SIRE516

Data visualization

Basic R plots
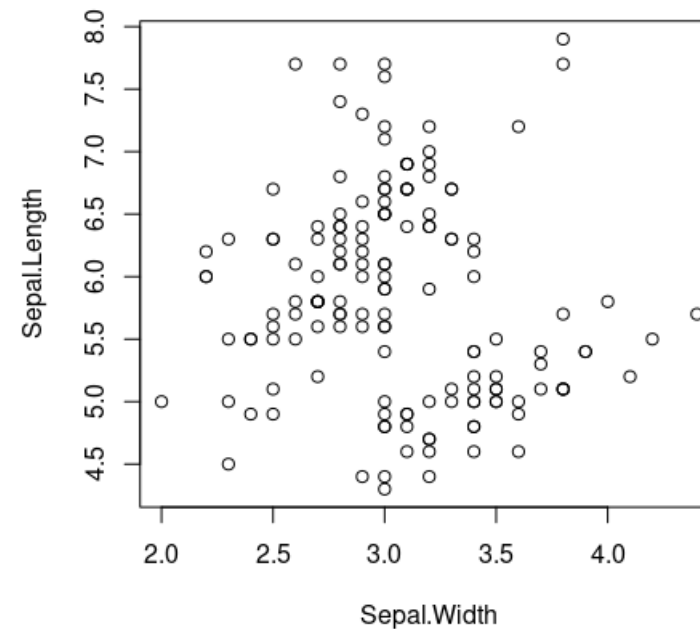
Basic R plots

- Pro: Fast and simple
- Con: More difficult customization

- In this class, we will use these plots for quick visualization.
- I will not cover customization of these plots.
- Plot customizations will be done in 'ggplot.'

Scatter plot

- Showing two dimensional data
- Two numerical values

```
1 ▾  #### Scatter plot ####
2    ?iris
3
4    plot(Sepal.Length ~ Sepal.Width, iris)
```

