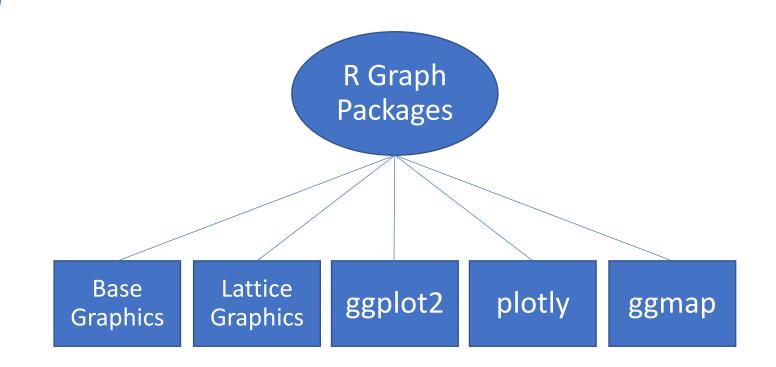


- Review EDA II Wrangling
- Foundation of many graphic applications
 - Grammar of Graphics
- Focus on teaching the underlying theory of ggplot2
- How is Grammar of Graphics reflected in the API
- Chart types
- Dealing with colors

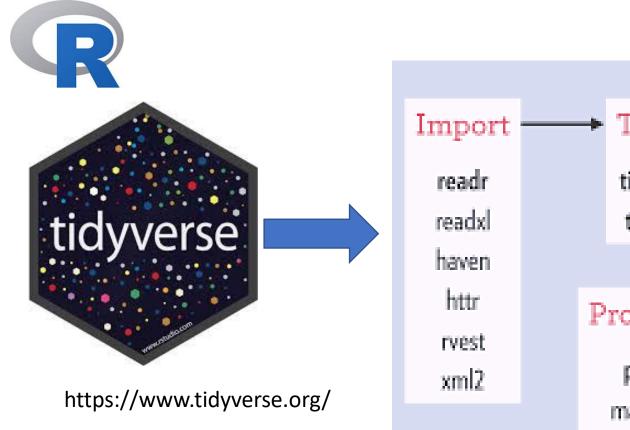


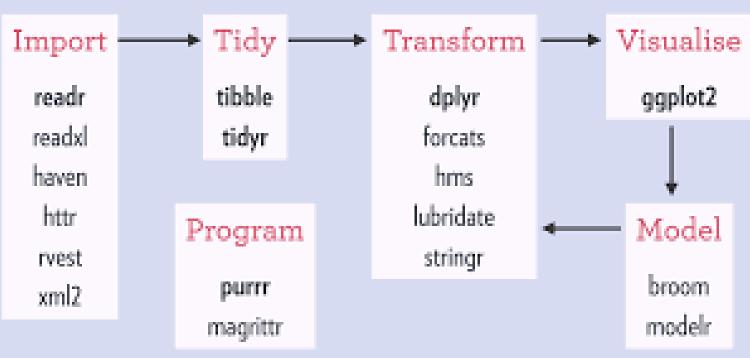


Data visualization in R can be performed in the following ways:







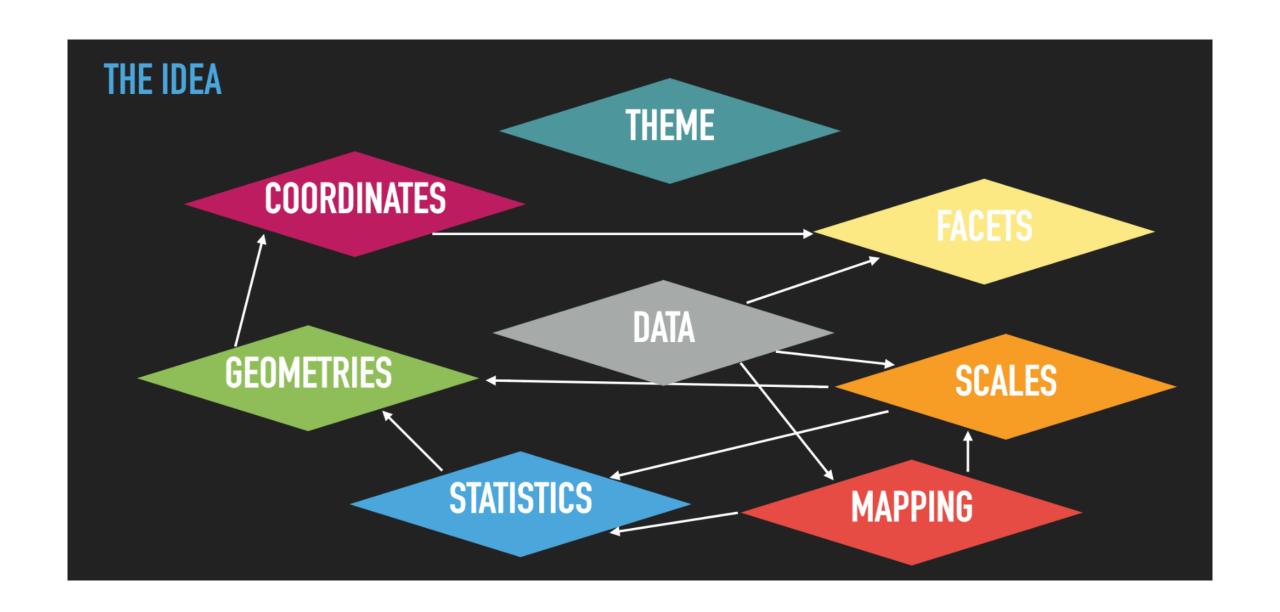


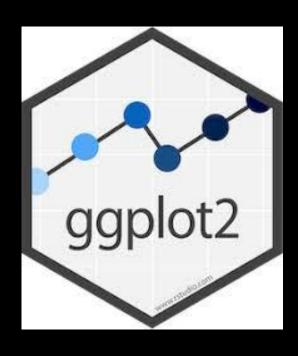
install.packages("tidyverse")

library("tidyverse")



ggplot2 is one of the most elegant and most versatile. ggplot2 implements the grammar of graphics, a coherent system for describing and building graphs. With ggplot2, you can do more faster by learning one system and applying it in many places.

