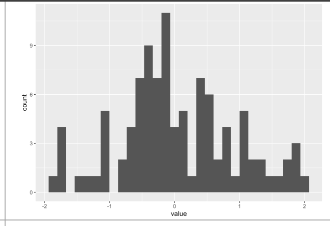
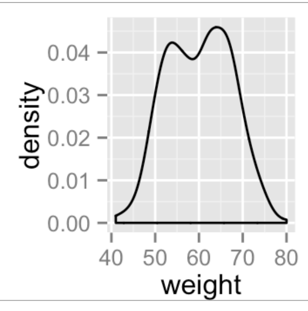
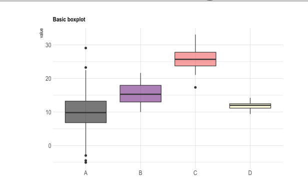
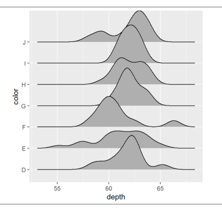
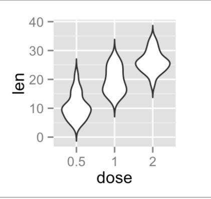
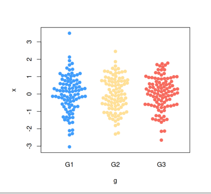
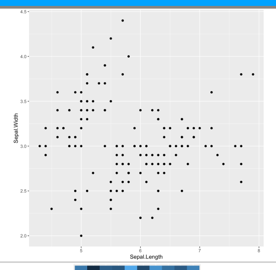
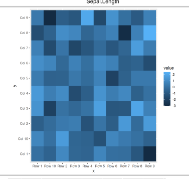
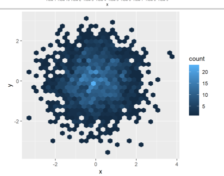
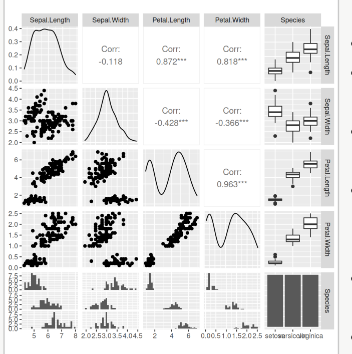
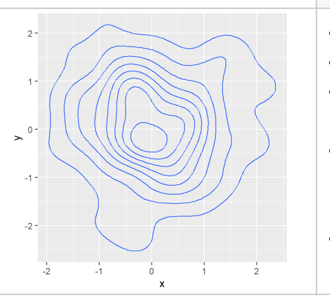
GGPLOT CHEATSHEET

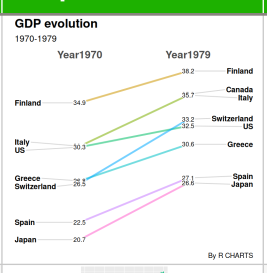
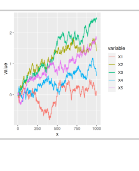
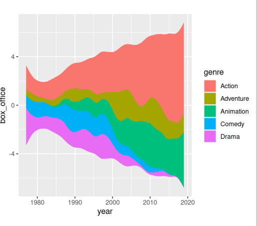
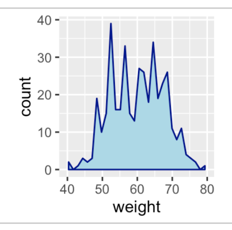
I. Distribution