Line plot

- Showing two dimensional data
- Two numerical values
- The line can show:
    - Repeated measurements over time (longitudinal data with X variable = time)
    - Trend of correlation between variables

```
 6 ▾ #### Line plot ####
 7   x <- -3:3
 8   y <- x^3
 9   y2 <- x^2
10
11   plot(x=x ,y = y) # Point by defualt
12   plot(x=x ,y = y, type = "l")
13
14   plot(x=x ,y = y, type = 'n') # Blank plot
15   lines(x = x, y = y, col = "red") # Add first line
16   lines(x = x, y = y2, col = "blue") # Add second line
17
```
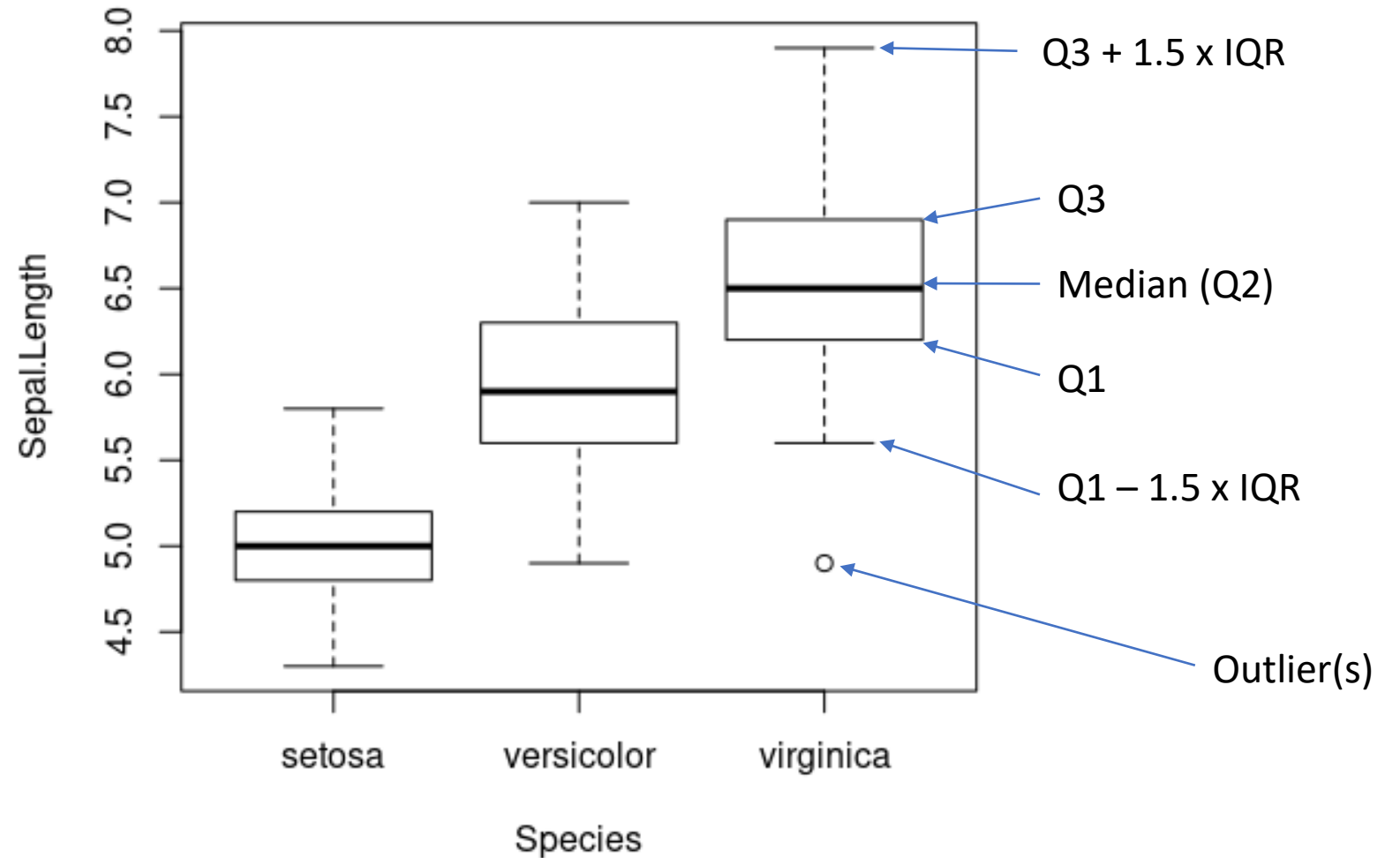
Box plot

- Showing two dimensional data
- One numerical value and one categorical value

```
6 ▾  #### Box plot ####
7
8    boxplot(Sepal.Length ~ Species, iris)
9
```
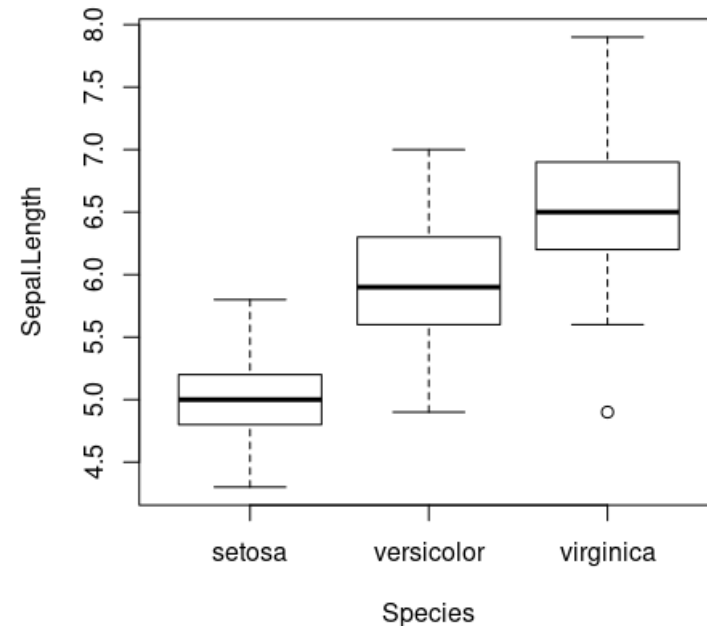
Box plot