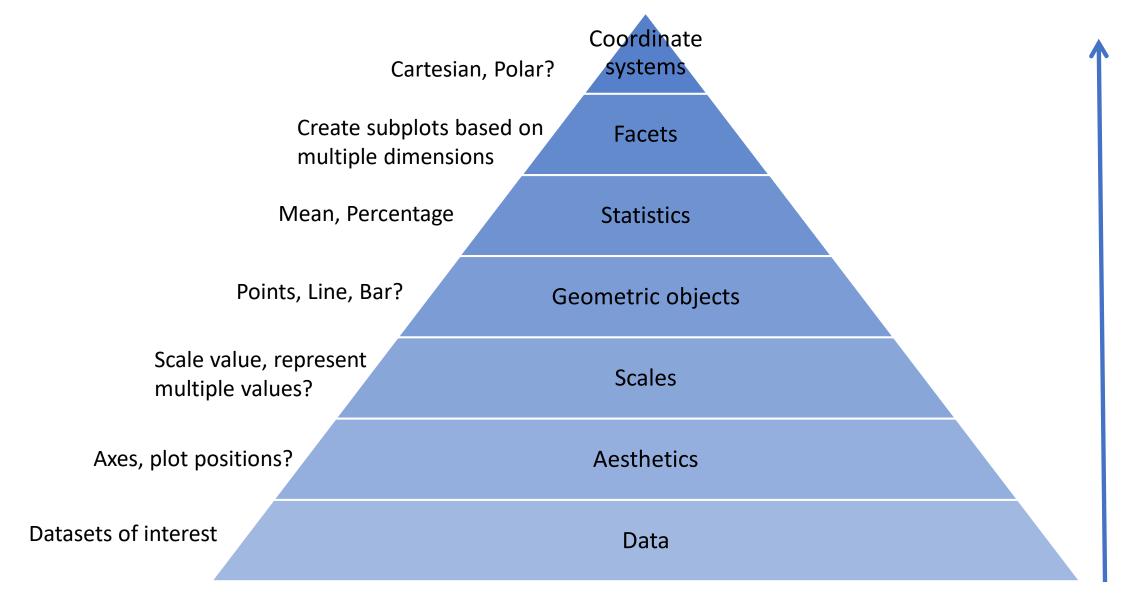




THEME COORDINATES FACETS GEOMETRIES SCALES STATISTICS MAPPING DATA



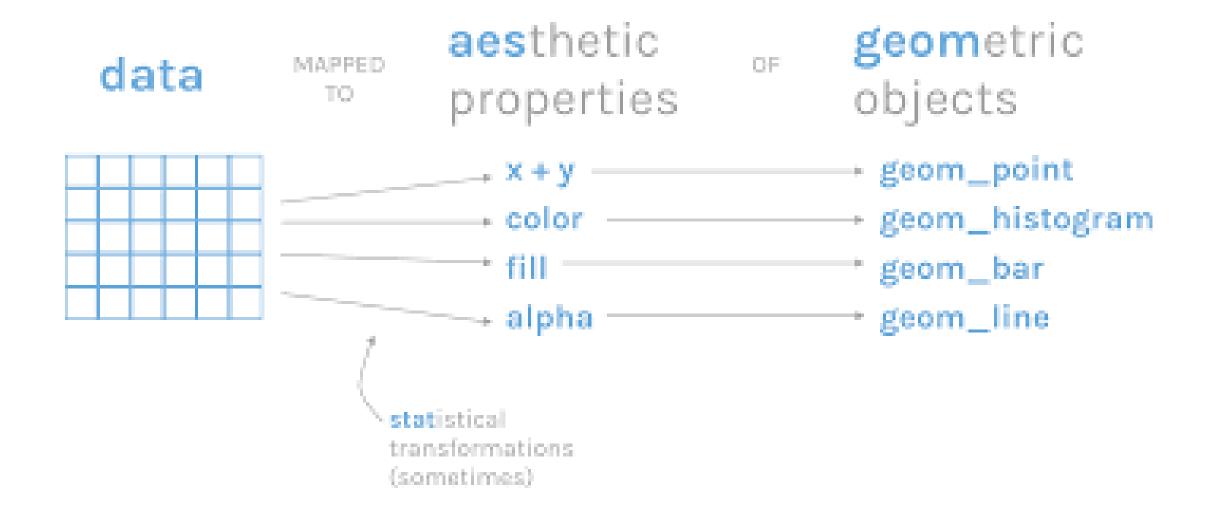
Components of the Grammar of Graphics



- 1. Data are the values represented in the visualization.
- 2. **Aesthetic mappings** are directions for how data are mapped in a plot in a way that we can perceive. Aesthetic mappings include linking variables to the x-position, y-position, color, shape, and size.
- 3. **Geometric objects** are representations of the data, including points, lines, and polygons.
- 4. **Scales** turn data values, which are quantitative or categorical, into aesthetic values.
- 5. **Coordinate systems** map scaled geometric objects to the position of objects on the plane of a plot. The two most popular coordinate systems are the Cartesian coordinate system and the polar coordinate system.
- 6. Facets (optional) break data into meaningful subsets. Split data into multiple panels.
- 7. **Statistical transformations (optional)** transform the data, typically through summary statistics and functions, before aesthetic mapping.
- 8. **Theme** controls the visual style of plot with font types, font sizes, background colors, margins, and positioning.

Plot Basics

ggplot()	Create a new ggplot
aes()	Construct aesthetic
	mappings
`+`(<gg>) `%+%`</gg>	Add components to a
	plot
ggsave()	Save a ggplot (or other
	grid object) with
	sensible defaults
qplot() quickplot	Quick plot
()	



Data visualization with ggplot2:: cheat sheet



continuous bivariate distribution

i <- ggplot(economics, aes)date, unemploy()

x, y, alpha, color, group, linetype, size

x, y, alpha, color, group, linetype, size

of <- data frame(grp = 1("A", "B"), fit = 4.5, se = 1.20) <- ggplot(off, seo(grp, fit, ymin = fit - se, ymax = fit + se()

[+ geom_errorban]) - x, ymax, ymin,

| + | 1 * geom_pointrange() - x, y, ymin, ymax,

data <- data frame/murder = USArrestsSMurder,

[* geom_linerange()

alpha, color, group, linetype, size, width Also geem, errorbarb().

x, ymin, ymax, alpha, color, group, tinetype, size

ymin, alpha, color, fill, group, linetype, size

i+geom_step(direction = "hu")

h c- ggplet(diamonds, aes)carat, price))

continuous function

visualizing error

ggplot2 is based on the grammar of graphics, the idea that you can build every graph from the same components: a data set, a coordinate system.



properties of the grow (seethetics) like size, color, and a and y locations.



Complete the template below to build a graph.

projet (data = COVD) = COLOR PROCESSOR Companies - and Chi start + CHILD, position + CROWNS COORDINATE FUNCTION CHACET FUNCTIONS CHEALE FUNCTIONS

ggplot()data = mpg, aes(x = cty, y = hwyl) (liegins a plot that you finish by adding layers to. Additione groom Rangtion per laver.

ggsawe("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.

Aes common aesthetic values.

color and fill - string ("red", "#RRCOBB") Snetype - integer or string (0 = "blank", 1 = "solid", 1 = "dashed", 3 = "dotted", 4 = "dotdash", 5 = "longdash",

Ensend - string ("round", "butt", or "square") Singlein-string ("round", "mitre", or "bevel")

size-integer (line width in mm) ! shape - integer/shape name or



Geoms Use a grown function to represent a a points, use the geom's aesthetic properties to represent variables.

