# Data Visualization with ggplot2

## Cheat Sheet

### Basics

**ggplot2** is based on the grammar of graphics, the idea that you can build every graph from the same few components: a data set, a set of geoms—visual marks that represent data points, and a coordinate system.

To display data values, map variables in the data set to aesthetic mappings, and `x` and `y` locations.

### Geoms

- **One Variable**
  - **Continuous**
    - `a <- ggplot(mpg, aes(hwy))`
    - `a + geom_area(stat = "bin") x, y, alpha, color, fill, linetype, size`  
    - `b + geom_area(aes(y = ..density..), stat = "bin") x, y, alpha, color, fill, linetype, size`  
    - `a + geom_density(kernel = "gaussian") x, y, alpha, color, fill, linetype, size, weight`  
    - `a + geom_dotplot() x, y, alpha, color, fill`  
    - `a + geom_freqpoly() x, y, alpha, color, linetype, size, weight`  
    - `a + geom_histogram(binwidth = 5) x, y, alpha, color, fill, linetype, size, weight`  
    - `Discrete`  
      - `b <- ggplot(mpg, aes(fuel)) x, alpha, color, fill, linetype, size, weight`  
  - **Two Variables**
    - `Continuous X, Continuous Y`  
      - `f <- ggplot(mpg, aes(cty, hwy))`  
      - `f + geom_blank() x, y, alpha, color, fill, linetype, size`  
      - `f + geom_jitter() x, y, alpha, color, fill, shape, size`  
      - `f + geom_point() x, y, alpha, color, fill, linetype, size`  
      - `f + geom_quantile() x, y, alpha, color, fill, linetype, size`  
      - `f + geom_rug(sides = "bl") x, y, alpha, color, linetype, size, weight`  
      - `f + geom_smooth(model = lm) x, y, alpha, color, fill, linetype, size, weight`  
      - `f + geom_text(aes(label = cty)) x, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust`  
    - **Discrete X, Continuous Y**  
      - `g <- ggplot(mpg, aes(class, hwy))`  
      - `g + geom_bar(stat = "identity") x, y, alpha, color, fill, linetype, size`  
      - `g + geom_boxplot() x, y, alpha, color, fill, linetype, shape, size, weight`  
      - `g + geom_errorbar() x, y, alpha, color, fill, linetype, shape, size, weight`  
      - `g + geom_hline() x, y, alpha, color, fill, linetype, size, weight`  
      - `g + geom_smooth() x, y, alpha, color, fill, linetype, size, weight`  
      - `Discrete X, Discrete Y`  
        - `h <- ggplot(movies, aes(year, rating)) x, y, alpha, color, fill, linetype, size`  
        - `h + geom_point() x, y, alpha, color, fill, linetype, size`  
      - **Continuous Bivariate Distribution**  
        - `i <- ggplot(movies, aes(year, rating))`  
        - `i + geom_bin2d(binwidth = c(5, 0.5)) xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size`  
        - `i + geom_density2d() x, y, alpha, color, fill, linetype, size`  
        - `i + geom_hex() x, y, alpha, color, fill size`  
    - **Continuous Function**
      - `j <- ggplot(economics, aes(date, unemploy))`  
      - `j + geom_area() x, y, alpha, color, fill, linetype, size`  
      - `j + geom_line() x, y, alpha, color, linetype, size`  
    - **Visualizing error**
      - `k <- ggplot(diamonds, aes(cut, color))`  
      - `k + geom_crossbar(fatten = 2) x, y, ymax, ymin, alpha, color, fill, linetype, size`  
      - `k + geom_errorbar() x, y, ymax, ymin, alpha, color, linetype, size, width (also geom_errorbarh())`  
      - `k + geom_linerange() x, y, ymax, ymin, alpha, color, linetype, size, shape, size`  
      - `k + geom_pointrange() x, y, ymax, ymin, alpha, color, fill, shape, size`  
    - **Maps**
      - `l <- ggplot(data, aes(fill = murder))`  
      - `l + geom_map(data, aes(fill = murder))`  
    - **Three Variables**
      - `m <- ggplot(seals, aes(x = long, y = lat))`  
      - `m + geom_contour(aes(z = z)) x, y, z, alpha, color, linetype, size, weight`  
      - `m + geom_raster(aes(fill = z), hjust=0.5, vjust=0.5, interpolate=FALSE) x, y, alpha, fill`  
      - `m + geom_tile(aes(fill = z)) x, y, alpha, color, fill, linetype, size`  

### Graphical Primitives

- `c <- ggplot(map, aes(long, lat))`  
- `d <- ggplot(economics, aes(date, unemploy))`  
- `e <- ggplot(seals, aes(x = long, y = lat))`  
- `i <- ggplot(movies, aes(year, rating))`  
- `j <- ggplot(diamonds, aes(cut, color))`  
- `k <- ggplot(diamonds, aes(xmin = long, ymin = lat, xmax = long + delta_long,ymax = ymin + delta_lat))`  
  
### Additional Elements

Add a new layer to a plot with a `geom_*()` or `stat_*()` function. Each provides a geom, a set of aesthetic mappings, and a default stat and position adjustment.

**last_plot()**

Returns the last plot.

**ggsave("plot.png", width = 5, height = 5)**

Saves last plot as 5’ x 5’ file named “plot.png” in working directory. Matches file type to file extension.

---

This cheat sheet is a summary of ggplot2 functions, which is a powerful tool for creating data visualizations. It includes examples of different types of plots, such as `geom_area()`, `geom_histogram()`, and `geom_point()`. Each function has its own parameters that can be adjusted to customize the appearance of the plot. The cheat sheet also includes a section on graphical primitives, which are the basic building blocks of ggplot2. These primitives can be combined to create more complex visualizations.

The cheat sheet is organized into sections based on the type of data visualization, such as one variable, two variables, and continuous bivariate distribution. It also includes sections on continuous function, visualizing error, maps, and three variables.

The cheat sheet is a useful resource for anyone who wants to create data visualizations using ggplot2. It provides a quick reference for the different functions and parameters available, as well as examples of how they can be used to create different types of plots. By using the cheat sheet, users can quickly find the function they need and understand how to use it to create the visualization they want.
Some plots visualize a transformation of the original data set. Use a `stat` to choose a common transformation to visualize, e.g. `a + geom_bar(stat = "bin")`

Each stat creates additional variables to map aesthetics to. These variables use a common...