	Code	Graph	Argument/function notes
Histogram (Display quantitative variable; Better visualizing the shape of the data distribution)	**ggplot**(df, **aes**(x = x)) + **geom_histogram**()		• **binwidth:** control bin size
Density Plot (Smoothed version of histogram)	**ggplot**(df, **aes**(x = x)) + **geom_density**()		• **fill:** change fill colors • **color:** change line colors • **linetype:** change line type • **adjust:** adjust the bandwidth(smoothness)
Boxplot (Display continuous variable; Better visualize summary statistics and outlying points individually)	**ggplot**(df, **aes**(y = y)) + **geom_boxplot**()		• **outlier.colour, outlier.shape, outlier.size:** Change the color, the shape and the size for outlying points • **notch: =TRUE** add notches to assess whether the medians are different
Rigdeline Plot (Useful for visualizing changes in distributions over time or space)	**library**(ggridges) **ggplot**(df, **aes**(x = value, y = category)) + **geom_density_ridges**()		• **stat:** change shape of each distribution • **rel_min_height**: cut the trailing tails • **scale**: control the scaling of the ridgelines relative to the spacing between them • **quantile**: control which or how how many quantiles are displayed
Violin Plot (Like density plot; Useful for comparison of distributions between several groups — peaks&valleys&tails)	**ggplot**(df, **aes**(x = category, y = value)) + **geom_violin**()		• **trim: =TRUE**(default) trim the tails of the violins • **fill**: change fill colors • **stat_summary**: add mean/median points • Can combine with **geom_boxplot()**
Beeswarm Plot (suitable for a relatively small number of measurements due to the non-overlapping of each point)	**library**(ggbeeswarm) **ggplot**(df, **aes**(x = group, y = y)) + **geom_beeswarm**()		• **cex:** change spacing • **size:** change size of the points • **priority:** change point layout • Can combine with **geom_boxplot()**

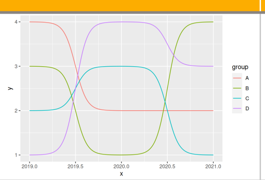
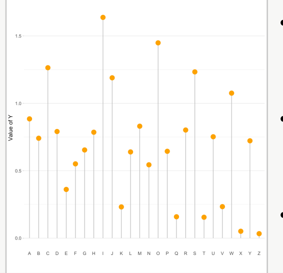
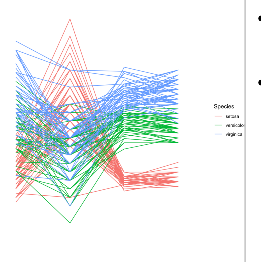
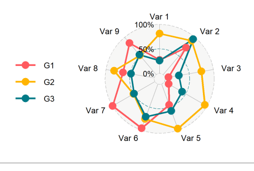
II. Correlation

	Code	Graph	Argument/function notes
Scatterplot (Display the relationship between two quantitative variables)	**ggplot**(cars, **aes**(x = x, y = y)) + **geom_point**()		• **geom_text:** label points • **geom_smooth/geom_abline:** add regression line
Heatmap (Better visualize the volume of locations/events within a dataset)	**ggplot**(df, **aes**(x = category, y = category, fill = value)) + **geom_tile**()		• **color:** change tiles color • **lwd:** change line width • **linetype:** change line type • **geom_text(aes(label=value)):** add the values over the tiles
Hexbin chart (Useful for visualizing the relationship of two quantitative variables with a lot of data points)	**ggplot**(df, **aes**(x = x, y = y)) + **geom_hex**()		• **bin:** change number of bins • **color:** change border color • **fill:** change color for all hexagons • **alpha:** control the transparency
Paris plot (Plot matrix — quickly see all relationships between each variables no matter categorical or numerical)	**library**(GGally) **ggpairs**(df)		• **columns:** select the columns of dataset to be plotted. • **aes(color=var):** color by groups • **upper/lower/diag:** represents the location of the panel • upper=list(**continuous=“smooth”**): add scatterplots with lines • “densityDiag”,“barDiag”, “blankDiag” can be assigned to continuous • **combo:** assign a different chart for the categorical variables. • “facetdensity”, “count” can be assigned to combo
Contour plot (visualize 3d surfaces in 2d)	**ggplot**(df, **aes**(x = x, y = y)) + **geom_density_2d**()		• can combine with **geom_point()** • **color:** change line color • **aes(color=..level..):** colorize each contour line based on the level • **geom=“ploygon”/ geom_density_2d_filled():** fill the contour • **scale_fill_brewer()/ scale_fill_manual():** change fill color