TWO VARIABLES

both continuous

e <- agplotimpg, sesicty, hwyl)

e + grom_quantite()

one discrete, one continuous

e + grom_rug(sides = "ht")

e + geom_smooth(method = lm)

f+ peam_cal() x, y, alpha, color, fill, group, linetype, size

x, y, alpha, color, fill, group

8 # f + green_violin(scale = "area")

g <- ggstot/diamends, sestout, color)

both discrete

a @ g + geom, count()

THREE VARIABLES

e + geom_label(aestlabet = ctpl, madge_x = I.,

x, y, alpha, color, group, linetype, size, weight

x, y, alpha, color, fill, group, linetype, size, weight

• + geom_text[seulabel = ctyl, nudge_s = 1,

nudge_y = 1) - x, y, label, sipha, angle, color, family, fontface, hjust, lineheight, size, vjust

x, y, lower, middle, upper, ymax, ymin, alpha.

octor, fill, group, linetype, shape, size, weight

f + green_desplot(binaris = "y", stackdir = "center")

v. alpha, color, fill, group, linetype, size, weight

nudge_y = 13 - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, viust

e + geom_point() x, y, sloha, color, fill, shape, size, stroke

GRAPHICAL PRIMITIVES

a <- ggplot(economics, assidate, unemploy() b <- ggplot(seals, seals = long, y = lat()

a + geom_blank() and a + expand_limits()

long + 11, curvature = 13 - x, send, y, yend, alpha, angle, color, curvature, linetype, sine

a . geem_path(incend = "butt". x, y, alpha, color, group, linetype, size

a • geom_polygon(ses(sipha = 50)) - x, y, sipha, color, fill, group, subgroup, linetype, size b + geom_rect(assismin = long, ymin = lat, smax = long + 1, ymax = lat + 1)) - xmax, emin,

ymax, ymin, alpha, color, fill, linetupe, size a + geom ribbonizesiymin r unemploy : 100. ymax - unemploy + 5000) - s, ymax, ymin, alpha, color, fill, group, linetype, size

LINE SEGMENTS.

on aesthetics: x, y, alpha, color, livetype, size

b + geom_abline(sesCintercept = 0, slope + 1)} b + gages Milnelpen/viotercept = latit b + peem_vilee(sestkintercept = long()

b * geom_segment(ses);end * lat * 1, send * long * 1() b * geom_spoke(ses)angle * 1:1155, radius * 10

ONE VARIABLE continuous

c <- ggplotimpg, senihwyl); c2 <- ggplotimog)

c * geom_area(stat = "bin") x, y, sipha, color, fill, linetype, size

c * geom_density(kernel + *goussian*) n, y, alpha, color, fill, group, linetype, size, weight

c+peem_detplat()

c + geom frespoly() s, y, alpha, color, group, licetype, size

c + geom_histogram(binwidth + 5) x, y, sipha, color, fill, linetype, size, weight

if v- ggplot/mog, aes/f()

alpha, color, fill, limitype, size, weight

1+ geom_contour_filled(ses(fill+z)) x, y, alpha, color, fill, group, linetype, size, subgroup

x, y, alpha, color, fill, shape, size, stroke

y a + groom, jitter(height + 7, width + 2)

x, y, alpha, color, fill, shape, size

I+ peem contourient(r = ri)

x, y, z, alpha, color, group, linetype, size, weight

seels(c) <- with(seels, sqrt)(elta_long*2 + delta_lat*2()) | <- gaplot(seels, seellong, lat() I - geom_raster(ses/58 - z), Nust + 0.5.

k • geom_map(sen(map_id = state), map = map).

+ expand_timits(x = mapSlong, y = mapSlat) map_id_alpha, color; fill_linetype, size

mus or man distrat*state*)

alpha, color, fill, linetype, size, width

t = geom_tile(ses(fill = z))







Stats An alternative way to build a layer. A stat builds new variables to plot (e.g., count, prop). =

geom coordinate plot

h • geom_bla2d(binwidth + c(0.25, 500)) x, y, alpha, color, fill, linetype, size, weight Viscolize a stat by changing the default stat of a goom function, geom_bar(state* count*) or by using a state. function, stat_count(grown"bar"), which calls a defaul h • geom_density_2d() x, y, alpha, color, group, linetype, size grow to make a layer (equivalent to a grow function). Use _name_ syntax to map stat variables to sesthetics.

data stat



Continue Continue Commen I - stat_density_2d(sm[III - .level.), * doors , hopking,)

e + stat_bin(binwidth + 1, boundary + 10) e + stat_count(width + 1) x, y | _count_, _prop. c = stat_density(adjust = 1, kernet = "gaussian")

e + stat_bin_2d(bins + 30, drop + 1)

e + stat bin healbins = 30) x, y, 600 | .count., .density. e = stat_density_2d(contour = TRUE, n = 100) e, y, color, size | _level.

++stat_ellipse(level = 0.55, segments = 51, type = "t")

1 + stat_contour(next) = ([] x, y, x, order [.level. (- stat, nummary hardwell - z), bins - 30, fun - mad x, y, x, fill) unidos.

I = stat_summary_2d(senit = s), (sins = 30, fun = mean) 8, y, r, fill | _solver.

f + stat_basplet(cod + 1.5) x, y | .lease , models , upper , width , .ymin , .ymax f - stat_ydensity(kernel = "gaussian", scale = "area") #, y | density scaled count n violineable, saleh

e+stat_endf(n+40) x,y | x, y, e * stat_quantile(quantiles * c(0.1, 0.5), formula * y = log(x), method * 'rq') x, y | _quantile.

a e stat amouth) method = "im", formula = y - x, se = T.

ggplot() = xlim(-5, 5) = stat_function(lun = dnorm, 20, geom = "point") x |y.

ggplot() + stat_oq(ses(sample + 1:100)). x, y, sample | _sample__, theoretical_

e + stat_oum() x, y, size |prop... e e stat summary(fun data = "mean of boot")

h + stat_nummary_bin(fun + "meon", grom + "bor") a + stat_identity()

p + scale_shape() + scale_size()

 p + scale_radius(range + c(), 6) p + scale_size_area(mix_size = 6)

Scales overide defaults with scales package. Coordinate Systems

Scales man data values to the visual values of an sesthetic. To change a mapping, add a new scale.



GENERAL PURPOSE SCALES

scale_*_coetinoous() - Map cont' values to visual ones. acale_*_discrete() - Map discrete values to visual ones. arate * himself] . Han continuous values to discrete him

ecale, *_lidentity() - Use data values as visual ones.

scale_"_dene(dute_labels = "Non/Not"); date_breaks = "2 weeks") - Treat data values as dates scale * datetime() - Treat data values as date times. Same as scale * date(). See hitrotime for label format

X & Y LOCATION SCALES

Use with x or y anothetics to shown here! scale_x_log10() - Plot x on log10 scale. scale_x_reverse() - Reverse the direction of the x axis. scale_x_sgrt() - Plot x on square root scale.

