Introduction to Programming in R

Centers for Disease Control and Prevention

Division of Vector-Borne Diseases

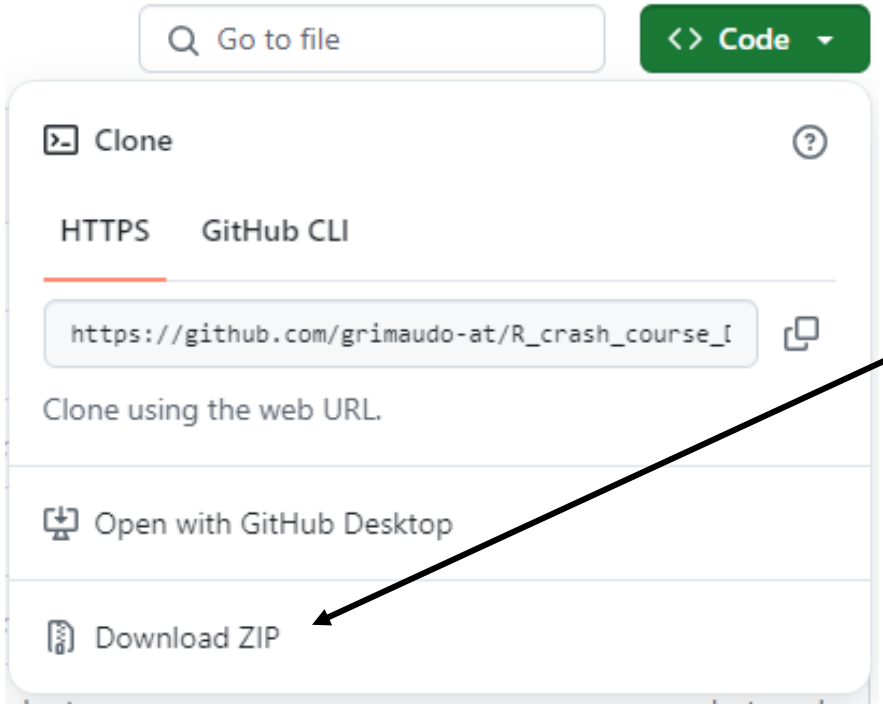
Carol Liu, PhD, MSc

Alex Grimaudo, PhD

Emma Jones, MS

Download workshop materials

Step 1: Navigate to this URL: github.com/grimaudo-at/R_crash_course_DVBD



Step 2: Click down-arrow in green box.

Step 3: Download the ZIP file and save on your device somewhere accessible.

Step 4: Open the R Project file to initialize your Rstudio session.

Agenda

1. Lecture – Introduction to R and Rstudio (~15-20 minutes)
2. Practical – Reading in, wrangling, exploring, and saving data (~40-45 minutes)
3. Lecture – Introduction to *ggplot2* and data visualization in R (~15-20 minutes)
4. Practical – Working with Date data and constructing various types of plots (~40-45 minutes)

There are **many** ways to do things in R.

What is R and Rstudio?