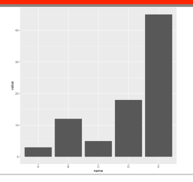
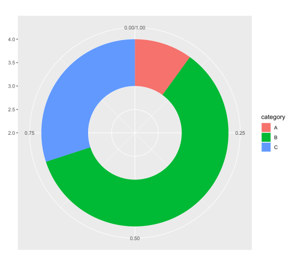
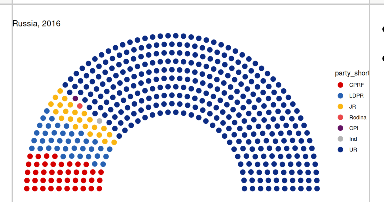
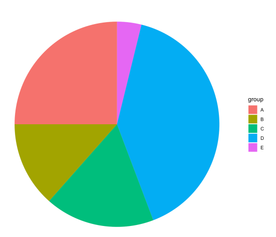
III. Evolution

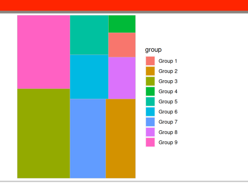
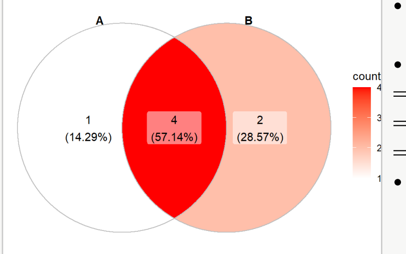
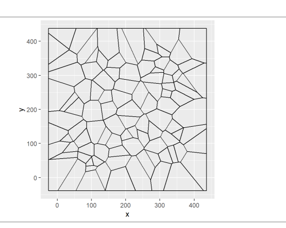
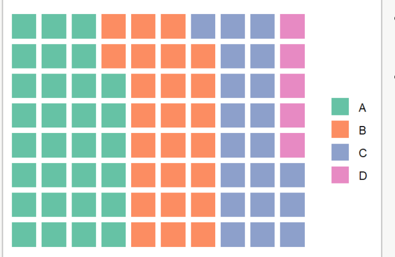
	Code	Graph	Argument/function notes
Slopegraph (Continuous data: highlighting change over time) Categorical data: highlighting difference between two categories)	**library**(CGPfunctions) **newggslopegraph**(dataframe = df, Times = Year, Measurement = GDP, Grouping = Country)		• **ReverseXAxis/ReverseYAxis:** reverse axis • **LineColor:** change line color • **LineThickness:** change line width • **ThemeChoice:** modify theme(“ipsum”/ “econ”/“gdocs”)
Line Graph (Comparison between different variables)	**ggplot**(df, **aes**(x = index, y = value, color = variable)) + **geom_line**()		• **linetype:** change line style • **lwd:** change line width
Streamgraph (Display the evolution of a numeric variable for several groups)	**library**(ggstream) **ggplot**(df, **aes**(x = year, y = value, fill = genre)) + **geom_stream**()		• **geom_stream_label(aes(label=)):** add the labels to each area of the streamgraph • **type:** change type(default:“mirror”; “ridge”:stacks from x-axis; “proportional”: streams sum up to 1) • **color:** change border color • **scale_fill_manual(values=cols):** change the fill colors
Area plot (Color under density curve)	**ggplot**(df, **aes**(x = index, y = value)) + **geom_area**()		• **aes(y=..density..):** set y axis as density value • **color:** change line color • **fill:** change fill color • **linetype:** change line type • **facet_grid():** split plots in multiple panels