COLOR AND FILL SCALES (DISCRETE)

n + scale_fill_brewer(points = "libes"). For palette choices: 9Colorforwer: display brewer.all() end = 0.0, na value = "red")

COLOR AND FILL SCALES (CONTINUOUS)

a <- c + geom_dotplot(ses(fill + .x.))

e + scale fill distiller(palette + "Bloes")

a - scale_fill_gradient()own"red", high="yellow") e + scale_fill_gradient2(low = "red", high = "blue",

a - scale_fili_gradients(colors - topo.colors(t))
 Also: rainbow(), heat.colors(), ternain.colors(), (m.colors(), RColorBrewer;brewer.pst()

SHAPE AND SIZE SCALES prine region, point(ses)shape r ft, size r cylit

p + scale_shape_manual(values = c0x10)

y 4 - d + geom_bar() r • coord_cartesian(xlim • c(0, s() - xlim, ylim. The default cartesian coordinate system.

r + coord_fixed(ratio + 1/2) ratio, xlim, ylim - Cartesian coordinates with fixed aspect ratio between x and y units.

appliet(mpg, senily = f() + geom_bar() x and y aesthetic mappings.

Position Adjustments

Position adjustments determine how to arrange grown

+ + geom_bar(position = "dodge")

e + peom_point(position + "jitter")

e + geom. (abel/position = "nudge")

each element to avoid overploiting.

s * geom, bar(position = "stack")

Each position adjustment can be recest as a function

with manual width and height arguments:

Themes

retherne built

r + thome_grap()

inny background

r + thome_dark()

r + theme() Customize aspects of the theme such

with grid lines.

+ + geom_bar(position + "fill")

that would otherwise occupy the same space.

sic-applicationed sessit, fill individual

Stack elements on top of one another, normalize height.

r + coord_polar(theta > "x", direction=1) theta, start, direction - Polar coordinates

r + coord, transity = "sort") - x, y, slim, ylim. Transformed cartesian coordinates. Set strano and ytrans to the name of a window function.

Ser scales to let asis limits your across facets. we count quickmant m + coord_map(projection = "ortho", orientation = c(41, -74, 00 - projection, xlim, ylim t + facet_grid(rows = vars(drv), cols = vars(ft), Map projections from the mappets package (mercator idefault), asequatarea, lagrange, etc.).

s and y axis limits adjust to individual facets:
"free_x" - y axis limits adjust. "free_y" - y axis limits adjust

Set labeller to adjust facet label.

Faceting

Facets divide a plot into subplots based on the

values of one or more

t + facet_grid(cels = vars(f), labeller = label_both) Ar Ar Ar Ar

t <- ggplot(mpg, set(xty, hwy() + geom, point))

| | | | t + facet_grid(cols = wars(f())

t + facet_grid(rows = vars(year))

t + facet wrapivarsiff()

t + facet_grid/rows = vars(year), cels = vars(f())

t - facet_grid(rows - varsiff), labeller = label becombisions = (fill) n' n' n' n' n'

ggplot2

Labels and Legends

Clor label) to label the elements of your plot. t + labs(x = "New x axis label", y = "New y axis label", subtitle = "Add a subtitle below title" caption = "Add a caption below plot" all + "Add all tent to the plot"; all > "New out to legend side"]

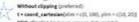
t = amoutabe(geom = "bext"; a = 6, y = 9, label = "W"); Places a grom with manually selected anotherics

g • guidents = guide, anion dodge = 2]] Avoid crowded or overlapping labels with guide, anion dodge or angle! n • guides(fill = "none") Set legend type for each sestnetic: colorbar, legend, or none (no legend).

n • thems(legand.position = "bottom"). Place legand at "bottom", "top", "left", or "right"

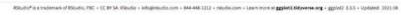
n • scale fill, discrete(name = "Title", tabels = C"A", "b", "C", "b", "E")) Set legend title and labels with a scale function.

Zooming



With clipping (removes unseen data points): 6 + x6im(0, 100) + y6im(10, 20)

t + scale_x_continuous(limits + c(0, 100)) + scale_y_continuous(limits + c(0, 100))



[r+theme_classic])

I r + theme linedraw()

. I r+theme_light()

cetterie minimali)

rethema void)

This call fully specifies the five components to the layer:

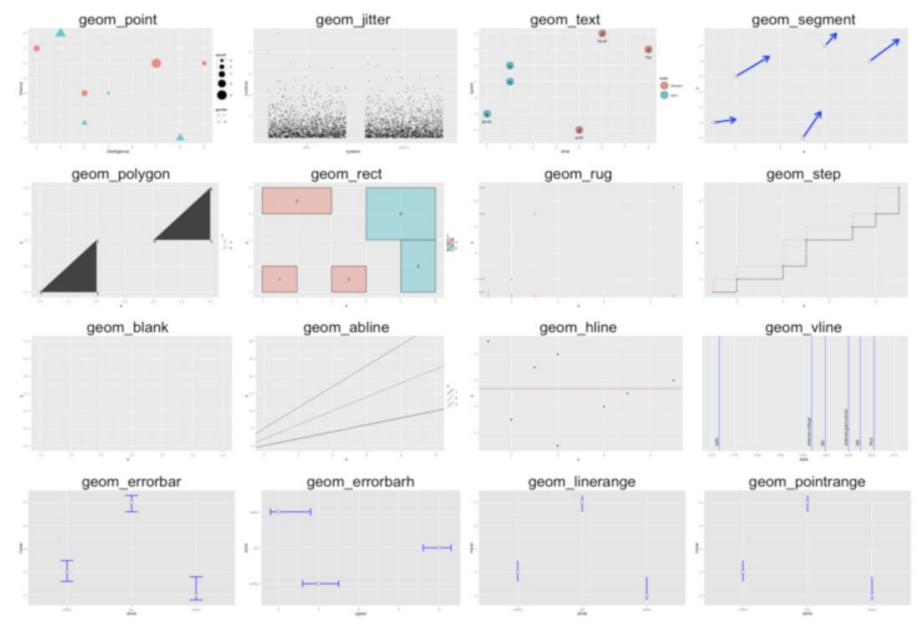
- 1. **mapping**: A set of aesthetic mappings, specified using the aes() function and combined with the plot defaults as described in aesthetic mappings. If NULL, uses the default mapping set in ggplot().
- 2. **data**: A dataset which overrides the default plot dataset. It is usually omitted (set to NULL), in which case the layer will use the default data specified in ggplot(). The requirements for data are explained in more detail in data. ggplot needs a data frame object for the input parameter, data, and not a table. If you specify a table as an input parameter, data then it will result in an error.
- 3. **geom**: The name of the geometric object to use to draw each observation. Geoms can have additional arguments. All geoms take aesthetics as parameters.

```
geom_jitter
                                                                      geom_hist
         geom_bin
                                                                                                          geom_label
                   geom_boxplot
                                  geom_density
                                                            geom_hex
                                                                                 geom_hline
                                                                                                                      geom_line
geom_bar
                                                geom_error
geom_point
                                     geom_ribbon
                                                                                                             geom_violin
                                                                          geom_smooth
                                                                                      geom_text
            geom_polygon
                                                                                                  geom_tile
                                                                                                                          geom_vline
                         geom_rect
                                                  geom_rug
                                                            geom_segment
```

ggplot2 builds charts through layers using geom_ functions. Here is a list of the different available geoms.