- Interpretation
- IQR = Q3-Q1

Bar plot

- One-dimension data → Counts of categorical variables
- Two-dimension data → One categorical and one central value (e.g. mean) of numerical variables
  - Mean + Standard error of the mean: If data are not normally distributed, the shape of distribution is lost.
  - Boxplot for showing how data are distributed

```
22 ▾  #### Bar plot ####
23    rows <- sample(nrow(iris),size = 97, replace = F)
24    IRIS <- iris[rows,]
25    tb <- table(IRIS$Species)
26
27    print(tb)
28    barplot(tb)
```
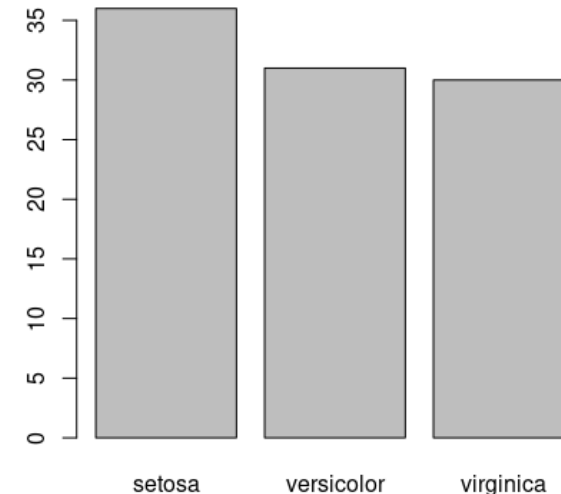
Pie chart

- One-dimension data → Counts of categorical variables
- Caution: Pie chart poorly presents data (angle vs height in bar plot). Avoid using it

```
30 ▾  #### Pie chart ####
31    pie(tb)
```
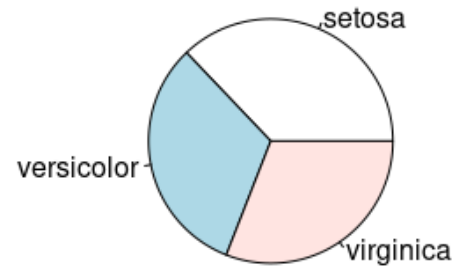
How to save R plots

- Specify where (what file format) you want to direct the plot from the screen/terminal to

- Plot the plot

- Turn off the direction

```
33  #### Save R plots ####
34  png("PIE.png")
35  pie(tb)
36  dev.off()
```

Practical 1

Practical 1

- We will use basic R plots to explore 'mtcars' dataset
- Try to select appropriate plot types

```
1    ?mtcars
2
3    #1) Count of cars by the type of transmission
4    #2) MPG by the type of transmission
5    #3) Weight vs MPG
```

ggplot2

Why ggplot2?

- Coding is more intuitive?
- Customizations!
- Plots with publication quality
  - Resolution
  - Format (fonts and special characters e.g. the Lancet's decimal: 23·4, not 23.4)
- Packages for advanced plots built from ggplot2
  - Survival analysis
  - Spatial statistics
- Require some data preparations

Basic structure of ggplot2

```
graph <- ggplot(data = data1, aes(x = x1, y = y1, col = z1)) +
        geom_point() +
        geom_line(data = data2, aes_string(x = "x2", y = "y2", col = "z2"))+
        scale_color_discrete(…)+
        scale_x_continuous(…)+
        theme(panel.grid = …)
```

- Plot elements are added step-by-step

- Aesthetic mappings: aes (no "_") and aes_string (with "_") telling what plot element connect to what variable

- Assign plot elements outside of aes() → Constant value (x = 1, col ="red")

Basic structure of ggplot2

```
graph <- ggplot(data = data1, aes(x = x1, y = y1, col = z1)) +
        geom_point() +
        geom_line(data = data2, aes_string(x = "x2", y = "y2", col = "z2"))+
        scale_color_discrete(…)+
        scale_x_continuous(…)+
        theme(panel.grid = …)
```

- geom_[plot type] → add data points to the plot as indicated by "plot_type"

- scale_[something]_[data_type] → scale plot elements (usually) mapped by aes in a certain manner (e.g. log10-scale, customized color)