IV. Ranking

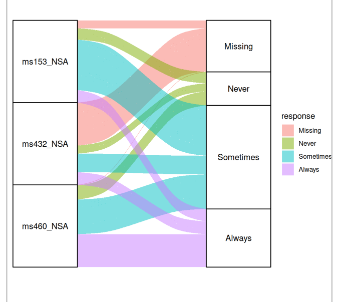
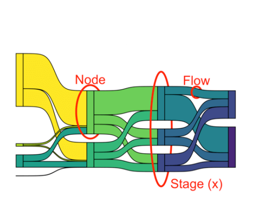
	Code	Graph	Argument/function notes
Bump Chart (Visualize change in rankings of different groups over time)	**library**(ggbump) **ggplot**(df, **aes**(x = yera, y = ranking, color = group)) + **geom_bump**()		• Can combine with **geom_point()** • **scale_fill_brewer()/ scale_fill_manual():** change color of the lines and points
Lollipop chart (kind of bar chart; useful for making comparisons between different categories. Also, ranking or showing trends over time)	**ggplot**(df, **aes**(x = group, y = value)) + **geom_segment**(**aes**(x = group, xend = group, y = 0, yend = value)) + **geom_point**()		• **coord_flip():** flip the chart; better to use when there are too many categories • **linetype:** change line type(“dotted”, “dashed”, “dotdash”) • **geom_segment(y=??):** change base line
Parallel Coordinate (visualizing high-dimensional datasets; data frame must have several numeric variables)	**library**(GGally) **ggparcoord**(data = df, column=1:4, groupColumn:5) (**column:** several numeric variables to be axes **groupColumn:** a single categorical variable used to color lines)		• **showPoints:** add dots • **alphaLines:** modify line transparency • **Scale:** scaling data(“globalminmax”-no scaling; “uniminmax”-min=0&max=1; “std”-normalize; “center”-standardize and center)
Radar chart (Compare two or more groups on various characteristics)	**library**(ggradar) **ggradar**(df)		• **values.radar:** label the grid • **axis.labels:** label the variables • **Group.colours:** change line colors

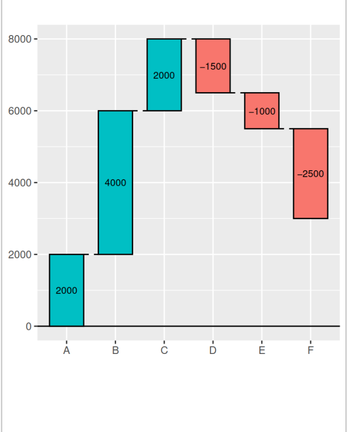
V. Part of a Whole

	Code	Graph	Argument/function notes
Bar chart (Display categorical variables)	**ggplot**(df, **aes**(x = x, fill = group)) + **geom_bar**()		• **width:** control bin width • **coord_flip():** change to horizontal barplot
Donut Chart (Display individual categories percentages of the whole; can compare a handful of categories)	**ggplot**(df, **aes**(x = hsize, y = value, fill = group)) + **geom_col**() + **coord_polar**(theta = **"y"**)		• **theme_void():** get rid of unnecessary background, axis, etc. • **Hsize:** change hole size • **geom_label():** add labels
Parliament Diagram (Visualize parliament layouts)	**library**(ggparliament) **ggplot**(df, **aes**(x = x, y = y, colour = category)) + **geom_parliament_seats**() + **theme_ggparliament**()		• **type:** change type(eg.“circle”) • **geom_parliament_bar():** add a parliament bar showing the proportion of seats by party.
Pie chart (Compare different segment proportion of the data; only one category)	**ggplot**(df, **aes**(x = **""**, y = value, fill = group)) + **geom_bar**() ***#geom_col()*** + **coord_polar**(theta = **"y"**)		• **theme_void():** get rid of unnecessary background, axis, etc. • **geom_text():** add labels

	Code	Graph	Argument/function notes
Tree maps (Display data that is grouped and nested in a hierarchical structure)	**library**(treemapify) **ggplot**(df, **aes**(area = value, fill = group/value)) + **geom_treemap**()		• **geom_treemap_text():** add labels to the tiles For above function, argument grow=TRUE: fit the text to the tiles
Venn Diagram (Illustrate logical relationships between two or more variables)	**library**(ggVennDiagram) **ggVennDiagram**(list)		• **category.names:** change and label group names • **label:** change label type =“percent”: labels with percentage =“count”: labels with count =NULL: remove labels • **label_alpha:** modify label transparency
Voronoi Diagram (scattering points at random on a Euclidean plane)	**library**(ggvoronoi) **ggplot**(df, **aes**(x, y)) + **stat_voronoi**(geom = **"path"**)		• Can combine with **geom_point()** • **aes(fill=var):** pass a variable to fill argument can create a Voronoi heatmap • **outline:** change shape of bounding box
Waffle chart (Effective when comparing numbers that are highly variant)	**library**(waffle) x <- **c**(30, 25, 20, 5) **waffle**(x, rows = 8 Or **ggplot**(df, **aes**(fill=group, values=value) +**geom_waffle**(n_rows=8, size=0.33)		• **iron():** combine different waffle charts • **keep=FALSE:** get rid of unused categories