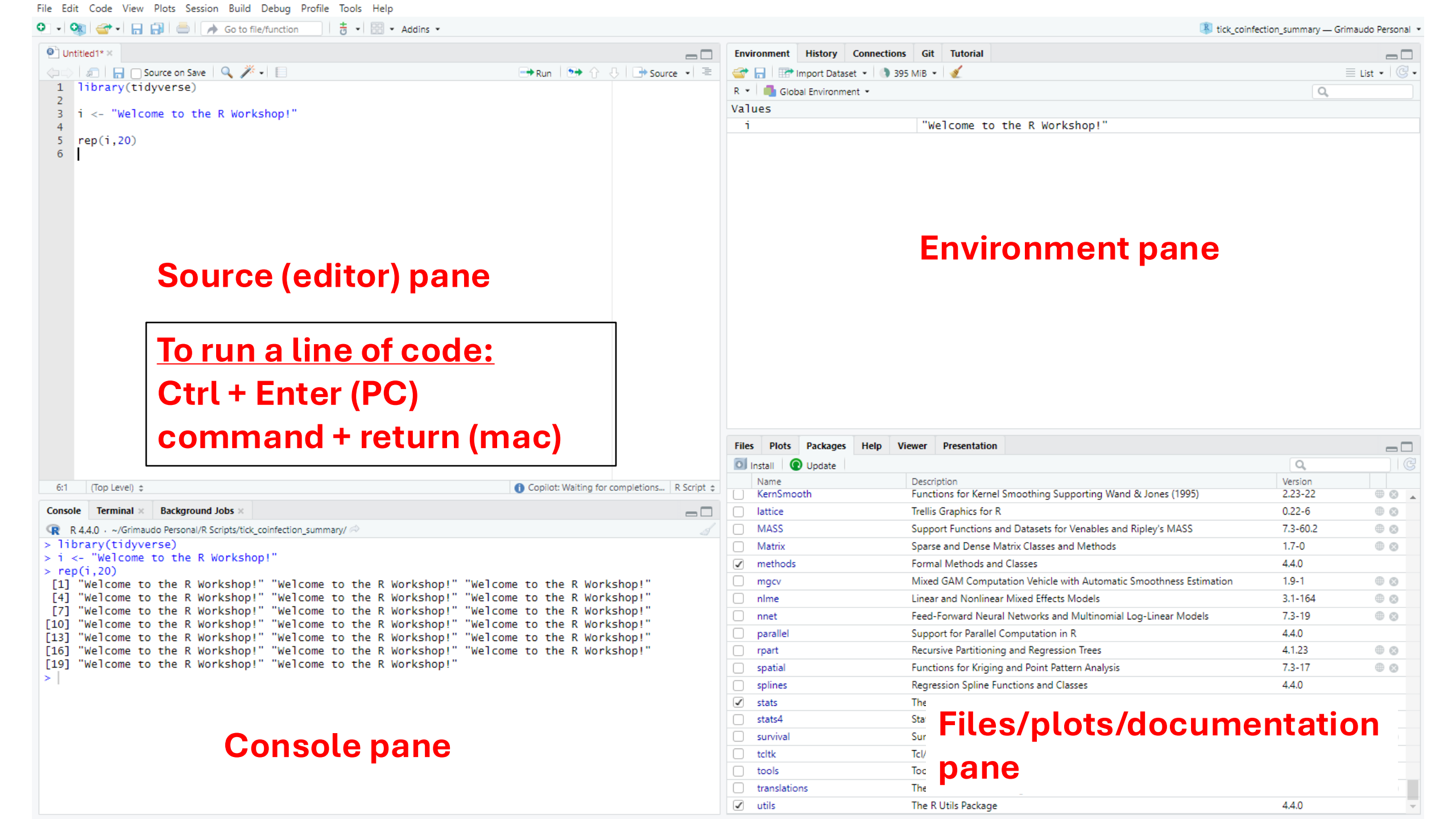
- R is a free, *open-source* programming language.
  - Similar to Python, C, JavaScript, etc., R is a programming language used to develop computer programs, used heavily in **data science and analytics**.
  - *Open-source:* The source code of R is publicly available, and anyone can change or contribute to it, creating a highly collaborative community.
    - Many of the tools you will use in R were created by various members of the community.
    - You can develop and share your own toolsets ("packages") for others to use!

What is R and Rstudio?

- Rstudio is a graphical user interface (GUI) developed by Posit and freely available to the public.
  - Offers a user-friendly interface for writing, debugging, and visualizing R code.

R is a language.

Rstudio is a tool to write code in R.