Find a complete list of all Geoms Here

Geometric objects (geoms) are the visual representations of (subsets of) observations. Their requirements differ.



This call fully specifies the five components to the layer:

4. **stat(istic)**: The name of the statistical transformation to use. A statistical transformation performs some useful statistical summary is key to histograms and smoothes. To keep the data as is, use the "identity" stat.

You only need to set one of stat and geom: every geom has a default stat, and every stat a default geom.

Most stats take additional parameters to specify the details of statistical transformation. You can supply params either in ... (in which case stat and geom parameters are automatically teased apart), or in a list called stat_params.

5. **position**: The method used to adjust overlapping objects, like jittering, stacking or dodging. More details in position.

Using stat_identity: the identity statistic leaves the data unchanged.

Geom	Description	Default Stat
geom_bar() # not pre-counted data where each observation contributes one unit to the height of the bar	Bar chart	stat_bin()
geom_col() # pre-counted data. y-position aesthetic mapped to the variable that has the counts.	I	
geom_point()	Scatterplot	stat_identity()
geom_line()	Line diagram, connecting observations in order by x -value	stat_identity()
geom_boxplot	Box-and-whisker plot	stat_boxplot()
geom_path	Line diagram, connecting observations in original order	stat_identity()
geom_smooth	Add a smoothed conditioned mean	stat_smooth()
geom_histogram	An alias for geom_bar() and stat_bin()	stat_bin()

Colour and fill

Lines

Polygons

Point

Text

ggplot2: Layers: Aesthetics

aes colour fill alpha

aes group order

aes linetype size shape

Colour related aesthetics: colour,

fill, and alpha

Aesthetics: grouping

Differentiation related

aesthetics: linetype, size, shape

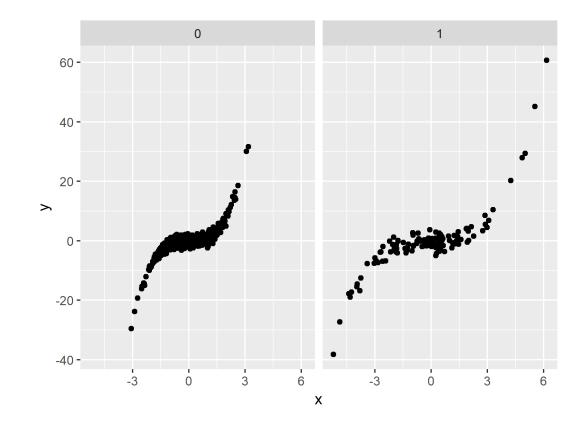
aes position

Position related aesthetics: x, y, xmin, xmax, ymin, ymax, xend, yend

https://ggplot2.tidyverse.org/reference/

ggplot2: Layers: Facetting

Facetting generates small multiples, each displaying a different subset of the data. Facets are an alternative to aesthetics for displaying additional discrete variables.



ggplot2: themes

This theme creates a set of rules for styling the components of the grammar of graphics. The first part of the theme creates the function theme_urban(), which handles sizes, spacing, font families, orientation, and the placement of elements.

```
→ ggplot2
 default
           theme_bw()
                        theme minimal()
                                           theme classic()
 theme_gray()
→ ggthemes
theme_excel()
                 theme economist()
 theme_fivethirtyeight()
                            theme_tufte()
                                             theme_gdocs()
theme_wsj()
               theme_calc()
                               theme hc()
\rightarrow other
 theme article()
                    theme pubclean()
                                       theme_bigstatsr()
 theme_ipsum()
```

The theme is based on ggplot2's web of inheritances. At the top, the theme sets default line, rectangle, and text styles. These three attributes are then passed as defaults to the next layer of arguments.



Example: Air Quality

- https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/airquality.html
- Daily air quality measurements in New York, May to September 1973.
- Daily readings of the following air quality values for May 1, 1973 (a Tuesday) to September 30, 1973.
 - Ozone: Mean ozone in parts per billion from 1300 to 1500 hours at Roosevelt Island
 - Solar.R: Solar radiation in Langleys in the frequency band 4000–7700 Angstroms from 0800 to 1200 hours at Central Park
 - Wind: Average wind speed in miles per hour at 0700 and 1000 hours at LaGuardia Airport
 - Temp: Maximum daily temperature in degrees Fahrenheit at La Guardia Airport.
 - Month: Numeric value between 1-12
 - Day: Numeric value between 1-31

Upload Numeric Data

```
## Download data from R using data() and see what the set is composed of
   ## Make sure you download data in the working directory
16
   data("airquality")
18 str(airquality)
10
  > str(airquality)
   'data.frame': 153 obs. of 6 variables:
    $ Solar.R: int 190
                          118 149 313 NA NA 299 99 19 194
                                  11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
    $ Wind
              : num
     Temp
              : int
                                    56 66 65 59 61 69 ...
   $ Month : int
   $ Day : int 1 2 3 4 5 6 7 8 9 10 ...
```

