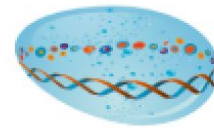

R WORKSHOP

Exploring the variability of the immune responses
using the Milieu Intérieur datasets

INTRODUCTION

Vincent Rouilly, TIL, Institut Pasteur Paris.



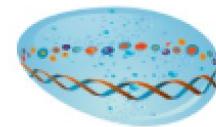
Milieu
Intérieur



June 2025, Tartu, Estonia.

R WORKSHOP TEAM

- **Darragh Duffy**, TIL lab PI, Institut Pasteur Paris.
- **Etienne Villain**, TIL lab, Institut Pasteur Paris.
- **Anthony Bertrand**, TIL lab, Institut Pasteur Paris.
- **Vincent Rouilly**, TIL Lab, Institut Pasteur Paris.

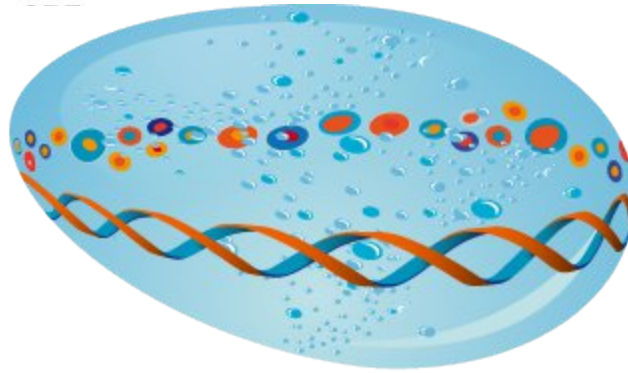


Milieu
Intérieur



The Milieu Intérieur Project

Hosted by Institute Pasteur on behalf of the Milieu Intérieur Consortium

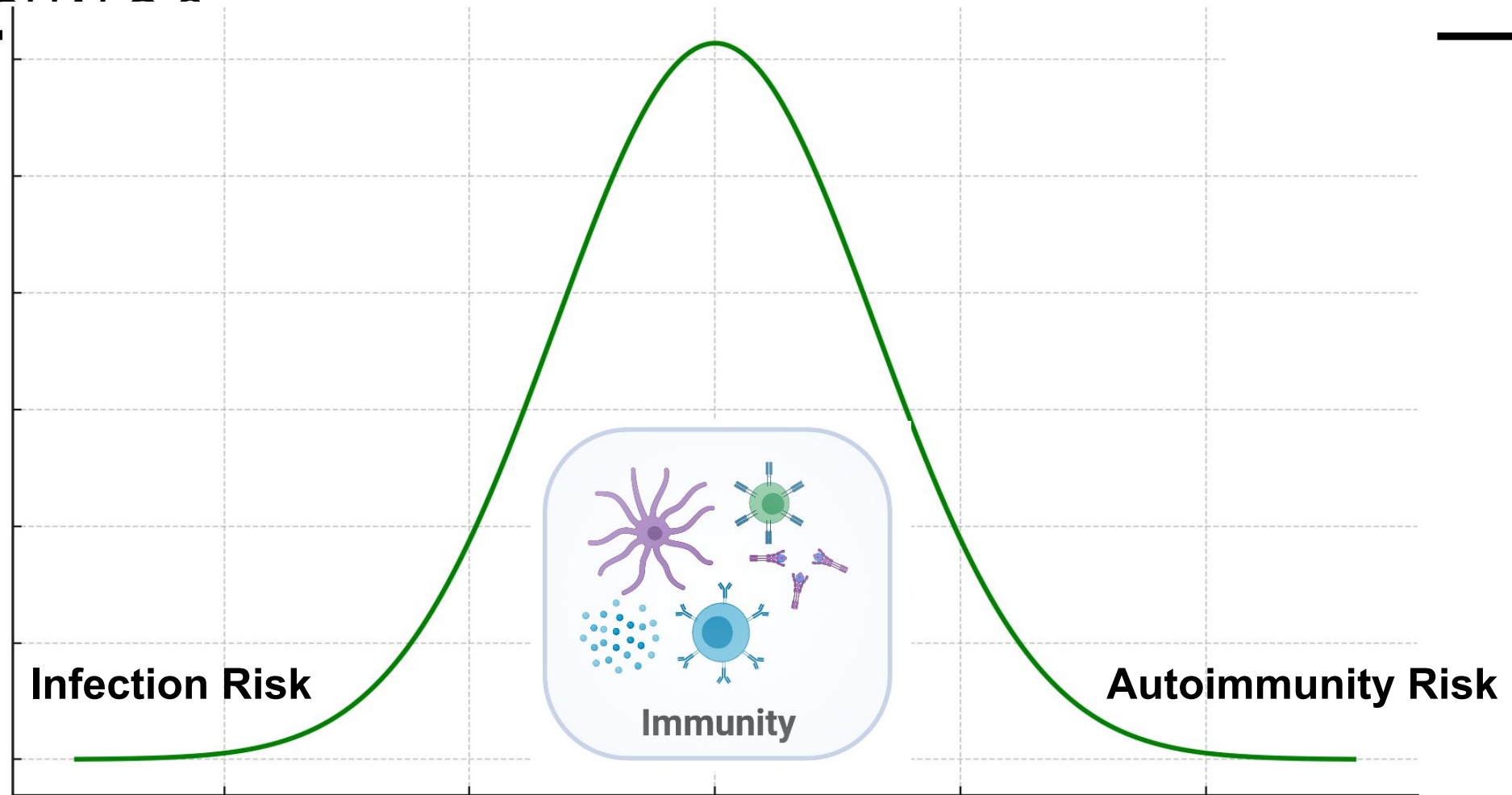


Milieu Intérieur

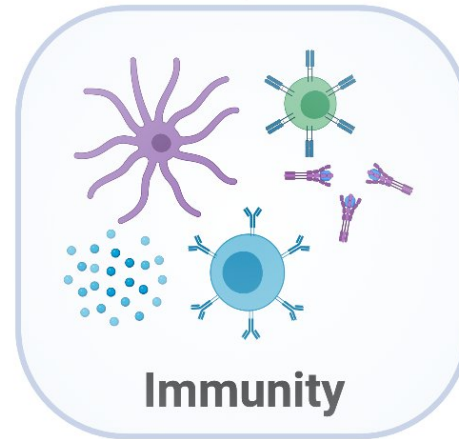
Vers une médecine personnalisée

Milieu Intérieur explores the **human immune system** by examining the **genetic** and **environmental** factors contributing to the **variability of healthy immune responses**.

WHY ARE IMMUNE RESPONSES DIFFERENT BETWEEN INDIVIDUALS?

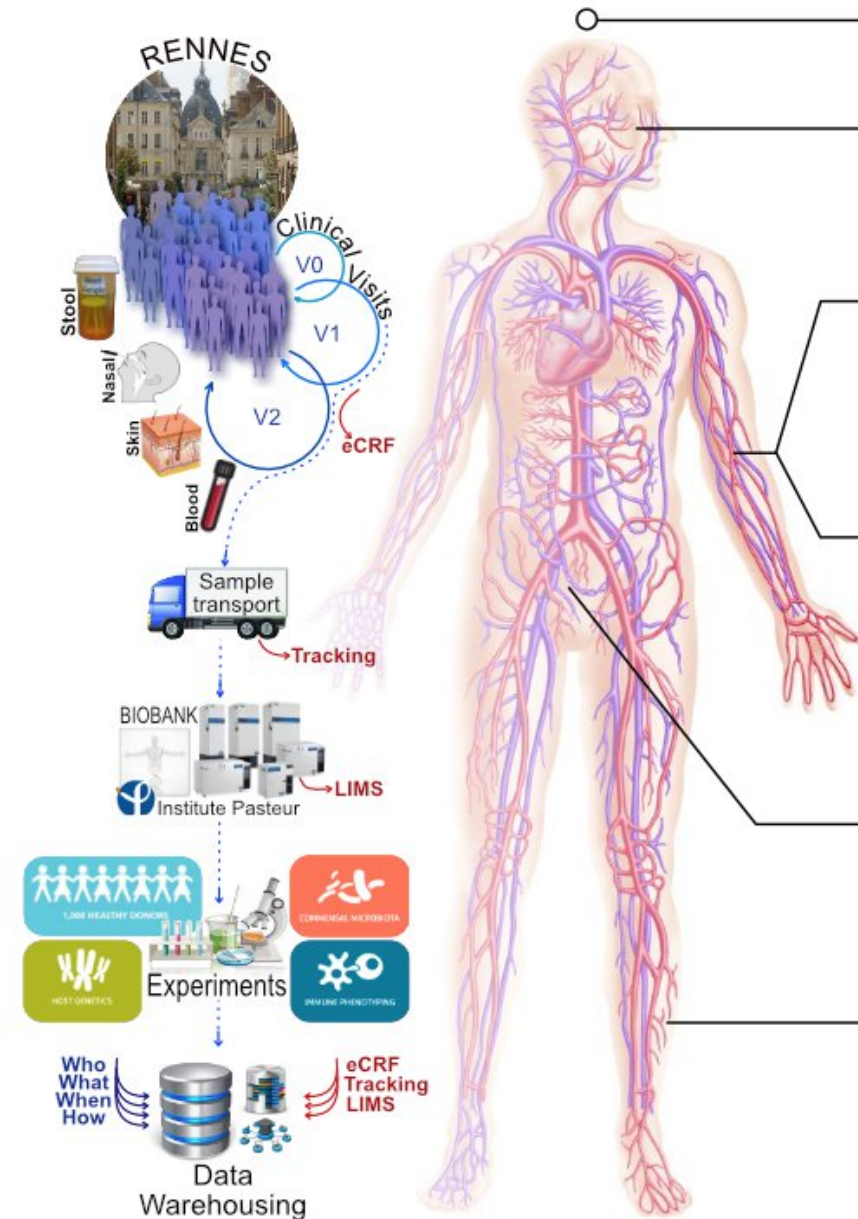


WHY ARE IMMUNE RESPONSES DIFFERENT BETWEEN INDIVIDUALS ?