File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
tick_coinfection_summary — Grimaudo Personal
Untitled1*
Source on Save
Run
Source
1 library(tidyverse)
2
3 i <- "Welcome to the R Workshop!"
4
5 rep(i,20)
6
Source (editor) pane
To run a line of code:
Ctrl + Enter (PC)
command + return (mac)
6:1
(Top Level)
Copilot: Waiting for completions...
R Script
Environment
History
Connections
Git
Tutorial
Import Dataset
395 MiB
List
R
Global Environment
Values
i
"Welcome to the R Workshop!"
Environment pane
Console
Terminal
Background Jobs
R 4.4.0 · ~/Grimaudo Personal/R Scripts/tick_coinfection_summary/
> library(tidyverse)
> i <- "Welcome to the R Workshop!"
> rep(i,20)
[1] "Welcome to the R Workshop!" "Welcome to the R Workshop!" "Welcome to the R Workshop!"
[4] "Welcome to the R Workshop!" "Welcome to the R Workshop!" "Welcome to the R Workshop!"
[7] "Welcome to the R Workshop!" "Welcome to the R Workshop!" "Welcome to the R Workshop!"
[10] "Welcome to the R Workshop!" "Welcome to the R Workshop!" "Welcome to the R Workshop!"
[13] "Welcome to the R Workshop!" "Welcome to the R Workshop!" "Welcome to the R Workshop!"
[16] "Welcome to the R Workshop!" "Welcome to the R Workshop!" "Welcome to the R Workshop!"
[19] "Welcome to the R Workshop!" "Welcome to the R Workshop!"
>
Console pane
Files
Plots
Packages
Help
Viewer
Presentation
Install
Update
Name
Description
Version
KernSmooth Functions for Kernel Smoothing Supporting Wand & Jones (1995) 2.23-22
lattice Trellis Graphics for R 0.22-6
MASS Support Functions and Datasets for Venables and Ripley's MASS 7.3-60.2
Matrix Sparse and Dense Matrix Classes and Methods 1.7-0
methods Formal Methods and Classes 4.4.0
mgcv Mixed GAM Computation Vehicle with Automatic Smoothness Estimation 1.9-1
nlme Linear and Nonlinear Mixed Effects Models 3.1-164
nnet Feed-Forward Neural Networks and Multinomial Log-Linear Models 7.3-19
parallel Support for Parallel Computation in R 4.4.0
rpart Recursive Partitioning and Regression Trees 4.1.23
spatial Functions for Kriging and Point Pattern Analysis 7.3-17
splines Regression Spline Functions and Classes 4.4.0
stats The
stats4 Sta
survival Sur
tcltk Tcl/
tools Toc
translations The
utils The R Utils Package 4.4.0
Files/plots/documentation pane

Packages

- R **packages** are the "toolboxes" of R.
  - Contain sets of **functions**, the "tools" of R.
- Because R is open-source, *anybody* can develop a pack
  publish it on CRAN (Comprehensive R Archive Network)
  - Regardless of the type of analysis/wrangling/visualization you
    there's probably an R package for it.
    - Data visualization packages.
    - Frequentist and Bayesian statistical packages.
    - Geographic Information Systems (GIS) packages.
    - Agent-based modeling packages.
    - Packages for fun color combinations for data visualization.
- There are currently ~21.5 *thousand* R packages currently available on CRAN (Comprehensive R Archive Network).

Ram *et al.* (2023) CRAN

Tidyverse

- The *tidyverse* is a collection of some of the most common data cleaning, wrangling, and visualization packages.
  - We will be working with *tidyverse* packages today.

***tidyr*: data cleaning and tidying.**

***dplyr*: data wrangling and summarizing.**

***magrittr*: piping operators.**

***ggplot2*: data visualization.**

Objects in R

- An object is a data structure that holds a value or collection of values
  - Like a "container" or "storage unit" where information is stored
- Common classes of objects are:
  - Character: "a", "john", "positive", "puertorico"
  - Integer: 1,2,-100, 3245
  - Numeric: 5.4, 1, -10.3
  - Logical: TRUE/FALSE
  - Date: "2024-10-20"
- Common structures of objects

Structure	Dimension	Class
Vector	1-dimensional	Values of the same class
Matrices	2-dimensional	Values of the same class
Data frames	2-dimensional	Values of different classes
List	Collection of different types of data structures	

Common/Important syntax

- Base R uses syntax you will frequently encounter.
  - Packages (like those in *tidyverse*) will reduce the need for some of this syntax, **but it's important to recognize and be able to use it.**