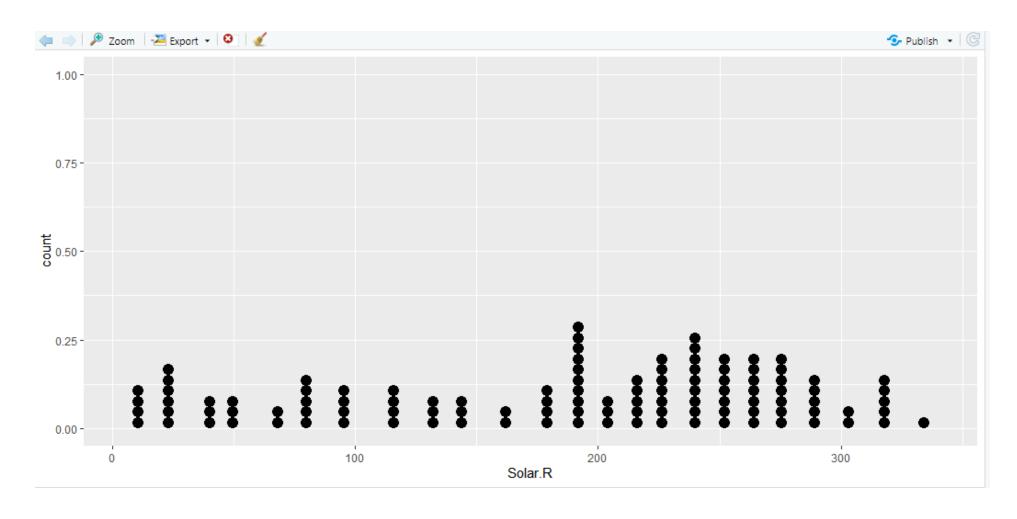
Data Cleaning

Output

```
'data.frame': 111 obs. of 6 variables:
 $ Ozone : int 41 36 12 18 23 19 8 16 11 14 ...
 $ Solar.R: int 190 118 149 313 299 99 19 256 290 274 ...
 $ Wind
        : num 7.4 8 12.6 11.5 8.6 13.8 20.1 9.7 9.2 10.9 ...
        : int 67 72 74 62 65 59 61 69 66 68 ...
 $ Temp
 $ Month : int 5 5 5 5 5 5 5 5 5 5 ...
          : int 1 2 3 4 7 8 9 12 13 14 ...
> ## To drop values option 2:
> y <- na.omit(airquality)</pre>
> str(y)
'data.frame': 111 obs. of 6 variables:
 $ Ozone : int 41 36 12 18 23 19 8 16 11 14 ...
 $ Solar.R: int 190 118 149 313 299 99 19 256 290 274 ...
        : num 7.4 8 12.6 11.5 8.6 13.8 20.1 9.7 9.2 10.9 ...
        : int 67 72 74 62 65 59 61 69 66 68 ...
 $ Temp
 $ Month : int 5 5 5 5 5 5 5 5 5 5 ...
          : int 1 2 3 4 7 8 9 12 13 14 ...
 - attr(*, "na.action")= 'omit' Named int [1:42] 5 6 10 11 25 26 27 32 33 34 ...
  ... attr(*, "names")= chr [1:42] "5" "6" "10" "11" ...
```

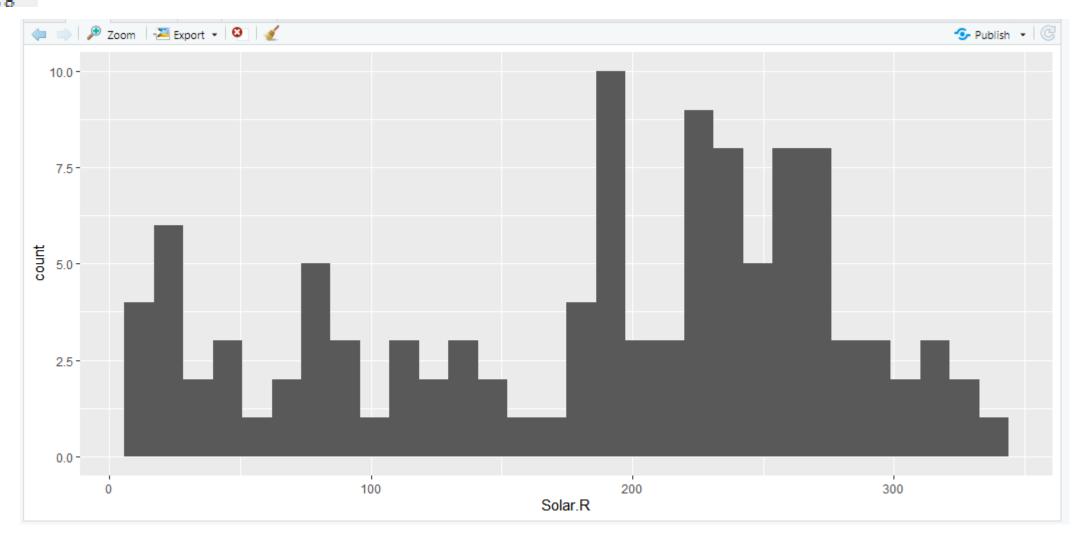
Dotplot

```
## Making a dotplot to show numerical data. It's like a bar chart,
## but with points stacked on top of each other
ggplot(y,aes(x=Solar.R)) + geom_dotplot(dotsize=0.4)
```



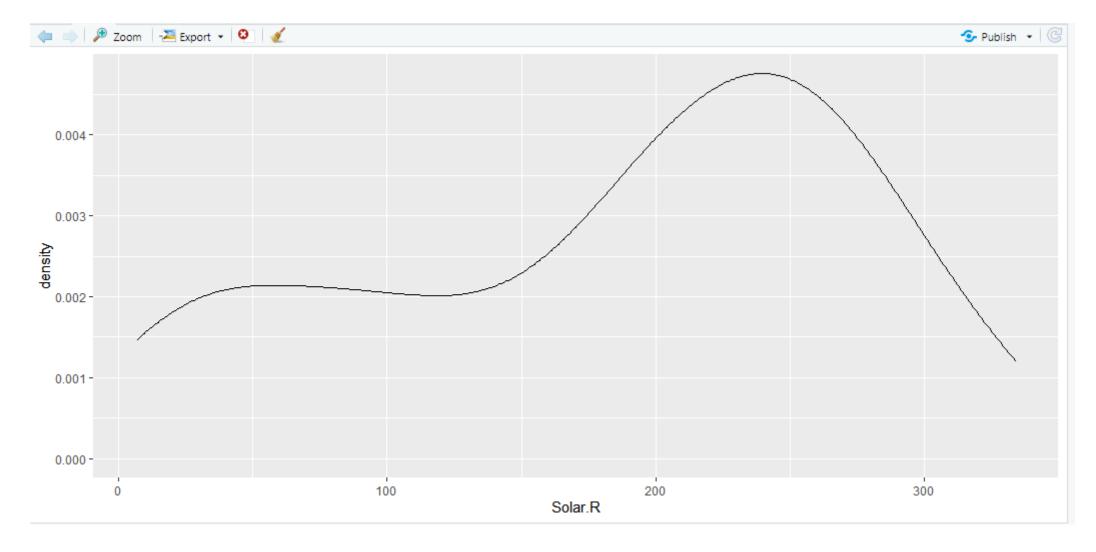
Histogram

```
## Histogram combines the dots, and the y axis now shows the actual count
ggplot(y,aes(x=Solar.R)) + geom_histogram()
```