The Milieu Intérieur Project

Hosted by Institute Pasteur on behalf of the Milieu Intérieur Consortium



Healthy cohort

1000 donors

50% Women – 50% Men
20yr < Age < 70yr

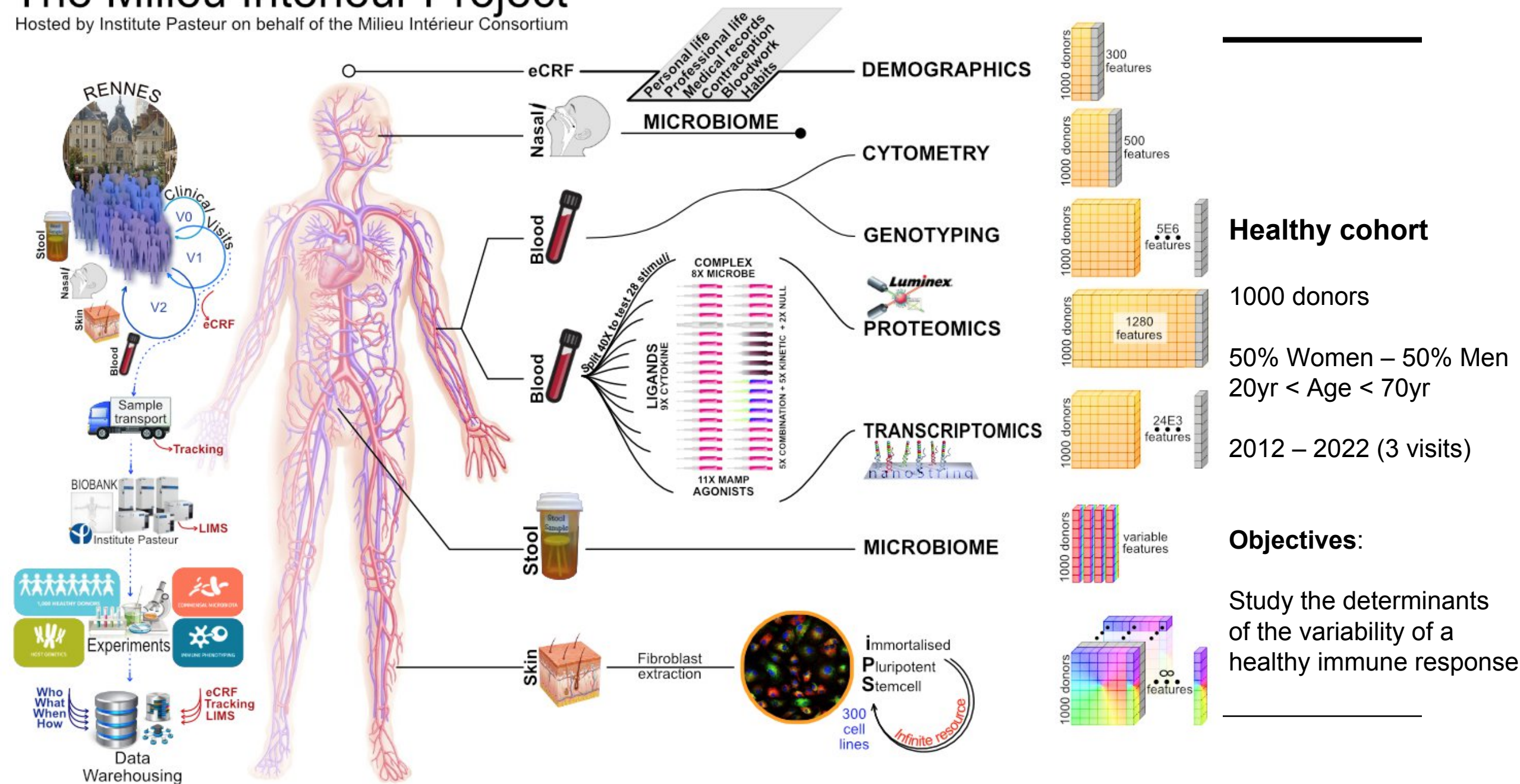
2012 – 2022 (3 visits)

Objectives:

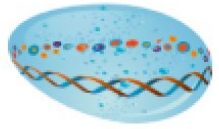
Study the determinants
of the variability of a
healthy immune response

The Milieu Intérieur Project

Hosted by Institute Pasteur on behalf of the Milieu Intérieur Consortium



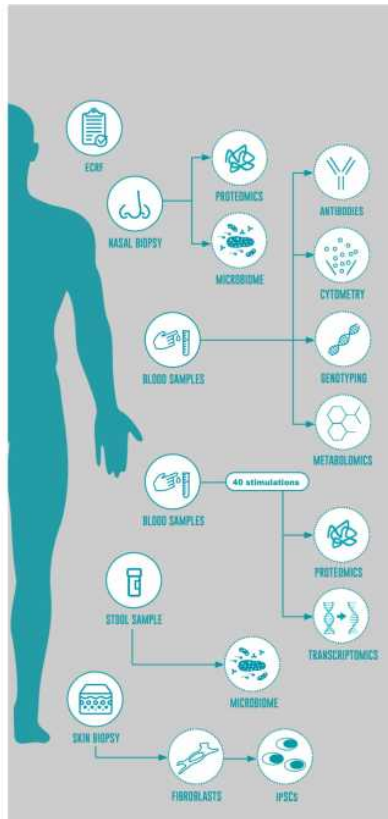
MILIEU INTÉRIEUR DATASETS FOR OUR WORKSHOP



Milieu
Intérieur



INSTITUT
PASTEUR



- eCRF variables
- TruCulture Luminex dataset
- TruCulture Nanostring dataset
- Flow Cytometry dataset

MI DATASETS: ECRF



- Age
- Sex
- Smoking
- CMV status
- Vaccination
- Physical Activity
- Sleep
- ...

MI DATASETS: TRUCULTURE LUMINEX

Article

Smoking changes adaptive immunity with persistent effects

<https://doi.org/10.1038/s41586-023-06968-8>

Received: 1 November 2022

Accepted: 13 December 2023

Published online: 14 February 2024

Open access

 Check for updates

Violaine Saint-André^{1,2}, Bruno Charbit³, Anne Biton², Vincent Rouilly⁴, Céline Possémé¹, Anthony Bertrand^{1,5}, Maxime Rotival⁶, Jacob Bergstedt^{6,7,8}, Etienne Patin⁶, Matthew L. Albert⁹, Lluís Quintana-Murci^{6,10}, Darragh Duffy^{1,3} & The Milieu Intérieur Consortium*

Individuals differ widely in their immune responses, with age, sex and genetic factors having major roles in this inherent variability^{1–6}. However, the variables that drive such differences in cytokine secretion—a crucial component of the host response to immune challenges—remain poorly defined. Here we investigated 136 variables and identified smoking, cytomegalovirus latent infection and body mass index as major contributors to variability in cytokine response, with effects of comparable magnitudes with age, sex and genetics. We find that smoking influences both innate and adaptive immune responses. Notably, its effect on innate responses is quickly lost after smoking cessation and is specifically associated with plasma levels of CEACAM6, whereas its effect on adaptive responses persists long after individuals quit smoking and is associated with epigenetic memory. This is supported by the association of the past smoking effect on cytokine responses with DNA methylation at specific signal *trans*-activators and regulators of metabolism. Our findings identify three novel variables associated with cytokine secretion variability and reveal roles for smoking in the short- and long-term regulation of immune responses. These results have potential clinical implications for the risk of developing infections, cancers or autoimmune diseases.

- 816 Donors
- 12 TruCulture Stimulations:
 - Null, TNFa, IL1b, IFNg, BCG, PolyIC, C.albicans, E.coli, Influenza, CD3+CD28, LPS, SEB
- 13-Plex:
 - CXCL5, CSF2, IFNg, IL1b, IL2, IL6, IL8, IL10, IL12p70, IL13, IL17, IL23, TNFa



Luminex
xMAP® Technology

MI DATASETS: TRUCULTURE NANOSTRING

PNAS

Distinctive roles of age, sex, and genetics in shaping transcriptional variation of human immune responses to microbial challenges

Barbara Piasecka^{a,b,1}, Darragh Duffy^{b,c,d,1}, Alejandra Urrutia^{c,d,e}, Hélène Quach^{a,f,g}, Etienne Patin^{a,f,g}, Céline Posseme^b, Jacob Bergstedt^{h,i}, Bruno Charbit^b, Vincent Rouilly^b, Cameron R. MacPherson^b, Milena Hasan^b, Benoit Albaud^j, David Gentien^j, Jacques Fellay^{k,l}, Matthew L. Albert^{b,c,d,e,2}, Lluís Quintana-Murci^{a,f,g,2,3}, and the Milieu Intérieur Consortium⁴

^aUnit of Human Evolutionary Genetics, Institut Pasteur, 75015 Paris, France; ^bCenter for Translational Research, Institut Pasteur, 75015 Paris, France; ^cLaboratory of Dendritic Cell Immunobiology, Department of Immunology, Institut Pasteur, 75015 Paris, France; ^dINSERM U1223, 75015 Paris, France; ^eDepartment of Cancer Immunology, Genentech Inc., San Francisco, CA 94080; ^fCNRS Unité de Recherche Associée 3012, 75015 Paris, France; ^gCenter of Bioinformatics, Biostatistics and Integrative Biology, Institut Pasteur, 75015 Paris, France; ^hDepartment of Automatic Control, Lund University, Lund SE-221, Sweden; ⁱInternational Group for Data Analysis, Institut Pasteur, 75015 Paris, France; ^jTranslational Research Department, Genomic Platform, Institut Curie, Paris Sciences et Lettres Research University, 75248 Paris, France; ^kSchool of Life Sciences, École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland; and ^lSwiss Institute of Bioinformatics, 1015 Lausanne, Switzerland

Edited by Jean-Laurent Casanova, The Rockefeller University, New York, NY, and approved December 1, 2017 (received for review August 22, 2017)

- 638 Donors
- 7 TruCulture Stimulations:
 - Null, BCG, C.albicans, E.coli, IAV, S.aureus, SEB
- 560 genes



MI DATASETS: FLOW CYTOMETRY

nature
immunology

RESOURCE

<https://doi.org/10.1038/s41590-018-0049-7>

Natural variation in the parameters of innate immune cells is preferentially driven by genetic factors