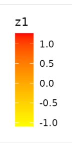
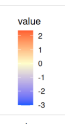
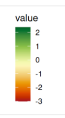
VI. Flow

	Code	Graph	Argument/function notes
Alluvial Plot (Visualize change in groups between states or over time/ useful for showing how features of a population are related)	**library**(ggalluvial) **ggplot**(data = df, **aes**(axis1 = survey, axis2 = response, y = freq)) + **geom_alluvium**(**aes**(fill = response)) + **geom_stratum**()		• **aes(axis1, axis2, axis3,..):** contain variables on x-axis • **curve_type(in geom_alluvium):** change flow type (eg. “linear”, “cubic”, “sigmoid”) • **fill(in geom_alluvium):** change flow color • **fill(in geom_stratum):** change stratum color
Sankey Diagram (Visualize the proportional flow between variables/ useful for showing flows or processes where the some quantity need to be tracked)	**library**(ggsankey) **ggplot**(df, **aes**(x = stage, next_x = next_stage, node = node, next_node = next_node, fill = **factor**(node))) + **geom_sankey**()		• **aes(label=) +geom_sankey_label():** add labels and change label appearance • **flow.alpha:** modify flow transparency • **node.color:** change node color

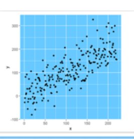
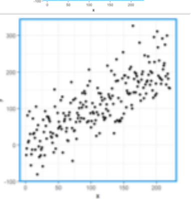
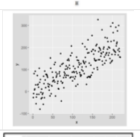
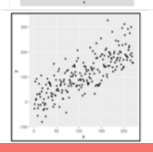
	Code	Graph	Argument/function notes
Waterfall Chart (Illustrate the gradual transition in the quantitative value)	**library**(waterfalls) **waterfall**(df)/ **waterfall**(values=value, labels=group)		• **calc_total=TRUE:** calculate the total(final result after the change) • **rect_width:** control rectangle width • **draw_line:** remove/add dashed line joining the rectangles • **linetype:** change line type • **fill_by_sign=TRUE:** positive&negative values each have same color • **fill_colours:** change rectangles color • **total_rect_color:** change total rectangle color • **rect_border:** change border rectangle color

CUSTOMIZATION

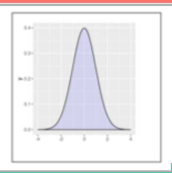
Color palette-1

Functions	Notes	Example	Graph
scale_fill_gradient	Allows changing the colors, setting a lower and a higher color to represent the values.	**scale_fill_gradient**(low="yellow", high="red")	
scale_fill_gradient2	Add a mid color	**scale_fill_gradient**(low="#075AFF", mid="FFFFCC", high="FF0000")	
scale_fill_gradientn	Use a customized color palette	**scale_fill_gradientn**(colors=hcl.colors(20, "RdYlGn")) [passing 20 colors of "RdYlGn" palette]	
scale_color_viridis_c	Use the viridis palette (most common form for color blindness)	argument option: There are some colormap options to use(A,B,C,D,E)	
scale_fill_brewer	Use color palette from RColorBrewer package	**scale_fill_brewer**(palette="Dark2")	🟩 F 🟧 M
scale_fill_manual	Use custom color palettes	**scale_fill_manual**(values=c("#999999", "E69F00")	⬜ F 🟨 M
scale_fill_grey	Use grey scale	**scale_fill_grey**()	⬛ F ⬜ M
scale_fill_hue	Quantitative color scale with evenly spaced hues		