+ - * / ^	Basic math operators	==	"Is equal to"
<- *or* =	Assignment operators	*!*=	"Is not equal to"
x[i]	Vector indexing	*!*	"Is not"
x[i,j]	Matrix/data frame indexing	*&*	"and"
$	Data frame column index	|	"or"
#	Annotation operator		

Functions

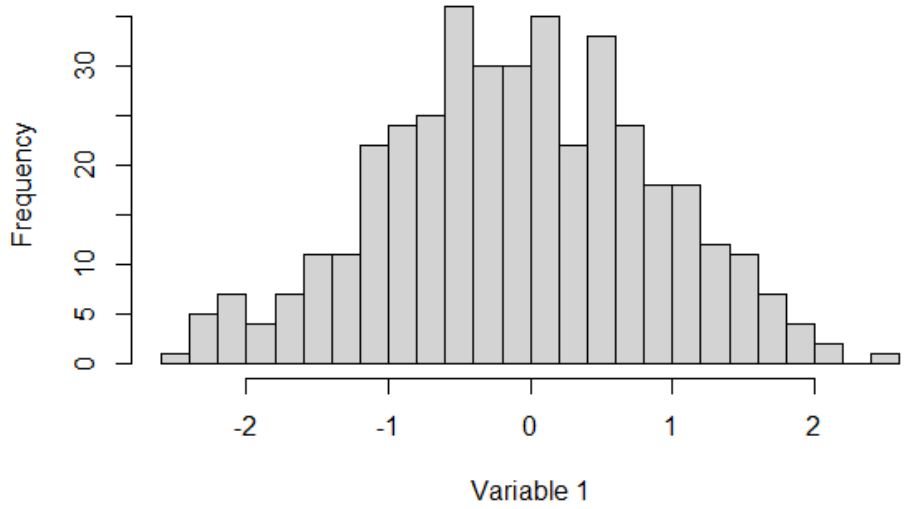
- The "tools" of R.
  - Each package contains a unique set of functions.

function(arg1, arg2, ...)

Function name Function arguments

Example

hist(x = my.data, breaks = 20, xlab = "Variable 1")



Case sensitivity

"PuertoRico" ≠ "puertorico"

"PuertoRico" ≠ "Puerto Rico"

"puertorico" ≠ "puertorico "

"puertorico" ≠ "puetrorico"

Pro tip: Use the *unique()* function to find these differences!

Data formats

- R can read in data in a variety of formats, including .xlsx, .csv, .Rds, .shp, .txt, .sas7bdat, etc.
  - May require specialized packages to read in data depending on the type.
- For reading in data from spreadsheets, .csv files are common in data science (more so than .xlsx). This is because:
  - Data is stored as plain text, making them smaller in size and human-readable.
  - Quicker to load and process because of lack of extraneous formatting options.
  - Cross-platform: can be opened by any operating system and variety of software types, whereas .xlsx files require Excel or other spreadsheet-viewing software.
  - Version control: changes made to plain-text .csv files is easier to track than more complex formatting changes made in .xlsx files.

Break for practical

Working with dates

- Dates in R can be tricky
- We need date class so we can do the following:
  - Filter and subset data based on dates
  - Calculations with dates
  - Plot time series data
- Use the function `as.Date()` to turn objects into date class
  - Example: `as.Date("2024-10-30")`
  - Format options:
    - `as.Date("30-10-2024", format = "%d-%m-%Y")`
    - `as.Date("30/10/2024", format = "%d/%m/%Y")`
- We can add and subtract date objects the same way as numeric objects
  - `as.Date("2024-10-30")+7`
  - `as.Date("2024-10-30")-as.Date("2024-10-20")`

Plotting in R using ggplot()

- Widely-used data visualization package in R
  - Part of tidyverse family
- Based on the idea of "Grammar of Graphics"
  - Structured and logical approach to creating plots
- Breaks data visualization into layers
  - Each layer represents a specific component of the data
  - Plots are described as layered building plots
- Each plot has at least the following components
  - **Data:** The dataset you are visualizing.
  - **Aesthetic mappings (aes):** Describes how data variables map to visual properties like position, color, size, etc.
  - **Geometries (geoms):** Defines the type of plot
    - Ex. Points (scatter plot), lines (line plot), bars (bar plot), etc.
  - Many other bells and whistles:
    - Ex. Controlling scales and legends, adding text, changing colors, facetting etc.