Etienne Patin^{1,2,3,26*}, Milena Hasan^{4,26}, Jacob Bergstedt^{5,6,26}, Vincent Rouilly^{3,4}, Valentina Libri⁴, Alejandra Urrutia^{4,7,8,9}, Cécile Alanio^{4,7,8}, Petar Scepanovic^{10,11}, Christian Hammer^{10,11}, Friederike Jönsson^{12,13}, Benoît Beitz⁴, Hélène Quach^{1,2,3}, Yoong Wearn Lim⁹, Julie Hunkapiller¹⁴, Magge Zepeda¹⁵, Cherie Green¹⁶, Barbara Piasecka^{1,2,3,4}, Claire Leloup¹⁴, Lars Rogge^{4,17}, François Huetz^{18,19}, Isabelle Peguillet^{20,21}, Olivier Lantz^{20,21,22,23}, Magnus Fontes^{6,24}, James P. Santo^{4,8,25}, Stéphanie Thomas^{4,7,8}, Jacques Fellay^{9,10}, Darragh Duffy^{4,7,8}, Lluís Quintana-Murci^{1,2,3,27}, Matthew L. Albert^{4,7,8,9,27*} and The Milieu Intérieur Consortium²⁸

- 816 Donors
- Immune cell populations



OUR LEARNING OBJECTIVES

- Explore the Variability of Healthy Immune Responses using the datasets from the Milieu Intérieur (MI) Project (milieuinterieur.fr).
 - Practice your R Skills
 - R/RStudio Good Practices
 - Tidy Data Principles and Tidyverse
 - Data Wrangling and Data Visualization
 - Foundational statistical methods (PCA, NHST, Linear Models)
 - Be equipped to continue your path in learning R and Statistics
-

R WORKSHOP PROGRAM

Times	Tuesday, June 17th.	
9am	Lecture: Welcome & Workshop Introduction Vincent Rouilly	
10am	Break	
10.15am	RStudio set-up & Student introduction	
12pm	Lunch	
1pm	Data Wrangling	
2.30pm	Break	
2.45pm	Data Wrangling / Data Visualization	
4pm	End of day	

R WORKSHOP PROGRAM

Times	Tuesday, June 17th.	Wednesday, June 18th.	
9am	Lecture: Welcome & Workshop Introduction Vincent Rouilly	Lecture: Milieu Intérieur Healthy Donor Studies Darragh Duffy	
10am	Break	Break	
10.15am	RStudio set-up & Student introduction	Data Visualization	
12pm	Lunch	Lunch	
1pm	Data Wrangling	PCA and Clustering	
2.30pm	Break	Break	
2.45pm	Data Wrangling / Data Visualization	PCA and Clustering	
4pm	End of day	End of day	

R WORKSHOP PROGRAM

Times	Tuesday, June 17th.	Wednesday, June 18th.	Thursday, June 19th	
9am	Lecture: Welcome & Workshop Introduction Vincent Rouilly	Lecture: Milieu Intérieur Healthy Donor Studies Darragh Duffy	Lecture: Comparing Immune Phenotypes Across Cohorts Etienne Villain	
10am	Break	Break	Break	
10.15am	RStudio set-up & Student introduction	Data Visualization	PCA and Clustering	
12pm	Lunch	Lunch	Lunch	
1pm	Data Wrangling	PCA and Clustering	Statistical tests (NHST)	
2.30pm	Break	Break	Break	
2.45pm	Data Wrangling / Data Visualization	PCA and Clustering	Statistical tests (NHST)	
4pm	End of day	End of day	End of day	

R WORKSHOP PROGRAM

Times	Tuesday, June 17th.	Wednesday, June 18th.	Thursday, June 19th	Friday, June 20th
9am	Lecture: Welcome & Workshop Introduction Vincent Rouilly	Lecture: Milieu Intérieur Healthy Donor Studies Darragh Duffy	Lecture: Comparing Immune Phenotypes Across Cohorts Etienne Villain	Lecture: Transcriptional gene regulation & SES effects Anthony Bertrand
10am	Break	Break	Break	Break
10.15am	RStudio set-up & Student introduction	Data Visualization	PCA and Clustering	Linear models
12pm	Lunch	Lunch	Lunch	Lunch
1pm	Data Wrangling	PCA and Clustering	Statistical tests (NHST)	Linear models
2.30pm	Break	Break	Break	Break
2.45pm	Data Wrangling / Data Visualization	PCA and Clustering	Statistical tests (NHST)	Linear models
4pm	End of day	End of day	End of day	End of workshop

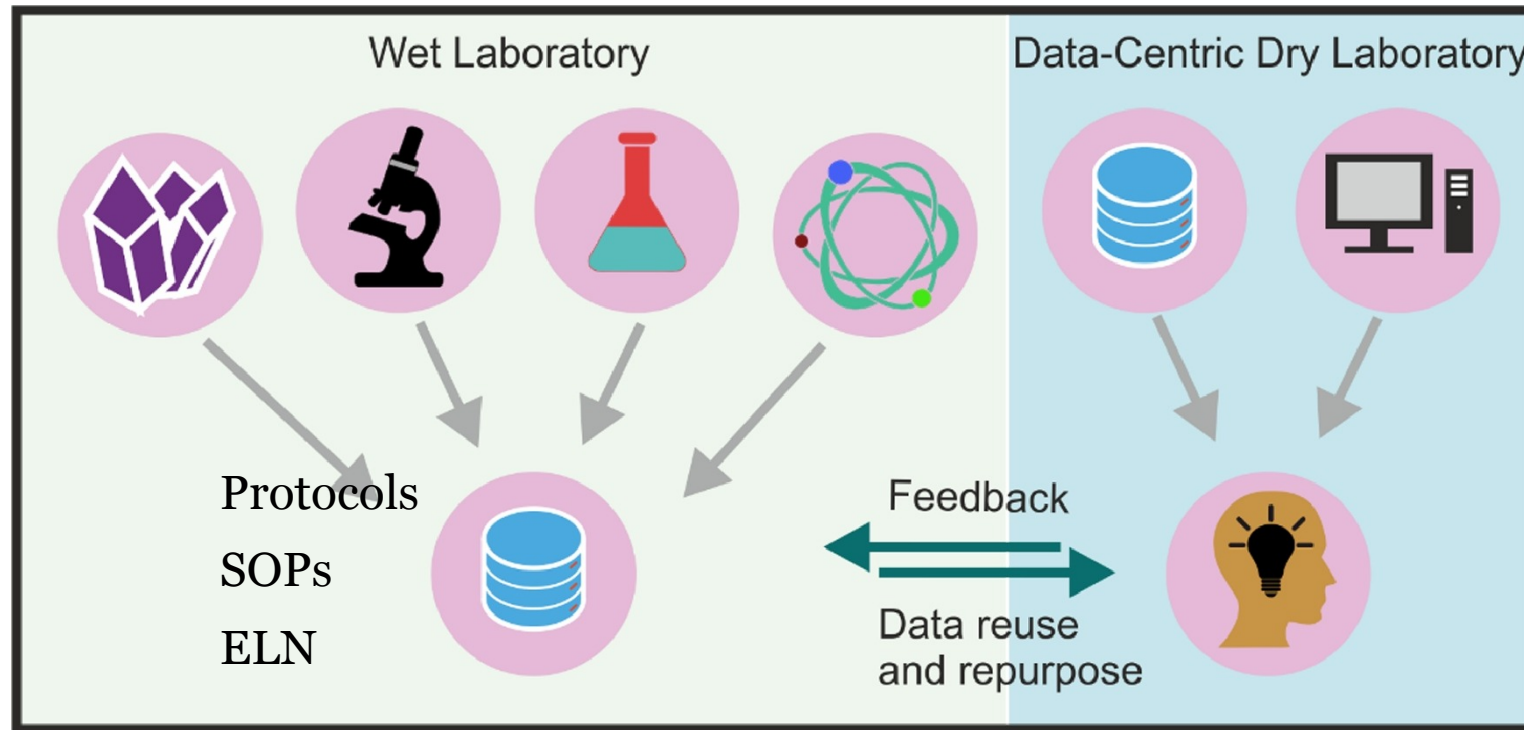
Questions?

Comments?