Color

Functions	Notes	Example	Graph
Panel	Background	**theme(panel.background** = **element_rect**(fill = "#67c9ff"))	
	Border	**theme(panel.border** = **element_rect**(fill = "transparent", # Needed to add the border color = 4, # Color of the border size = 2)) # Border width	
geom_label()	Background	**theme(plot.background** = **element_rect**(fill = "#67c9ff"))	
	Border	**theme(plot.background** = **element_rect**(color = "black", # Border Color size = 2)) # Border width	

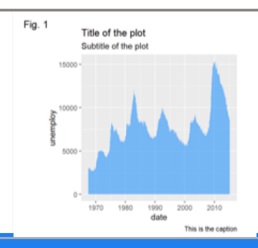
Margin

Customize margins	* set to 0 to remove margin * set to negative numbers to reduce more margin	theme(plot.margin = margin(t = 20, # Top margin r = 50, # Right margin b = 40, # Bottom margin l = 10)) # Left margin	

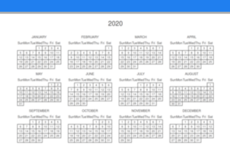
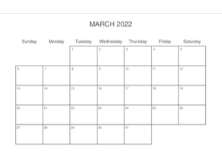
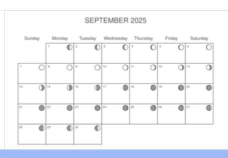
Text

Functions	Notes	Example	Graph
geom_text()	Allows adding text	**geom_text**(aes(x = -115, y = 25, label = "Map of the United States"), stat = "unique")	
		geom_text(aes(label = state))	
geom_label()	Allows adding label	**geom_label**(aes(x = -115, y = 25, label = "Map of the United States"), stat = "unique")	
		geom_label(aes(label = state))	
Package: ggrepel	Avoid overlapping	**geom_text_repel**(aes(label = state))	
		geom_label_repel(aes(label = state))	

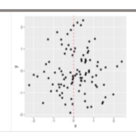
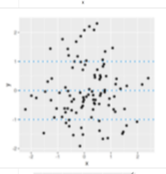
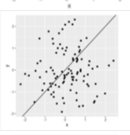
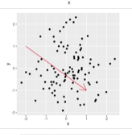
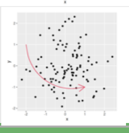
Title

Functions	Note	Example	Graph
labs()	Set a title, a subtitle, a caption and a tag.	**labs**(title = **"Title of the plot"**, subtitle = **"Subtitle of the plot"**, caption = **"This is the caption"**, tag = **"Fig. 1"**)	

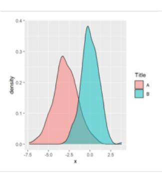
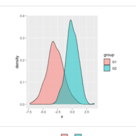
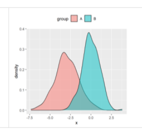
Calander

library(calendR) **calendR**()	Create a yearly calendar when specify the year in the year argument.	**calendR**(year = 2020)	
	Create a monthly calendar when specify the year in the year argument.	**calendR**(year = 2022, month = 3)	
	Create a lunar calendar when specify the year in the year argument.	**calendR**(year = 2025, month = 9, lunar = TRUE)	

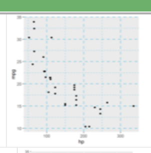
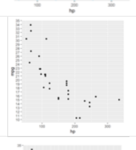
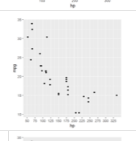
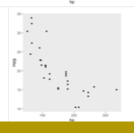
Lines, Arrows, Curves

	Notes	Example	Graph
Vertical line	Add vertical lines	**geom_vline**(xintercept = -1:1, linetype = 1, color = 2:4)	
Horizontal line	Add horizontal lines	**geom_hline**(yintercept = -1:1, linetype = 3, color = 4, lwd = 1)	
Diagonal line	Add diagonals	**geom_abline**(intercept = 0, slope = 1)	
Line Arrow	Add line arrow	**geom_segment**(x = -2, y = 1, xend = 1, yend = -1, color = 2, arrow = **arrow**())	
Curve Arrow	Add curve arrow	**geom_curve**(x = -2, y = 1, xend = 1, yend = -1, color = 2, arrow = **arrow**())	