- theme(…) → customize 'static' plot elements (e.g. fonts, grid)

Scatter plot

```r
38 ▾  #### ggplot2: scatter plot ####
39    library(ggplot2)
40    ?iris
41    g <- ggplot(data = iris) # Blank plot
42    print(g)
43    g1 <- g + geom_point(aes(y = Sepal.Length, x = Sepal.Width)) # Add data as scatter plot
44    print(g1)
45    plot(Sepal.Length ~ Sepal.Width, iris) # R plot
46    # Add data as scatter plot with colors of points indicatated by species
47    g2 <- g + geom_point(aes_string(y = "Sepal.Length", x = "Sepal.Width", col = "Species"))
48    print(g2)
```

Line plot

```r
#### ggplot2: line plot ####
x1 <- -3:3
y1 <- x1^3
y2 <- x1^2

#Plot from vectors
ggplot()+
  geom_line(aes(x = x1, y = y1), col = "red") +
  geom_line(aes(x = x1, y = y2), col = "blue")

#Plot from data frame
dat1 <- data.frame(x = x1, y1 =  y1, y2 = y2)
rm(x1,y1,y2)
ggplot()+
  geom_line(data = dat1, aes(x = x, y = y1), col = "red")+
  geom_line(data = dat1, aes(x = x, y = y2), col = "blue")

#Plot from melt data frame # For multiple lines (spaghetti plot)
library(reshape2)
mdat1 <- melt(dat1,id.vars = "x")
head(mdat1)
# 'group' tells which data points are on the same line.
# col (color) tells which data points have the same or different colors
ggplot(data = mdat1, aes(x=x,y=value,group=variable,col=variable)) +
  geom_line()
```

Box plot

```
76  #### ggplot2: box plot ####
77  ggplot(data = iris)+
78    geom_boxplot(aes(x= Species, y = Sepal.Length))
79  # col vs fill
80  ggplot(data = iris)+
81    geom_boxplot(aes(x= Species, y = Sepal.Length, col = Species), fill = "black")
82  ggplot(data = iris)+
83    geom_boxplot(aes(x= Species, y = Sepal.Length, fill = Species), col = "black")
```

Bar plot

```
85 ▾  #### ggplot2: bar plot ####
86    rows <- sample(nrow(iris),size = 97, replace = F)
87    IRIS <- iris[rows,]
88
89    #Plot uncounted data
90    ggplot(IRIS)+
91      geom_bar(aes(x = Species) ,stat = "count")
92
93    #Plot counted (aggregated) data
94    aIRIS <- data.frame(table(IRIS$Species))
95    print(aIRIS)
96    ggplot(aIRIS)+
97      geom_bar(aes(x = Var1, y = Freq) ,stat = "identity")
```

- stat = "identity" can be used to plot "center" values of aggregated data (e.g. mean of group)

Pie chart

- No pie chart function in ggplot2
- Geom_bar(…)+ coord_polar(…)

```
 99 ▾ #### ggplot2: pie chart ####
100   gpie <- ggplot(aIRIS)+
101     geom_bar(aes(x = "", y = Freq, fill = Var1) ,stat = "identity") #x must be ""
102   print(gpie)
103   gpie <- gpie + coord_polar("y", start=0)
104   print(gpie)
```

How to save ggplot2

- Simply use "ggsave()" function
- Name the type of the picture you want to save (e.g. tiff, jpeg, png)
- Specify dimension and units (e.g. width = 3, height = 4, unit = "in")
- Specify resolution (For publication dpi > 300)

```
106 ▾  #### save ggplot2 ####
107  ggsave(plot = gpie, file = "Plots/pie.tiff",width = 3, height = 3, units = "in",dpi = 600)
```