TIDY ANALYSIS PRINCIPLES IN R



WET LAB AND DRY LAB ARE BOTH EXPERIMENTS

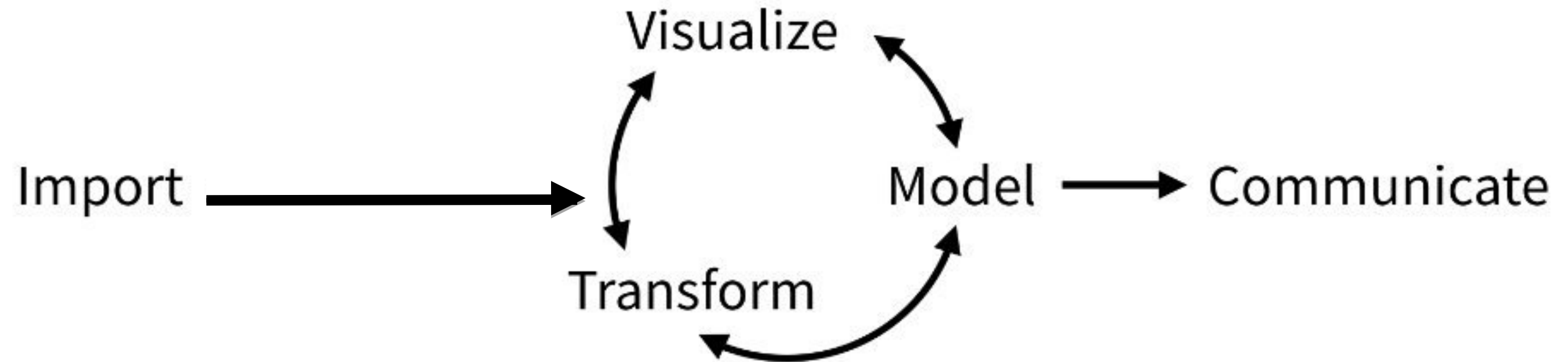


Many possible
computational methods
to try

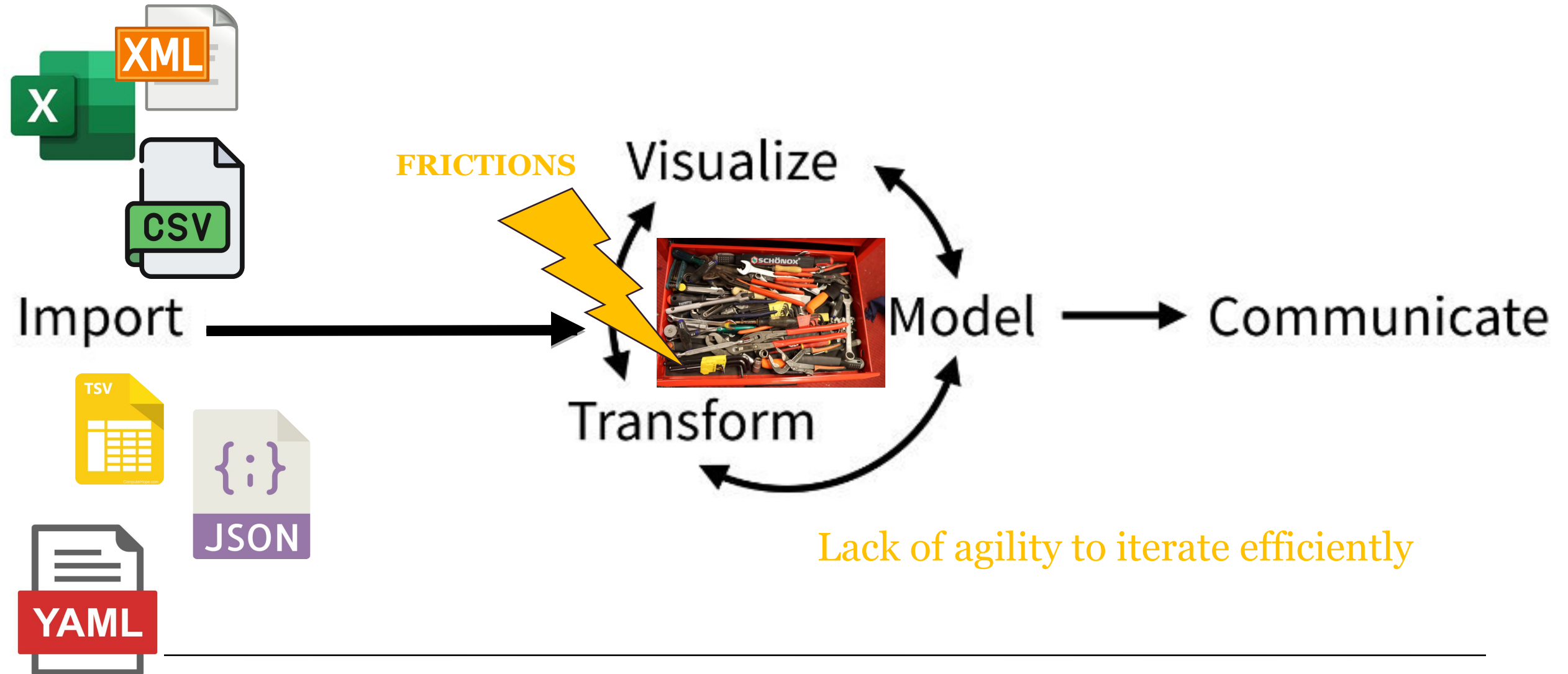
Both need to be **highly iterative**

Both need **Agility, Traceability** and **Reproducibility**

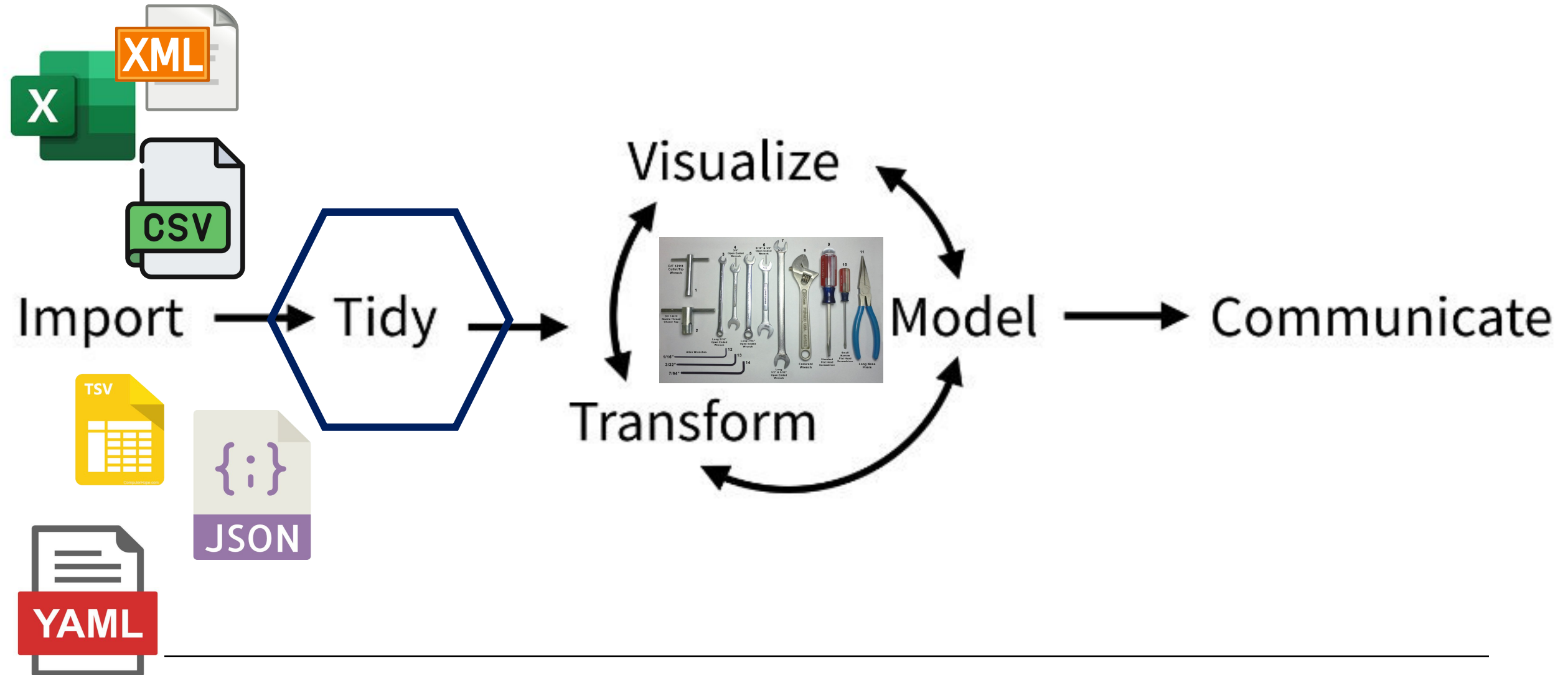
TYPICAL DATA ANALYSIS WORKFLOW



DIVERSE DATA REQUIRES DIVERSE CUSTOM TOOLS



TIDY DATA OPENS OPPORTUNITY FOR STANDARD TOOLS



TIDY PRINCIPLES



Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II.

<http://www.jstatsoft.org/>

Tidy Data

Hadley Wickham
RStudio

Abstract

A huge amount of effort is spent cleaning data to get it ready for analysis, but there has been little research on how to make data cleaning as easy and effective as possible. This paper tackles a small, but important, component of data cleaning: data tidying. Tidy datasets are easy to manipulate, model and visualise, and have a specific structure: each variable is a column, each observation is a row, and each type of observational unit is a table. This framework makes it easy to tidy messy datasets because only a small set of tools are needed to deal with a wide range of un-tidy datasets. This structure also makes it easier to develop tidy tools for data analysis, tools that both input and output tidy datasets. The advantages of a consistent data structure and matching tools are demonstrated with a case study free from mundane data manipulation chores.

Keywords: data cleaning, data tidying, relational databases, R.

“**TIDY DATA** is a standard way of mapping the meaning of a dataset to its structure.”

—HADLEY WICKHAM

In tidy data:

- each variable forms a column
- each observation forms a row
- each cell is a single measurement

each column a variable

id	name	color
1	floof	gray
2	max	black
3	cat	orange
4	donut	gray
5	merlin	black
6	panda	calico

each row an observation

Wickham, H. (2014). Tidy Data. Journal of Statistical Software 59 (10). DOI: 10.18637/jss.v059.i10

TIDY DATA

country	year	cases	population
Afghanistan	1999	7745	19987071
Afghanistan	2000	8266	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	127291272
China	2000	216766	128042583

variables

country	year	cases	population
Afghanistan	1999	7745	19987071
Afghanistan	2000	8266	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	127291272
China	2000	216766	128042583

observations

country	year	cases	population
Afghanistan	1999	7745	19987071
Afghanistan	2000	8266	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	127291272
China	2000	216766	128042583

values

DATA.FRAME OR TIBBLE AS A STANDARD CONTAINER

```
> flights
```

```
Source: local data frame [336,776 x 16]
```

	year	month	day	dep_time	dep_delay	arr_time	arr_delay	carrier	tailnum
	<int>	<int>	<int>	<int>	<dbl>	<int>	<dbl>	<chr>	<chr>
1	2013	1	1	517	2	830	11	UA	N14228
2	2013	1	1	533	4	850	20	UA	N24211
3	2013	1	1	542	2	923	33	AA	N619AA
4	2013	1	1	544	14	944	-18	B6	N804JB
5	2013	1	1	552	2	1017	-25	DL	N668DN
6	2013	1	1	554	0	1029	12	UA	N39463
7	2013	1	1	555	3	1035	19	B6	N516JB
8	2013	1	1	557	9	1045	-14	EV	N829AS
9	2013	1	1	559	8	1055	-8	B6	N593JB
10	2013	1	1	558	-2	753	8	AA	N3ALAA

```
Variables not shown: flight <int>, origin <chr>, dest <chr>, air_time  
<dbl>, distance <dbl>, hour <dbl>, minute <dbl>.
```