Plotting in R using ggplot()

- Example: Simple scatter plot

```
ggplot(data=simdat,                #data
       aes(x=age, y=wt_kg))+       #aes mapping of x and y axis
       geom_point()                #specifying scatter plot
```

Plotting in R using ggplot()

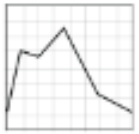
- Example: Scatter plot with title and axis-labels

```
ggplot(data=simdat,                                                    #data
       aes(x=age, y=wt_kg))+                                           #aes mapping of x and y axis
       geom_point()+                                                   #specifying scatter plot
       ggtitle("Scatter plot of age and weight among cases")+          #Plot title
       xlab("Age (years)")+                                            #X-axis label
       ylab("Weight (kg)")                                             #Y-axis label
```

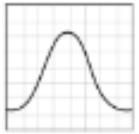
Examples of graphical elements in using ggplot()

One Variable (X)
- Continuous X
- Visualise distribution of X

geom_histogram()
- divide X into bins and count no. observation



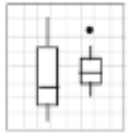
geom_freqpoly()
- display counts with lines
- able to overlay multiple distributions



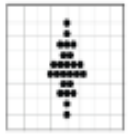
geom_density()
- smoothed version of the histogram

Two Variables (X,Y)
- Discrete X, continuous Y
- Visualise distribution of Y with respect to X

geom_col()
- heights of bars represent values



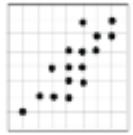
geom_boxplot()
- summarise distribution using median, hinges and whiskers



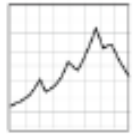
geom_jitter()
- adds jitter to prevent overplotting

geom_violin()
- mirrored density plot (smoothed distribution)

Two Variables (X,Y)
- Continuous X, continuous Y
- Visualise relationship between X and Y



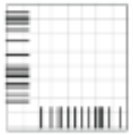
geom_point()
- scatterplot of X vs Y



geom_line()
- connect data points, ordered by X
- alt: geom_path()

geom_text()
- labelling data points

geom_smooth()
- add smoothed curve
- helps to see trends



geom_rug()
- supplement 2D plot with 1D distribution along X and Y

geom_area()
- can be stacked to see cumulative contribution

Visualising Errors and Uncertainties

geom_errorbar()
- uncertainty in continuous Y against discrete X

geom_ribbon()
- uncertainty in continuous Y against continuous X

Source: https://ouyanglab.com/covid19dataviz/ggplot2.html

Plotting in base R

- Use the `plot()` function
- Code below will give a scatter plot

```
plot(x=…, y=…,
     main=…,
     xlab=…, ylab=…)
```

← Specify the variables for x and y axis

← Add text for the plot title

← Add text for labels to the x and y axis

- You find these options (and many more) by checking the help file for the plot you want
- You can customize different aspects of the plot

Plotting in base R

- Histograms can be made using the `hist()` function
  - Introduced earlier

```
hist(x=…,
     main=…,
     xlab=…, ylab=…)
```

← Specify the variable for the histogram

← Add text for the plot title

← Add text for labels to the x and y axis

- You find these options (and many more) by checking the help file for the plot you want

Other resources for visualizations

- Future DVDB seminar on best practices led by Emma Jones

- The R Graph Gallery
  - https://r-graph-gallery.com/ggplot2-package.html
  - Gallery of graphs with corresponding code

- "ggmaps" package for mapping

- "ggpubr" package for arranging plots and saving into local directory

Additional resources in R

For self-learning/self-help

- Stack Overflow - Where Developers Learn, Share, & Build Careers
- The Epidemiologist R Handbook (epirhandbook.com)
- R for Data Science (2e) (hadley.nz)
- Ask Google/AI

Structured courses

- Coursera
- DataCamp

Break for practical