Practical 2

Practical 2

- As in practical 1, we will use ggplot2 to explore 'mtcars' dataset
- Try to select appropriate plots plot types

```
1   ?mtcars
2
3   #1) Count of cars by the type of transmission
4   #2) MPG by the type of transmission
5   #3) Weight vs MPG
```

ggplot2 customization

Naming title and axes

```
110 ▾ #### Naming title and axes ####
111    print(aIRIS)
112    g <- ggplot(aIRIS)+
113      geom_bar(aes(x = Var1, y = Freq, fill = Var1) ,stat = "identity")
114    print(g)
115    g <- g + ggtitle("iris dataset") # Add title
116    print(g)
117    g <- g + ylab("Count (n)") +  # Add/change axis titles
118      xlab("Species")
119    print(g)
```

Customize axes

```r
121 ▾  #### Customize axes ####
122    g <- ggplot(aIRIS)+
123      geom_bar(aes(x = Var1, y = Freq, fill = Var1) ,stat = "identity") +
124      ggtitle("iris dataset")
125
126    print(g)
127
128    # Specify numbers on the y-axis to be shown
129    g + scale_y_continuous(name = "Count (n)", breaks = seq(0,40,5))
130
131    # Plot data on log10 scale without log10-transforming the data
132    g + scale_y_log10(name = "Count (n)")
133
134    # Rename discrete values on X-axis
135    g + scale_x_discrete(name = "Species", labels = c("S","VE","VI"))
```

Customize color and fill

```
137  #### Customize color and fill ###
138  g <- ggplot(aIRIS)+
139    geom_bar(aes(x = Var1, y = Freq, fill = Var1, color = Var1) ,stat = "identity") +
140    ggtitle("iris dataset")
141
142  print(g)
143  #Customize/specify colors
144  g + scale_color_manual(name = "Species", values = c("yellow","purple","pink"))
145
146  #Customize/specify fills
147  g + scale_fill_manual(name = "Species", values = c("#ff0000","#00ff00","#0000ff"))
```

Color brewer

- https://colorbrewer2.org/
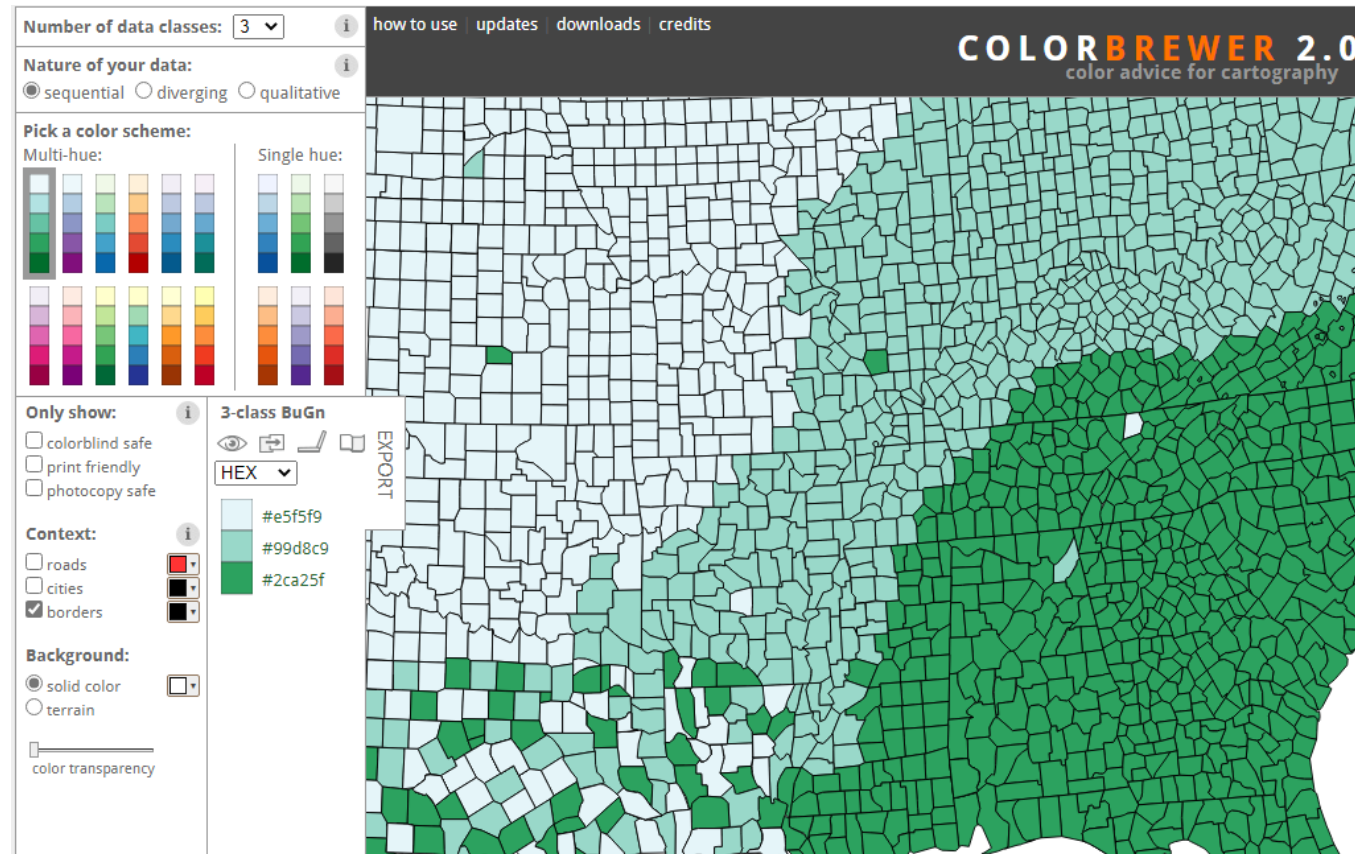
Color brewer

```
149  #### Color brewer ###
150  bdat <- expand.grid(x = 1:100, y = 1:100)
151
152  low <- exp(rnorm(n = 5000, mean = 1, sd = 0.5))
153  hi <- max(low) - low
154  bdat$z <- sample(c(hi,low),replace = F)
155  truehist(bdat$z)
156
157  g <- ggplot()+
158    geom_tile(data = bdat, aes(x=x,y=y,fill = z)) #Heatmap and Raster plot
159  print(g)
160  g + scale_fill_distiller(type = "div", palette = "RdBu")
161  g + scale_fill_distiller(type = "div", palette = "RdGy")
162  g + scale_fill_distiller(palette = "Spectral")
163  g + scale_fill_distiller(type = "seq", palette = "YlOrBr")
```