Column types

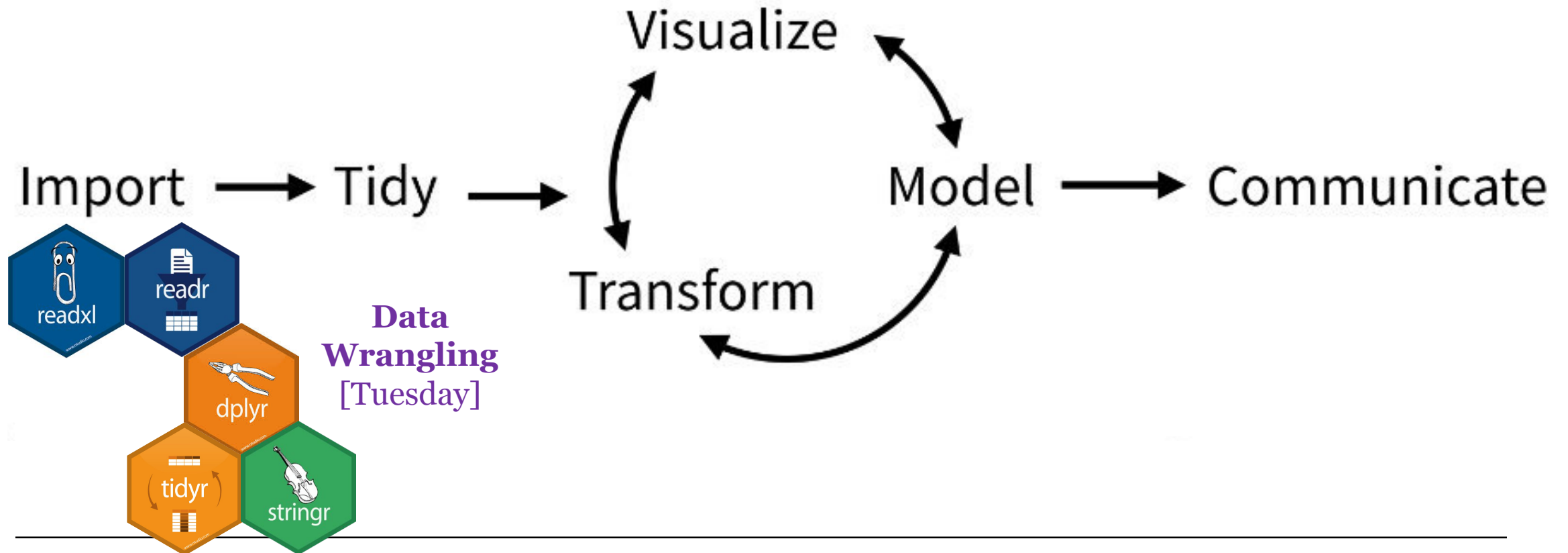
First 10 rows
printed by default
(Useful with large
data sets)

Columns that don't
fit to the screen are
not shown

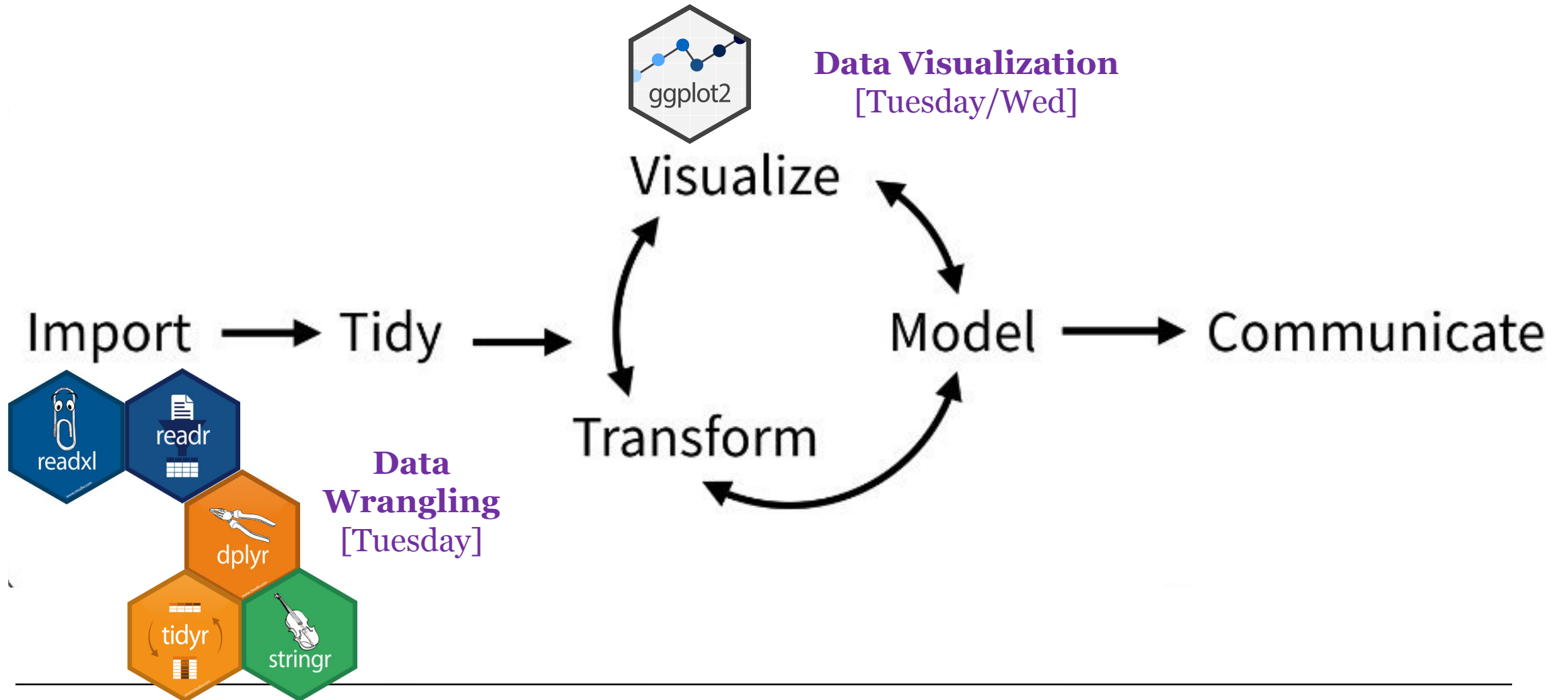


[Tibble vs Data.frame](#)

