, DITI Fellow
Questions? Contact us: nulab@northeastern.edu

exportation and easy to remember. This step will also serve you well as you manipulate your data in R/RStudio.

- **Make an accessibility-forward design.** Consider how font sizes, typeface, color choice, and other design elements might impact your user. This post from *Towards Data Science* links to three helpful readings on accessibility; *TDS* also has an "incomplete guide" with some best practices.

Useful terms and vocabulary

- **Function**: functions in R are code that performs a specific task.
- **Argument**: the inputs provided to functions. Functions can have multiple arguments, or none at all.
- **Aesthetics:** descriptions of how variables are mapped to visual properties or, "aesthetics"; uses aes() function. Used to dictate the appearance of a geom.
- **Aesthetic Mapping**: a set of aesthetic mappings. Usually a sequence of aesthetic-variable pairs—e.g., aes(x= , y=, colour=)
- **Geom**: the function used to select the type of geometric object for data visualization. Geoms will always have aesthetics as parameters to control how the visualization is displayed. ggplot2 supplies numerous geom functions for almost every graphing need.
- **Layer**: in ggplot2 layers are used to create graphs. Layers help define geometry, define and set scales, change styles, and more.
- **Scale**: scales control the details of how data values are translated to visual properties. You can use scale functions to tweak the appearance of x/y axes, legend keys, limits, and even color.
- **Facet**: an alternative way to set aesthetics and display variables.
- **Coordinates**: a system that determines how different x and y elements combine to position elements on the graph/plot.

Using ggplot2 for data visualization

To get started, you need to ensure ggplot2 is installed and loaded into your session.

1. Open up RStudio.
2. ggplot2 is an additional package not built into Rstudio.
    a. To download and install the package type install.packages("ggplot2") into the console.
    b. If you already have the ggplot2 package installed, you'll receive a message like this in the console.

```
> installed.packages("ggplot2")
     Package LibPath Version Priority Depends Imports LinkingTo Suggests Enhances
     License License_is_FOSS License_restricts_use OS_type Archs MD5sum
     NeedsCompilation Built
```

3. Load the package into your session by typing library("ggplot2") into the console.
4. ggplot2 is now installed on your device and active in your session.
5. *Remember* that once you have installed packages the first time, you do not need to do so again. **You do, however, need to *load* the packages every session** with the library() function. If you are doing something that requires a new package, then you will need to install and load it first
    a. To install a new package, use the install.packages() function following the same format as above. For this exercise you'll type: install.packages("ggplot2")
    b. To load the package use the function library(" ") and fill in the name of the specific package. For this exercise, you'll type library("ggplot2")
    c. You can also see all the packages you have installed, and even install new packages, in the "Packages" tab of the File pane (bottom right).

Walkthrough Instructions:

Understanding chains

ggplot2 operates via "chains," or a series of functions and arguments that connect a data source to a graph. Chains in ggplot2 begin with the **ggplot() function** and chains are linked together primarily by **aesthetics *aes()*** and **layers of geometric objects *geom_()*;** *secondary elements* like scales, coordinates, facets, positions, and other functions help further adjust the appearance of the graph. All these functions are added within the overall framework of the ggplot() function in order to complete a graph using the ggplot2 package.

Within R, other functions can be nested within larger functions to achieve certain goals (like data visualization). We explain the nesting logic of ggplot() chains in more detail below, but you can read more about the overall anatomy of a ggplot chain here.

Getting Started:

To begin creating a plot you must use the ggplot( ) function; then, you can add elements as needed. You can specify the aesthetic components with the aes() function and begin identifying information from a data source to map onto the plot you are creating.

Once you've set up the ggplot() function and aesthetic mapping, you can **add** the geometric object layer (graph/chart type) using a geom_() function. The type of geom function you choose will depend on what chart/graph type you desire. For example, if you wanted to make a histogram you would use the geom_histogram() function. Similarly, if you want to make a point plot, you would use

the geom_point() function. ggplot2 supplies numerous geoms to meet almost every graphing need. You can set additional aesthetic components for the geometric objects as well as the overall ggplot function. Set scales, coordinates, and facets as needed to further alter the appearance of your graph.

This demo will give you instructions on how to use RStudio and ggplot2 to create two graphs: a line graph *geom_line()* and a scatter plot *geom_point()*. Before you start, make sure you still have the ggplot2 package loaded in your session. If you don't, reload the package using the function: library("ggplot2") .