Gradient color/fill

```
165 ▾ #### Gradient color/fill ####
166   g + scale_fill_gradient2()
167   bdat$zbar <- bdat$z - mean(bdat$z) # Set the mid point to '0'
168
169   g <- ggplot()+
170     geom_tile(data = bdat, aes(x=x,y=y,fill = zbar))
171   print(g)
172   g + scale_fill_gradient2()
173   g + scale_fill_gradientn(colours = terrain.colors(10))
174   print(terrain.colors(10))
175   g + scale_fill_gradient(low = "yellow", high = "red")
```

Theme!!!

- Customize 'static' elements of the plot

```
177  #### Theme ####
178  g
179  g + theme_bw() # Good for publication!
180  g + theme_void()
181  g + theme_minimal()
182  g + theme_light()
183  g + theme_dark()
184  g + theme_test()
185  g + theme_classic() # Good for publication!
```

Theme!!!

- Customize 'static' elements of the plot
- There are so many "elements" in the theme so please refer to "https://ggplot2.tidyverse.org/reference/theme.html"

```
187 ▾ #### Theme pt.2 ####
188    ?iris
189    g <- ggplot()+
190      geom_point(data = iris, aes(x = Sepal.Length, y = Petal.Length, col = Species)) +
191      theme_bw()
192    g + theme(panel.grid.minor = element_blank()) # Remove grid lines
193    g + theme(panel.grid.minor.x = element_blank()) # Remove grid lines
194    g + theme(panel.grid.minor.y = element_blank()) # Remove grid lines
195    g + theme(axis.title = element_text(family = "Comic Sans MS", size = 20)) # Customize fonts
196    g + theme(axis.text = element_text(angle = 90)) # Rotate axis texts
197    g + theme(legend.position =  "top") # Move the legend box to the top
```