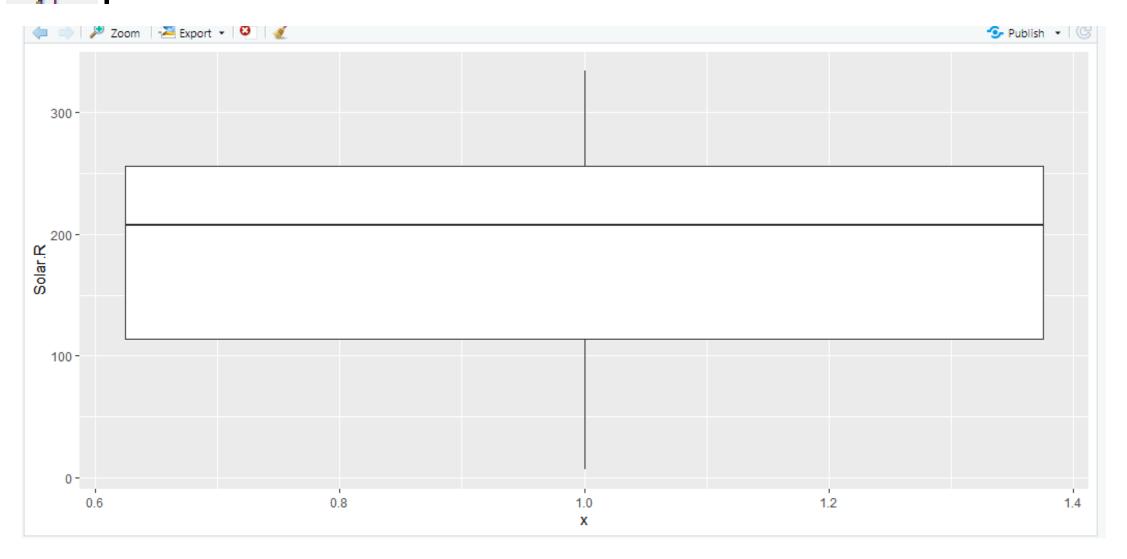
Density plot

```
## The shape of the distribution can be better represented with a density plot,
## without the stepwise nature of a histogram
ggplot(y,aes(x=Solar.R)) + geom_density()
```

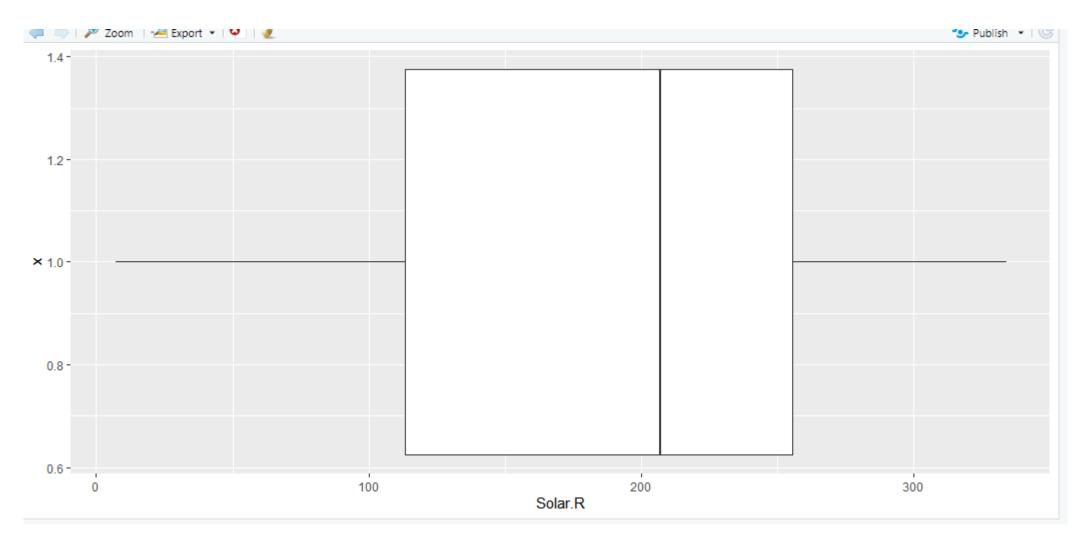


Boxplot

```
## Another view of distribution where you use a boxplot ggplot(y,aes(x=1,y=Solar.R)) + geom_boxplot()
```

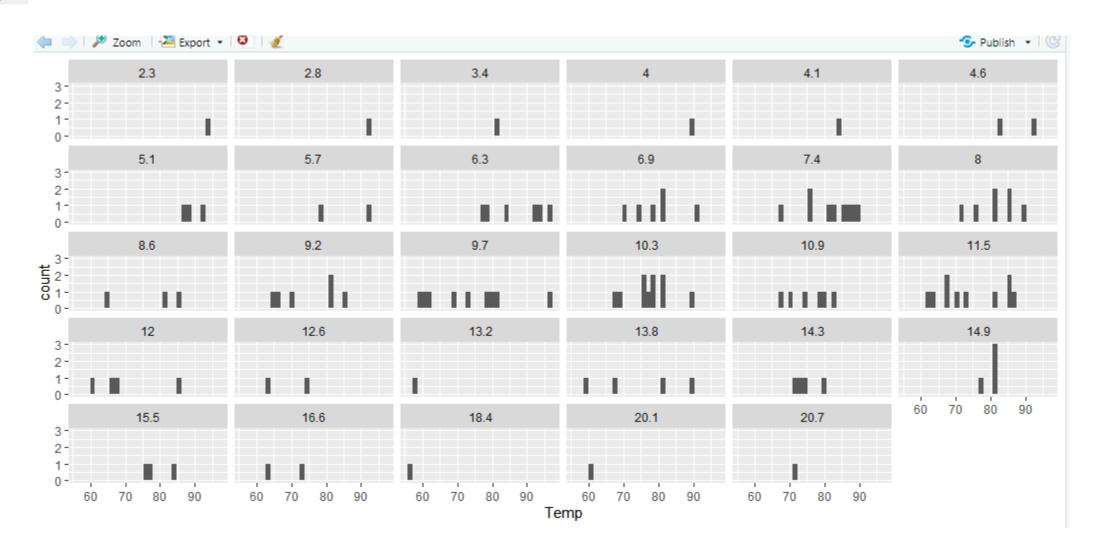


Boxplot (coord_flipped)



Faceted plots

```
48 ## Temperature faceted by wind speeds
49 ggplot(y,aes(x=Temp)) + geom_histogram() + facet_wrap(~Wind)
50
```



Dealing with colors in ggplot2

- Setting a color with fill and color
- geom_density(color="purple", # outlines the shape in the color

fill="69b3a2", ...) # fills the shape

- Picking a color with R:
 - Name: call a color by its name # colors()
 - rgb(): function builds a color using a quantity of red, green and blue. An additionnal parameter is available to set the transparency. All parameters ranged from 0 to 1.
 - *Number:* call a function by its number. (i.e. colors()[450]
 - Hex code: All colors can be defined by their hex code. A hex code looks like this: #69b3a2
- Change the color scale:
 - default (included in ggplot2)
 - R Color Brewer
 - Viridis
 - Paletteer