---

1. **Selecting a data source:** First, download your datasource.
    a. Save the workshop demo folder somewhere easily accessible on your computer.
        i. This contains everything you will need for the ggplot2 demo, including: a walkthrough RMD file, the sample dataset (trial-dataset.csv), and an RStudio project.
        ii. When you work with your own data, you should download the data you'll be using and save it in a folder with your project. Make sure you're familiar with the data—you may need to make changes to the data, which you can do in R or in other software for working with tabular data.
    b. Select/Import the file into RStudio.
        i. Right- or control-click on the file in the data pane in RStudio and click import dataset **or**
        ii. Type read.csv and your file path: read.csv("your file path here")
    c. To make this easier to work with, you can create a variable called "data" using the following code: data = read.csv("your file path here")
    d. To start the process of visualization in ggplot, you must select this data source within the ggplot function using : ggplot (data,) [Go to instruction #2 to see what else you can fill in]
2. **Setting an aesthetic mapping:** Set your axes and other values based on the type of plot you want to create.
    a. For both a line chart and a scatterplot, you would assign both an x axis and y axis, separated by commas
        i. For example, ggplot(data, aes(x=TIME, y=Value))
        ii. Make sure the function uses the same column titles as those in your dataset. If the capitalization isn't the same, RStudio won't pull the data.
3. **Choosing a "geometry" (or plot type):** Choose the visual best fit for your data. Consider what information might be most useful for your audience and the message you want to convey.
    a. For this exercise we will experiment with line graphs and scatter plots.

---

   i. Line graph: geom_line()
   ii. Scatter plot: geom_point()

4. **Setting scales:** Choose what aspect of your data will be visualized by the geometry.
   a. **For a line graph:** ggplot(data, aes(x=TIME, y=Value) + geom_line(aes(linetype=LOCATION, color=LOCATION))
   b. **For a scatter plot**: ggplot(data, aes(x=TIME, y=Value, color=LOCATION, shape=LOCATION)) + geom_point()
      i. You can choose to add a regression line to your scatter plot by adding the function geom_smooth()
      ii. Within the geom_point() function you can also specialize the size, shape, and color of the points.
         1. Typically, the ggplot2 package will automatically make this distinction for you; however, if you want more control you would use two additional arguments:
            a. To adjust colors: use the argument scale_color_manual and select the appropriate color. See this resource from Columbia University for the full shade range in RStudio.
            b. To adjust the shape of your plot points: use the argument scale_shape_manual. See this resource from the *R Graphics Cookbook* to learn more about the numeric categorization of shapes in RStudio.
5. **OPTIONAL: Design Choices (Themes, Labels, etc.):** These choices primarily affect the appearance of your chart for viewers, as well as their understanding of the material. For example, you might choose to add a label to your visual that tells viewers about the content of the graph/chart. This helps eliminate ambiguity
   a. **Setting a scale:** To add or change a label you would use the scale function: labs(). Typically, this will fall between the ggplot() function and the geom_YOURCHART() function.
      i. Here's an example: ggplot(data, aes(x=TIME, y=Value, color=LOCATION, shape=LOCATION)) **+ labs(x="YEAR", y="Gross Domestic Product")** + geom_point(size=5,shape=23, color="red") + geom_smooth(method = lm, se=FALSE)
   b. **For all graphs:** The labels available to be changed are: the x axis (x="x label"), the y axis (y="y label"), the title (title="title"), the subtitle (subtitle="subtitle"), and the caption (caption="="caption").
      i. In some cases, the y label may not be applicable. This depends on your choice of visual.

Developed by: Tieanna Graphenreed, DITI Fellow and Colleen Nugent, DITI Fellow
Taught by: Tieanna Graphenreed, DITI Fellow and Colleen Nugent, DITI Fellow
Questions? Contact us: nulab@northeastern.edu

Resources for RStudio, ggplot2, and data visualization:

"Colors in R." Columbia University. http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf

Prabhakaran, Selva. "The Complete ggplot2 Tutorial - Part 1 | Introduction To ggplot2 (Full R Code)" *r-statistcs.co*

Schmidt, Ben. "Visualizing Data: the basics"

Wickham, Hadley et. al. "ggplot2 part of the tidyverse: Reference"

"5.3 Using Different Point Shapes." R Graphics Cookbook: https://r-graphics.org/recipe-scatter-shapes

Developed by: Tieanna Graphenreed, DITI Fellow and Colleen Nugent, DITI Fellow
Taught by: Tieanna Graphenreed, DITI Fellow and Colleen Nugent, DITI Fellow
Questions? Contact us: nulab@northeastern.edu