Add other elements to ggplot2

Text annotations

```r
199 ▾  #### Text annotations ####
200    ?iris
201    g <- ggplot()+
202      geom_point(data = iris, aes(x = Sepal.Length, y = Petal.Length, col = Species)) +
203      theme_bw()
204
205    print(g)
206
207    g + annotate(geom="text", x=7, y=2, label="P-value = 0.023*",
208                 color="red")
209    lab <- expression(paste(italic("P"),"-value = 0.023*")) #Italicize "P"
210    g + annotate(geom="text", x=7, y=2, label=lab,
211                 color="red")
```

Lines and trends

```r
213 ▾ #### Lines and trends ####
214   ?iris
215   g <- ggplot()+
216     geom_point(data = iris, aes(x = Sepal.Length, y = Petal.Length, col = Species)) +
217     theme_bw()
218
219   g + geom_hline(yintercept = 2, col = "pink") # Horizontal line
220   g + geom_vline(xintercept = 5.5, lty = 2) # Vertical line
221   g + geom_abline(intercept = 0, slope = 1, lwd = 5) # Diagonal line
```

Lines and trends

```r
223 ▾ #### Lines and trends pt.2 ####
224   ndat <- data.frame(x = seq(1,10,0.1))
225   ndat$y <- pi*ndat$x - exp(1) + rnorm(n = nrow(ndat), mean = 1, sd = 5)
226
227   g <- ggplot(ndat, aes(x = x, y = y))+
228     geom_point() + theme_bw()
229   print(g)
230
231   g + geom_smooth(method = "lm") # Linear regression
232   g + geom_smooth(method = "lm", se = F) # No confident intervals
233   g + geom_smooth(method = "lm", formula = y ~ poly(x, 2)) # Quadratic model
234   g + geom_smooth(method = "lm", formula = y ~ poly(x, 3)) # Cubic model
235   g + geom_smooth(method = "loess") #Locally estimated scatterplot smoothing
236
237 ▾ f1 <- function(x){
238     y <- (0.0001*x^3) + (0.19*x^2) - 1.1
239     return(y)
240   }
241   g + geom_function(fun = f1) # Add curve of a function
```

Drawing

```
243  #### Drawing ####
244  g + geom_segment(aes(x = 1,y =10, xend = 2, yend =15), col = "orange") # Line
245  g + geom_curve(aes(x = 1,y =10, xend = 2, yend =15), col = "red", curvature = -1.2) # Curve
246  g + geom_rect(aes(xmin = 1,ymin =10, xmax =2, ymax =15), fill = NA, col = "blue") # Box/Rectangle
247
248  # Draw polygons
249  ap <- data.frame(x = c(1,2,3,1), y = c(30,20,25,30), id = rep("1",4))
250  g + geom_polygon(data = ap, aes(x = x, y = y, group = id), fill = "green", col = "black")
```

Practical 3

Practical 3

- Download COVID cases data from: https://data.go.th/dataset/8a956917-436d-4afd-a2d4-59e4dd8e906e/resource/be19a8ad-ab48-4081-b04a-8035b5b2b8d6/download/confirmed-cases.csv

- Plot monthly cases from 1/1/2020 – 31/12/2020 (announce_date)