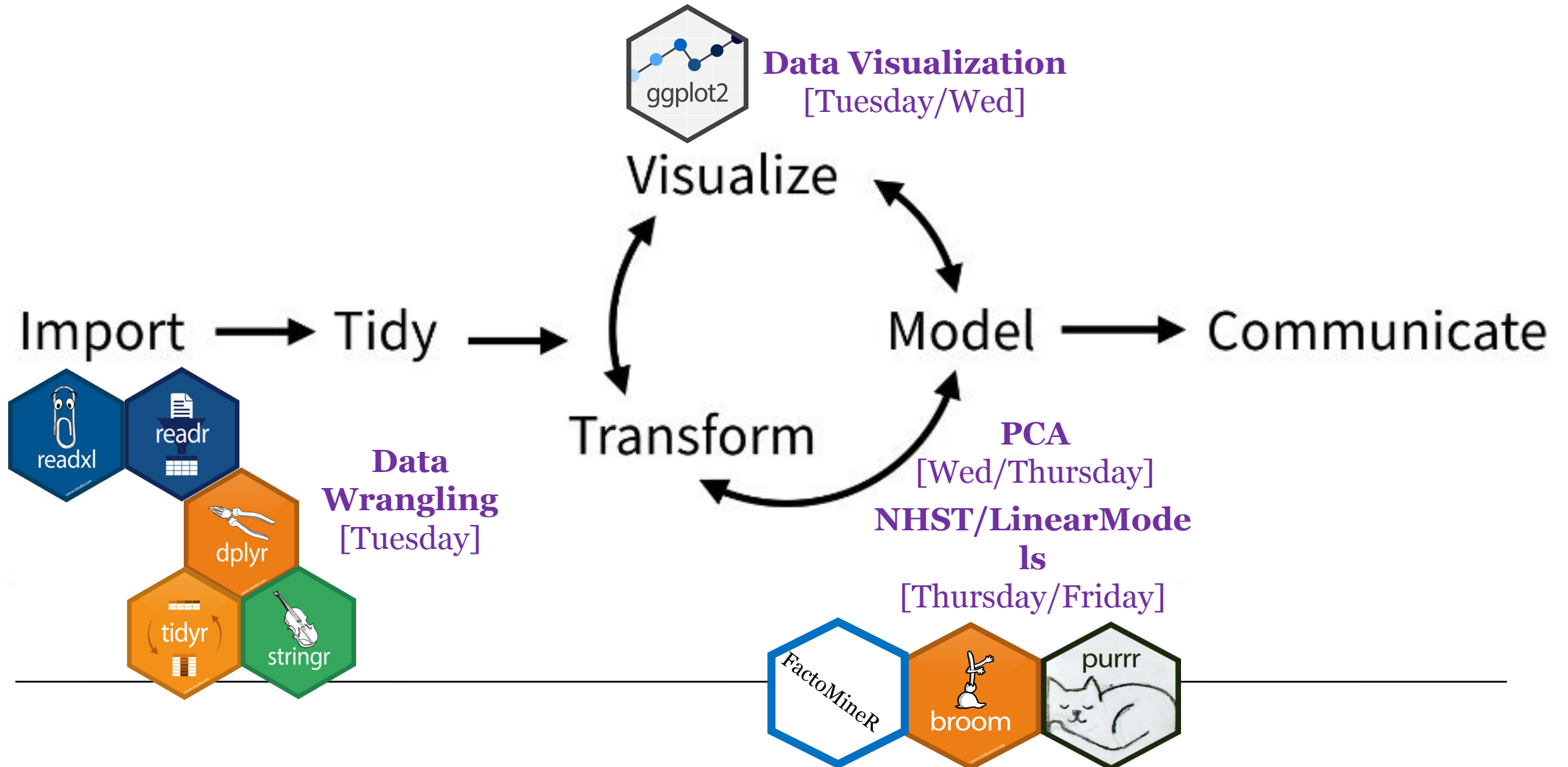
TIDY DATA ANALYSIS



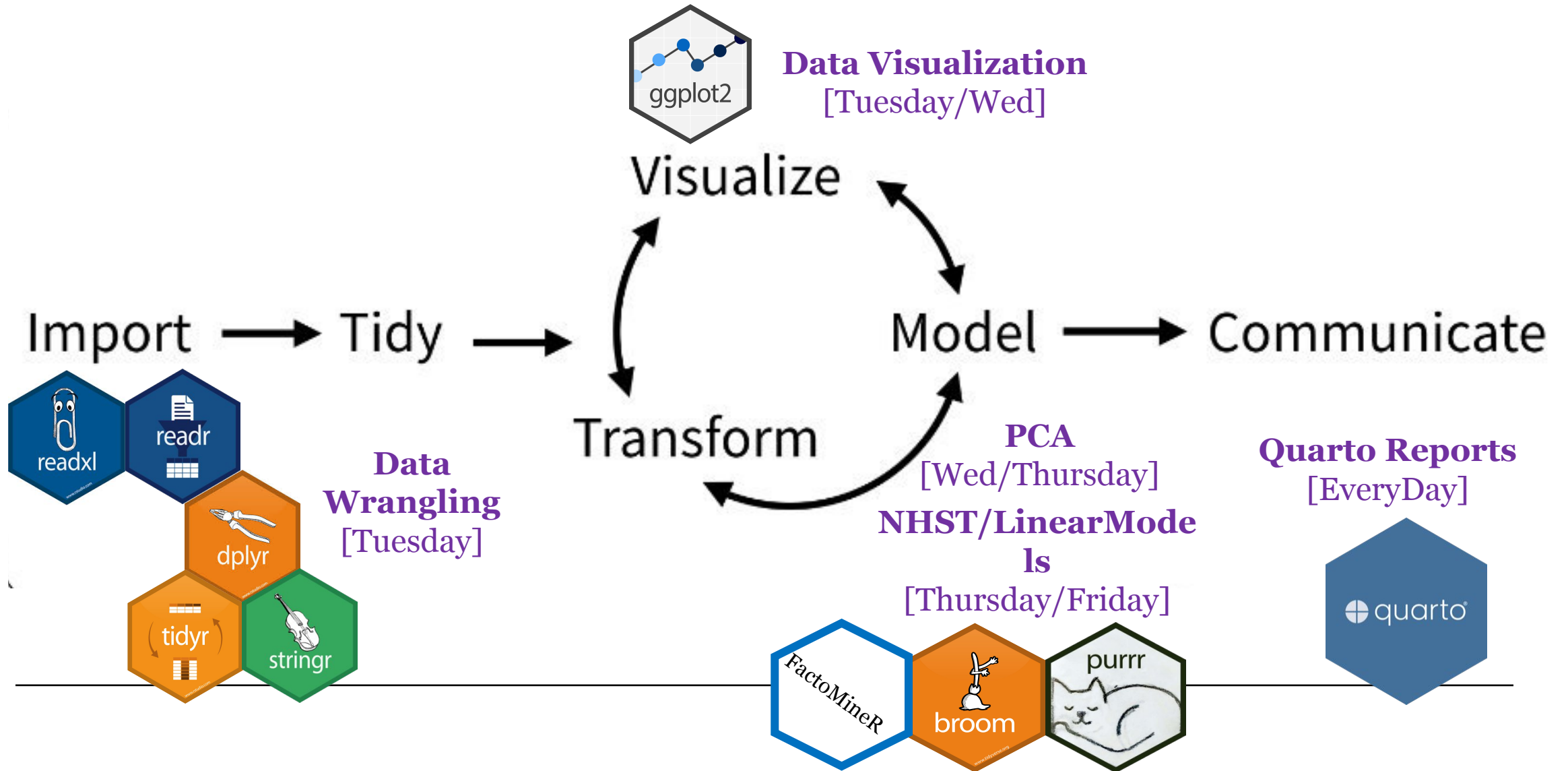
TIDY DATA ANALYSIS



TIDY DATA ANALYSIS



TIDY DATA ANALYSIS



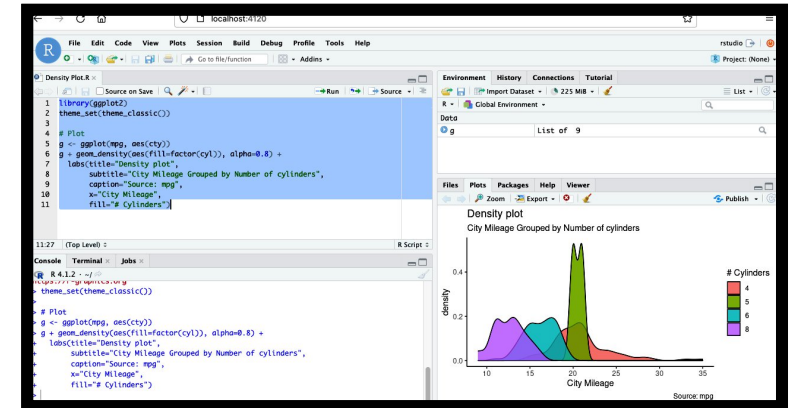
GENERAL WORKSHOP PRINCIPLES



Topic Introduction
Overview



Cookbook
R Recipes
Synthetic Data
Use case



Application
to MI Datasets

R CHOICES WE MADE FOR THIS WORKSHOP

- R Coding-related:
 - Favor Tidyverse packages
 - Extensive use of R Pipes: `|>` or `%>%`
 - Package prefixing e.g. `dplyr::filter(...)`
 - Use of Synthetic Data for R examples
 - Use of Quarto instead of RMarkdown
 - Use of RStudio Cloud instead of a local RStudio
-

ALWAYS FAVORS TYDIVERSE R PACKAGES



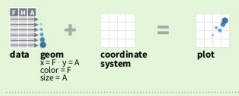
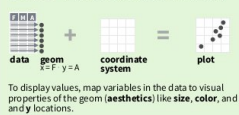
R CHEAT SHEETS

Data visualization with ggplot2 : : CHEATSHEET



Basics

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**, and **geoms** - visual marks that represent data points.



To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **angle** locations.

Complete the template below to build a graph.

```
ggplot(data = DATA) +  
  GEOM FUNCTION(mapping = aes(MAPPINGS),  
  stat = STAT, position = POSITION) +  
  COORDINATE FUNCTION +  
  THEME FUNCTION
```

ggplot(data = mpg, aes(x = city, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

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.

Aes Common aesthetic values.

color and **fill** - string ("red", "RRGGBB")

linetype - integer or string (0 = "blank", 1 = "solid", 2 = "dashed", 3 = "dotted", 4 = "dotdash", 5 = "longdash", 6 = "twodash")

size - integer (in mm for size of points and text)

linewidth - integer (in mm for widths of lines)

shape - integer/shape name or a single character ("a")



Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

GRAPHICAL PRIMITIVES

a <- ggplot(economics, aes(date, unemployment))
b <- ggplot(seals, aes(x = long, y = lat))

a + geom_blank() and **a + expand_limits()**
Ensure limits include values across all plots.

b + geom_curve() aes(yend = lat + 1, xend = long + 1, curvature = 1) - x, y, end, y, end, alpha, angle, color, curvature, linetype, size

a + geom_path() linetype = "butt", linejoin = "round", linemitre = 1
x, y, alpha, color, group, linetype, size

a + geom_polygon() aes(alpha = 50) - x, y, alpha, color, fill, group, subgroup, linetype, size

b + geom_rect() aes(xmin = long, ymin = lat, xmax = long + 1, ymax = lat + 1) - xmin, xmax, ymin, ymax, alpha, color, fill, linetype, size

a + geom_ribbon() aes(ymin = unemployment - 900, ymax = unemployment + 900) - x, y, alpha, color, fill, group, linetype, size

LINE SEGMENTS
common aesthetics: x, y, alpha, color, fill, group, linetype, size

b + geom_abline() aes(slope = 1, intercept = 0) - x, y, alpha, color, fill, group, linetype, size

b + geom_vline() aes(x = 0) - x, y, alpha, color, fill, group, linetype, size

b + geom_hline() aes(y = 0) - x, y, alpha, color, fill, group, linetype, size

b + geom_segment() aes(x1 = 0, x2 = 1, y1 = 0, y2 = 1) - x, y, alpha, color, fill, group, linetype, size

b + geom_spoke() aes(angle = 0) - x, y, alpha, color, fill, group, linetype, size

ONE VARIABLE CONTINUOUS
c <- ggplot(mpg, aes(hwy))

c + geom_area() stat = "area", aes(x, y, alpha, color, fill)
c + geom_density() stat = "density", aes(x, y, alpha, color, fill)
c + geom_dotplot() stat = "dotplot", aes(x, y, alpha, color, fill)
c + geom_freqpoly() stat = "frequency", aes(x, y, alpha, color, fill)
c + geom_histogram() stat = "histogram", aes(x, y, alpha, color, fill)
c2 + geom_qq() aes(x, y, alpha, color, fill)

discrete
d <- ggplot(mpg, aes(class))

d + geom_bar() aes(x, y, alpha, color, fill)

Data transformation with dplyr : : CHEATSHEET



dplyr functions work with pipes and expect tidy data. In tidy data:

Each variable is in its own column. Each observation, or case, is in its own row. x |> f(y) becomes f(x, y)

Summarize Cases

Apply **summary functions** to columns to create a new table of summary statistics. Summary functions take vectors as input and return one value (see back).

summarize(data, ...) Compute table of summaries. mtcars > summarise(avg = mean(mpg))

count(data, ..., wt = NULL, sort = FALSE, name = NULL) Count number of rows in each group defined by the variables in ... Also **tally()**, **add_count()**, **add_tally()**. mtcars > count(cyl)

Group Cases

Use **group_by**(data, ..., add = FALSE, drop = TRUE) to create a "grouped" copy of a table grouped by columns in ... dplyr functions will manipulate each "group" separately and combine the results.

mtcars > group_by(cyl) > summarise(avg = mean(mpg))