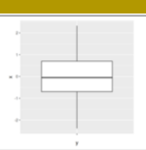
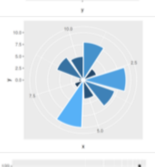
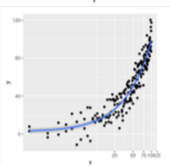
Legend

	Notes	Example	Graph
Add	*color*, *fill*, *shape* or *alpha* inside aes		
Title	Change legend	**guides**(fill = guide_legend(title = "Title")) **labs**(fill = "Title") **scale_fill_discrete**(name = "Title")	
Label	Change label	**scale_fill_hue**(labels = c("G1", "G2"))	
Position	Change position	**theme**(legend.position = "top")	
Remove	Turn of the legend	**theme**(legend.position = "none")	

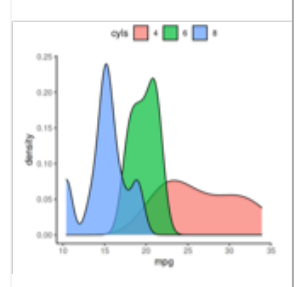
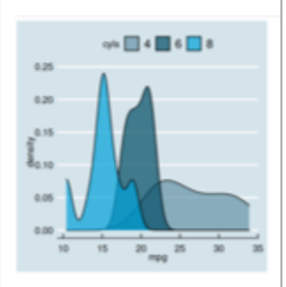
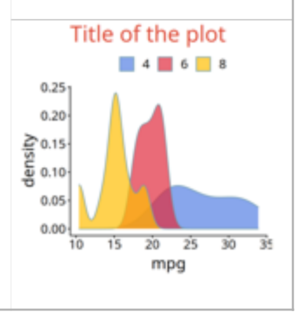
Grid

Customization	Set the grid aesthetics and customize the color, line width and line type	**theme(panel.grid** = **element_line**(color = "#8ccde3", size = 0.75, linetype = 2))	
Grid Break	Customize the number of grid breaks	**scale_y_continuous**(breaks = seq(10, 35, by = 1))	
		scale_x_continuous(breaks = seq(50, 350, by = 25))	
Remove Grids	Remove Grids	**theme(panel.grid** = element_blank())	

Coordinate

coord_flip()	Rotate the axes	ggplot(df, aes(x = x, y = "")) + geom_boxplot() + **coord_flip**()	
coord_trans()	Create transformed cartesian coordinate systems	ggplot(df, aes(x = x, y = y)) + geom_point() + geom_smooth(method = "lm")+ **coord_trans**(x = "log")	
coord_polar()	Create polar coordinates	ggplot(df, aes(x = x, y = y, fill = y)) +geom_bar(stat = "identity", color = "white",lwd = 1, show.legend = FALSE) + **coord_polar**()	

Themes

	Notes	Example	Graph
In-Buit Themes	theme_grey() [default]; theme_bw(); theme_light(); theme_linedraw(); theme_dark(); theme_void(); theme_minimal(); theme_classic()	ggplot(mtcars, aes(x = mpg, fill = cyls)) + geom_density(alpha = 0.7) + **theme_classic()** + theme(legend.position = "top")	
Package:ggthemes	The package contains several very popular themes. Some of them also come with their corresponding color scales.	ggplot(mtcars, aes(x = mpg, fill = cyls)) + geom_density(alpha = 0.7) + **theme_economist() + scale_fill_economist() +** theme(legend.position = "top")	
Package:ggtech	The package provides themes inspired by tech companies, such as Airbnb, Google, Twitter or Facebook.	ggplot(mtcars, aes(x = mpg, fill = cyls)) + geom_density(alpha = 0.7) + ggtitle("Title of the plot") + **theme_tech(theme = "google") + scale_fill_tech(theme = "google") +** theme(legend.position = "top")	

Teammate: Yiqin Shi(ys3481), Yuxuan Liang(yl4871)
Citation: "GGPLOT2 Package." R CHARTS | A Collection of Charts and Graphs Made with the R Programming Language, https://r-charts.com/ggplot2/.