- Show data from Bangkok, Chiang Mai, Khon Kaen, Songkhla (province_of_isolation)
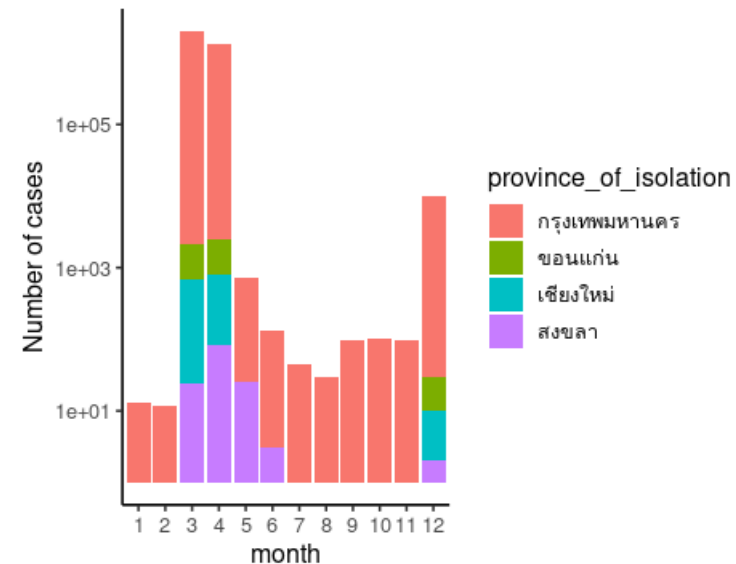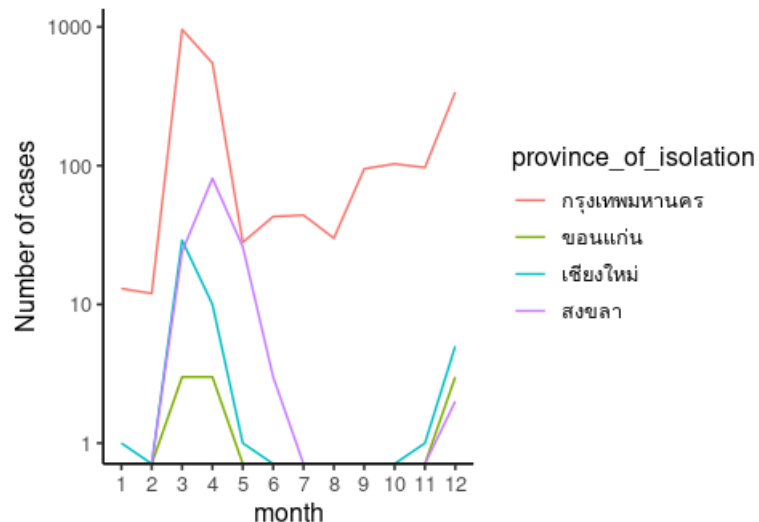
Multiple plots

Split plot with 'facet'

```r
252 ▾ #### Split plot with 'facet' ####
253   ?iris
254   g <- ggplot()+
255     geom_point(data = iris, aes(x = Sepal.Length, y = Petal.Length, col = Species)) +
256     theme_bw()
257   print(g)
258
259   g + facet_grid(rows = vars(Species))
260   g + facet_grid(cols = vars(Species))
261   g + facet_wrap(vars(Species), nrow = 2)
262   g + facet_wrap(vars(Species), ncol = 1)
```

Combine multiple plots

- ggarrange() in "ggpubr"

```
264  #### Combine multiple plots ####
265  # install.package("ggpubr")
266  library(ggpubr)
267  g1 <- ggplot()+
268    geom_point(data = iris, aes(x = Sepal.Length, y = Petal.Length, col = Species)) +
269    theme_bw()
270
271  rows <- sample(nrow(iris),size = 97, replace = F)
272  IRIS <- iris[rows,]
273  aIRIS <- data.frame(table(IRIS$Species))
274  print(aIRIS)
275  g2 <- ggplot(aIRIS)+
276    geom_bar(aes(x = Var1, y = Freq, fill = Var1), stat = "identity") +
277    theme_classic()
278
279  ndat <- data.frame(x = seq(1,10,0.1))
280  ndat$y <- pi*ndat$x - exp(1) + rnorm(n = nrow(ndat), mean = 1, sd = 5)
281  g3 <- ggplot(ndat, aes(x = x, y = y))+
282    geom_point() + theme_bw()
283
284  gall <- ggarrange(g1, g2, g3,
285            labels = c("A", "B", "C"),
286            ncol = 2, nrow = 2)
287  ggsave(filename = "Plots/Combine.jpeg",gall)
```

Resources

- Cheat sheets: https://github.com/rstudio/cheatsheets/blob/main/data-visualization-2.1.pdf

- Tutorial: http://r-statistics.co/Complete-Ggplot2-Tutorial-Part1-With-R-Code.html

- ggpubr (improve ggplot2 for publication): http://www.sthda.com/english/articles/24-ggpubr-publication-ready-plots/