Use rowwise(data, ...) to group data into individual rows. dplyr functions will compute results for each row. Also apply functions to list-columns. See tidy cheat sheet for list-column workflow.

starwars > rowwise() > mutate(film_count = length(films))

ungroup(x, ...) Returns ungrouped copy of table. s_mtcars <- mtcars > group_by(cyl) > ungroup(s_mtcars)

Manipulate Cases

EXTRACT CASES
Row functions return a subset of rows as a new table.

filter(data, ...) preserve = FALSE Extract rows that meet logical criteria. mtcars > filter(mpg > 20)

distinct(data, ...) keep_all = FALSE Remove rows with duplicate values. mtcars > distinct(gear)

slice(data, ...) preserve = FALSE Select rows by position. mtcars > slice(10:15)

slice_sample(data, ..., n, prop, weight_by = NULL, replace = FALSE) Randomly select rows. Use n to select a number of rows and prop to select a fraction of rows. mtcars > slice_sample(n = 5, replace = TRUE)

slice_min(data, order_by, ..., n, prop, with_ties = TRUE) and **slice_max**(data, order_by, ..., n, prop, with_ties = TRUE) Select rows with the lowest and highest values. mtcars > slice_min(mpg, cyl, after = last_col())

slice_head(data, ..., n, prop) and **slice_tail**(data, ..., n, prop) Select the first or last rows. mtcars > slice_head(n = 5)

Logical and boolean operators to use with filter()

== **<** **<=** **is.na()** **%in%** **!is.na()** **!** **&** **xor()**

See **7basics:Logic and Comparison** for help.

ARRANGE CASES

arrange(data, ..., by_group = FALSE) Order rows by values of a column or columns (low to high), use with **desc()** to order from high to low. mtcars > arrange(mpg)

ADD CASES

add_row(data, ..., before = NULL, after = NULL) Add one or more rows to a table. cars > add_row(speed = 1, dist = 1)

Manipulate Variables

EXTRACT VARIABLES
Column functions return a set of columns as a new vector or table.

pull(data, var = 1, name = NULL, ...) Extract column values as a vector, by name or index. mtcars > pull(wt)

select(data, ...) Extract columns as a table. mtcars > select(mpg, wt)

relocate(data, ..., before = NULL, after = NULL) Move columns to new position. mtcars > relocate(mpg, cyl, after = last_col())

Use these helpers with select() and across()
e.g. mtcars > select(mpg, cyl)

contains(match), **num_range**(prefix, range), **ends_with**(match), **all_of**(v1, ..., vars), **matches**(match), **everything()**

MANIPULATE MULTIPLE VARIABLES AT ONCE
df <- tibble(x_1 = c(1, 2), x_2 = c(3, 4), y = c(4, 5))

c_across(cols) Compute across columns in row-wise data. df > summarise(across(everything(), mean))

MAKE NEW VARIABLES

Apply **vectorized functions** to columns. Vectorized functions take vectors as input and return vectors of the same length as output (see back).

mutate(data, ..., keep = "all", before = NULL, after = NULL) Compute new column(s). Also **add_column()**. mtcars > mutate(gpm = 1 / mpg)

rename(data, ...) Rename columns. Use **rename_with()** to rename with a function. mtcars > rename(miles_per_gallon = mpg)

Data tidying with tidyr : : CHEATSHEET



Tidy data is a way to organize tabular data in a consistent data structure across packages.

A table is tidy if:

Each variable is in its own column. Each observation, or case, is in its own row.

Access variables as vectors. Preserve cases in vectorized operations.

Reshape Data

Pivot data to reorganize values into a new layout.

Table 4a

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 1999 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

country year cases
A 1999 0.7K
B 1999 37K
C 2000 213K

Expand Tables

Create new combinations of variables or identify implicit missing values (combinations of variables not present in the data).

expand(data, ...) Create a new tibble with all possible combinations of the values of the variables listed in ... Drop other variables. expand(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

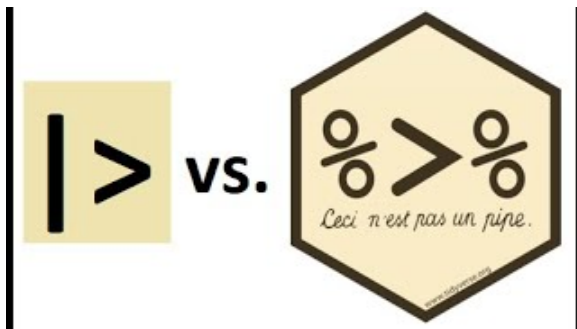
complete(data, ..., fill = list()) Add missing possible combinations of values of variables listed in ... Fill remaining variables with NA. complete(mtcars, cyl, gear, carb)

CC BY SA Posit Software, PBC · info@posit.co · posit.co · Learn more at tidyverse.org · HTML cheatsheets at pos.cheatsheets · tidyr 1.3.1 · tibble 3.2.1 · Updated: 2024-05

R PIPE APPROACH

Cognitive process:

1. Take the **ydat** dataset, *then*
2. **filter()** for genes in the leucine biosynthesis pathway, *then*
3. **group_by()** the limiting nutrient, *then*
4. **summarize()** to correlate rate and expression, *then*
5. **mutate()** to round *r* to two digits, *then*
6. **arrange()** by rounded correlation coefficients



USE OF SYNTHETIC DATASETS

SYNTHETIC DATA
GENERATION



Great for testing

No confidentiality issue

Reproducible

TRACEABILITY, REPRODUCIBILITY AND COMMUNICATION

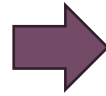
```
Console Terminal Background Jobs
R 4.3.0 - ~/Desktop/Intro-to-R/

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

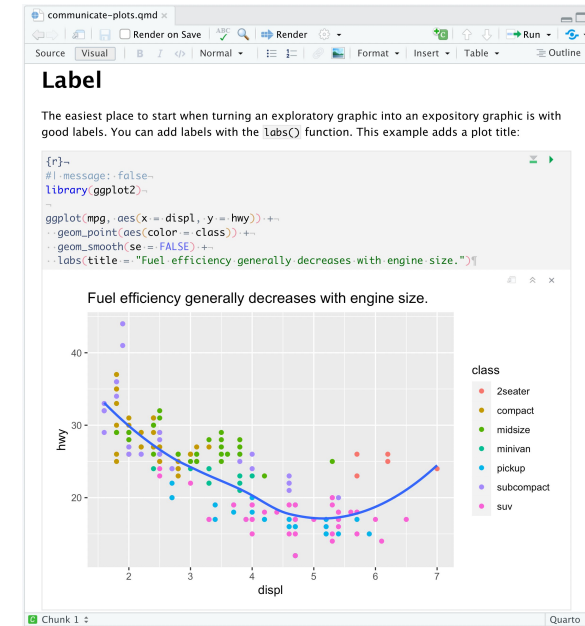
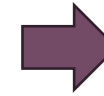
> 3 + 5
[1] 8
>
```

R Console



```
27 freq.reg <- lm(Calories ~ Duration*Weight, data=data)
28
29 mod <- lm(Calories ~ ., data=data)
30
31 # With AIC
32 modAIC <- MASS::stepAIC(mod, k = 2)
33 summary(modAIC)
34
35 modBIC <- MASS::stepAIC(mod, k = log(nrow(data)))
36 summary(modBIC)
37
38 modZero <- lm(Calories ~ 1, data = data)
39 mod_forward <- MASS::stepAIC(modZero, direction = "forward",
40 scope = list(lower = modZero, upper = mod), k = log(nrow(data)))
41
42 freq.reg <- lm(Calories ~ Duration*Weight*Age*Height*Heart_Rate*Body_Temp, data=data)
43 summary(freq.reg)
44 plot(freq.reg)
45 predict(freq.reg)
46 # plot(log(Volume), predict(freq.reg), pch=16)
47 predict(freq.reg, interval="confidence")
48 predict(freq.reg, interval="prediction")
49 confint(freq.reg, level=.95)
50 #
51 # Bayesian method
52 #
53 library(MCMCglmm)
54 bayes.reg <- MCMCglmm(Calories ~ Duration, data=data)
55 plot(bayes.reg)
```

R Script



R Report

QUARTO REPORTS

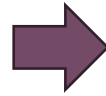
```
Console Terminal Background Jobs
R 4.3.0 - ~/Desktop/Intro-to-R/

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

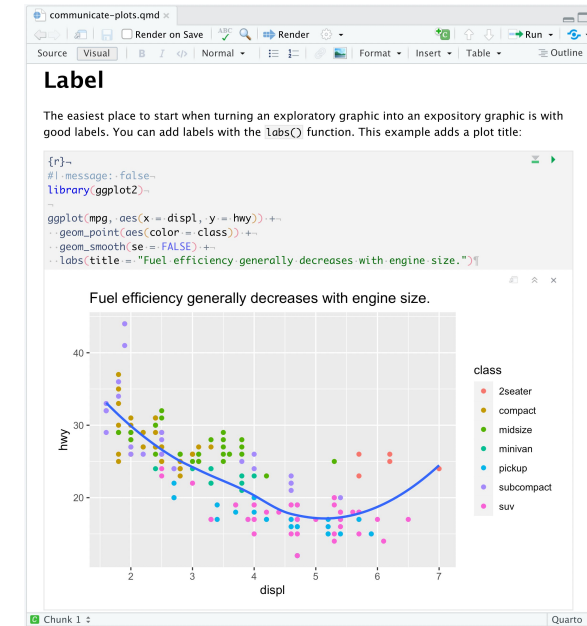
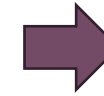
> 3 + 5
[1] 8
>
```