R Color Brewer

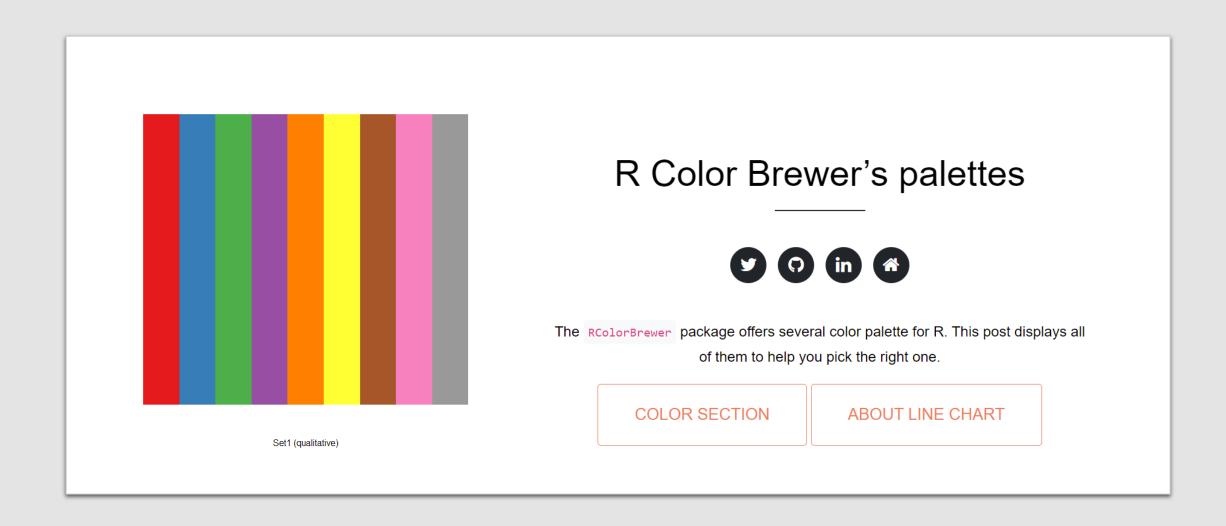
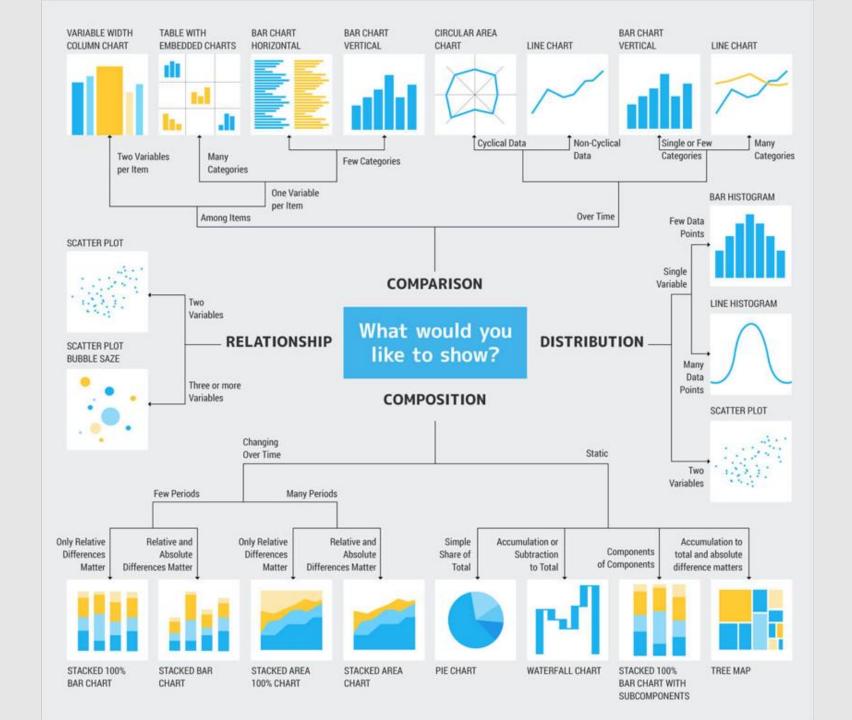




Chart types: <u>examples here</u>







Make your ggplot2 chart interactive with plotly()

https://plotly.com/r/



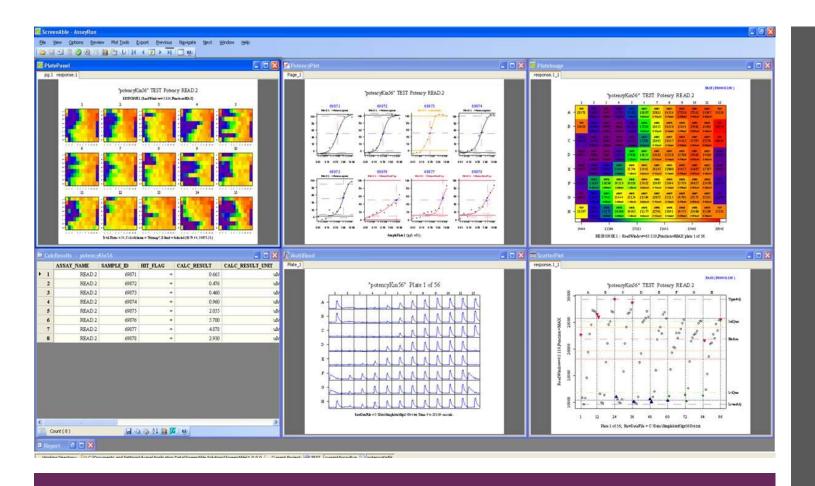
Plotly R Open Source Graphing Library

Plotly's R graphing library makes interactive, publication-quality graphs. Examples of how to make line plots, scatter plots, area charts, bar charts, error bars, box plots, histograms, heatmaps, subplots, multiple-axes, and 3D (WebGL based) charts.

Plotly.R is free and open source and you can view the source, report issues or contribute on GitHub.

Deploy R AI Dash apps on private Kubernetes clusters: Pricing | Demo | Overview | AI App Services

Fundamentals More Fundamentals >> The Figure Data Structure Structure Structure Structure Structure More Fundamentals >> More Fundamentals >> Structure More Fundamentals >> Structure Structure More Fundamentals >> Structure Exporting Graphs as Static Images



Packages for more visualization options

- ggforce: a collection of features providing missing functionality with the only commonality being their tie to the ggplot2 API.
- ggthemes: extra themes, geoms, and scales for ggplot2
- ggalluvial: mapping survey data
- esquisse: a Shiny gadget to create ggplot charts interactively with drag-anddrop to map your variables, explore and visualize your data
- dichromat: accommodating color blindness

Additional references:

- https://ggplot2.tidyverse.org/index.html
- https://towardsdatascience.com/tenrandom-but-useful-things-to-know-aboutggplot2-197dc4439d10
- https://datavizpyr.com/category/r/ggplot2/