R Console

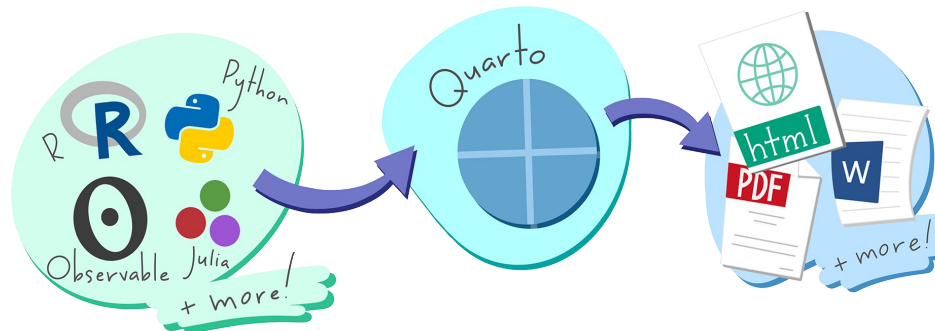


```
27 freq.reg <- lm(Calories ~ Duration-Weight, data=data)
28
29 mod <- lm(Calories ~ ., data=data)
30
31 # With AIC
32 modAIC <- MASS::stepAIC(mod, k = 2)
33 summary(modAIC)
34
35 modBIC <- MASS::stepAIC(mod, k = log(nrow(data)))
36 summary(modBIC)
37
38 modZero <- lm(Calories ~ 1, data = data)
39 mod_forward = MASS::stepAIC(modZero, direction = "forward",
40                             scope = list(lower = modZero, upper = mod), k = log(nrow(data)))
41
42 freq.reg <- lm(Calories ~ Duration-Weight-Age-Height-Heart_Rate-Body_Temp, data=data)
43 summary(freq.reg)
44 plot(freq.reg)
45 predict(freq.reg)
46 # plot(log(Volume), predict(freq.reg), pch=16)
47 predict(freq.reg, interval="confidence")
48 predict(freq.reg, interval="prediction")
49 confint(freq.reg, level=.95)
50 #
51 # Bayesian method
52 #
53 library(MCMCglmm)
54 bayes.reg <- MCMCglmm(Calories ~ Duration, data=data)
55 plot(bayes.reg)
```

R Script



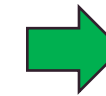
R Report



[Quarto Documentation](#)

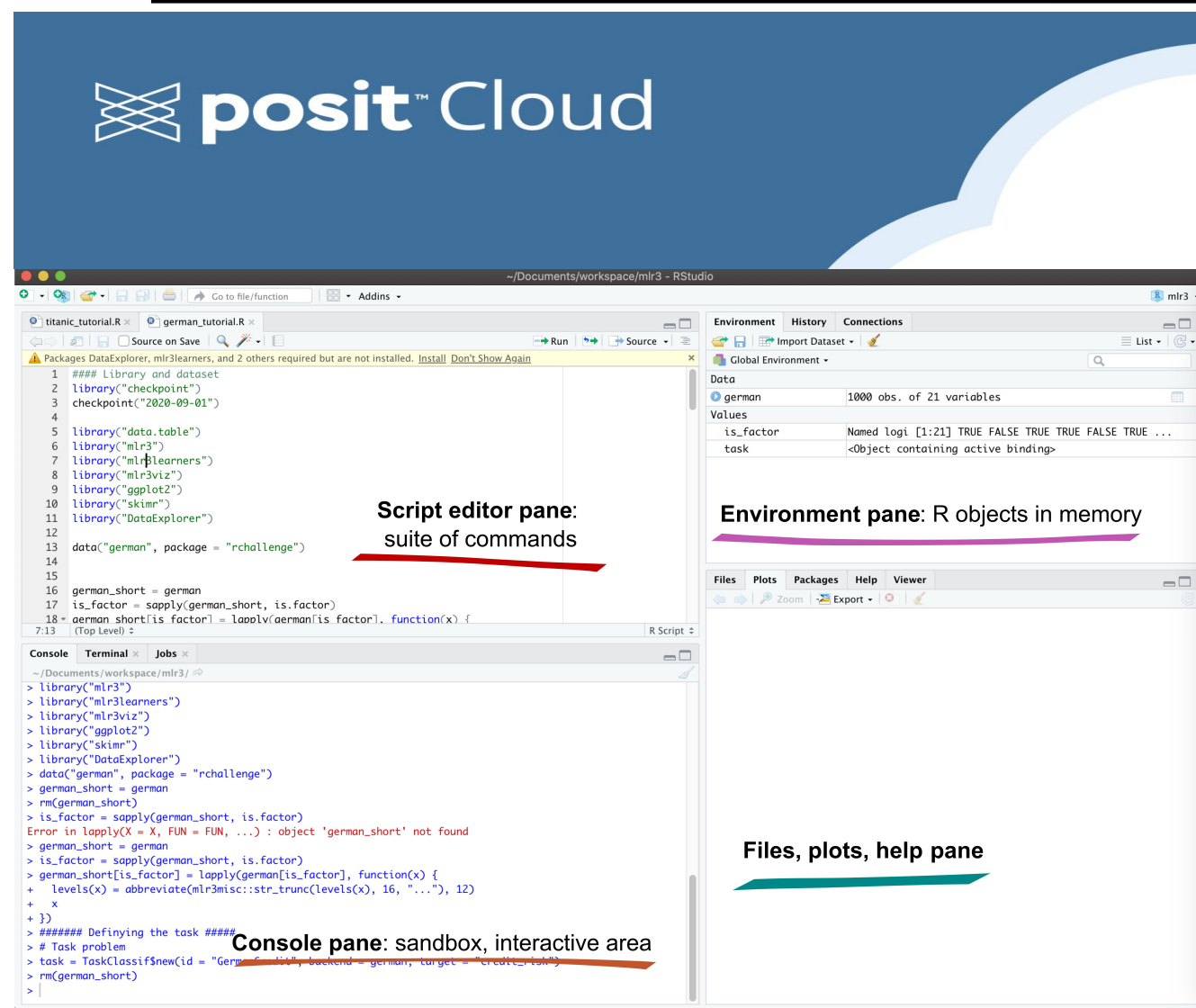


RMarkdown



Quarto

R ENVIRONMENT FOR THE WORKSHOP



- Web application
- Uniform R environment for all
- Use of a shared RStudio project template
 - Pre-installed libraries
 - Pre-installed files for workshop
- But you can still download the RStudio project to run locally

R: WHERE TO FIND HELP



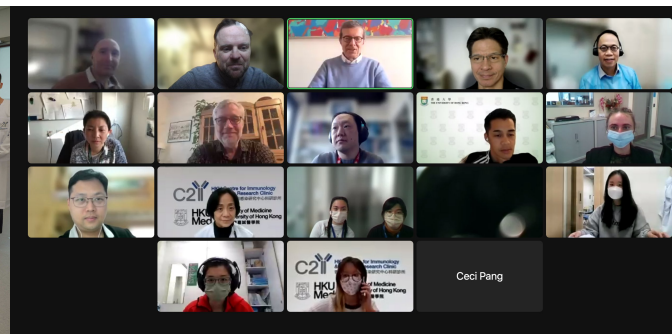
- Your Instructors for this week
- Within the group
- ?function in R console
- R Cheat Sheets
- StackOverflow / Google
- ...
- And ...

A NEW TOOL IN OUR TOOLBOX



- Have you already used it for R programming ?
 - To explain some code you found
 - To customize some existing code
 - To explain an error message and suggest a fix
 - ... other case ?
- What is your favorite prompt for R questions ?

“Art is I, Science is We” Claude Bernard 1865



Translational Immunology Unit

Darragh Duffy
Chloe Albert
Anthony Bertrand
Vincent Bondet
Tom Dott
Florian Dubois
Elizabeth Maloney
Thomas Moreau
Farah Rahal
Marie Robert
Vincent Rouilly
Jamie Sugrue
Violaine Saint-Andre
Divya Unni
Etienne Villain
MC Voungny
Nader Yatim

Human Evolutionary Genetics Unit

Yann Aquino
Etienne Patin
Maxime Rotival
Lluís Quintana-Murci

Innate Immunity Unit

James Di Santo
Jean Marc Doisne
Pedro Goncalves

CBUTechS

Slobodan Culina
Milena Hasan

DARRI

Isabelle Buckle
Omar Choa
Camille Tribout
Lea Vanni

C2i, HK

Wilson Ng
Rex Hung
Rosanna Fong
Raven Cheng
Cynthia Wong
Hebie Chen
Minnie Lo
Max Hui
Jeffrey Yu
Cheska Li
Anna Lee
Hilda On
Garrick Yip
Simon Muller
Amandine Schneider
Peter Chung
Christy Wong
Grace Wu
Theodora Luk
Jimmy Lai
Rachel Telford

Elina Li
Kylie Chang
Raina Mok
Hebronson Yung
Atilla ERISIR
Sonia YOUNAS
Ashley Li

HKU-Pasteur Research Pole, HKU/C2i

Roberto Bruzzzone
Malik Peiris
Leo Poon

The University of Hong Kong

Yu Lung Lau
Daniel Leung
Jaime Rosa Duque

FAMILY Cohort, HKU

Gabriel Leung
Michael Ni

Milieu Interieur Consortium
Lluís QUINTANA MURCI
Darragh DUFFY
Matthew AILBERT
Laurent ABEL (Necker)
Andres ALCOVER
Hughues ASCHARD
Philippe BOUSSO
Nollaig BOURKE (TCD)
Petter BRODIN (Karolinska)
Pierre BRUHNS
Nadeine CERF-BENSUSSAN (Imagine)

Ana CUMANO
Caroline DEMANGEL
Christophe D'ENFERT
Ludovic DERIANO
Marie-Agnes DILLIES
James DISANTO
Gérard EBERL
Jost ENNINGA

Jacques FELLAY (EFL)
Ivo GOMPERTS-BONECA
Milena HASAN
Magnus FONTES (Roche)

Gunilla KARLSSON HEDESTAM (Karolinska)
Serge HERCBERG (Université Paris)

Molly INGERSOLL
Rose ANNE KENNY (TCD)
Olivier LANTZ (Curie)
Frederique MICHEL
Hugo MOUQUET
Cliona O'FARRELLY (TCD)
Eteinne PATIN

Antonio RAUSELL (Imagine)
Lars ROGGE

Anavaj SAKUNTABHAI
Olivier SCHWARTZ
Benno SCHWIKOWSKI
Spencer SHORTE
Frédéric TANGY

Antoine TOUBERT (St Louis)
Mathilde TOUVIER (Université Paris)

Marie-Noelle UNGHEUER
Christophe ZIMMER

THANK YOU

Questions?

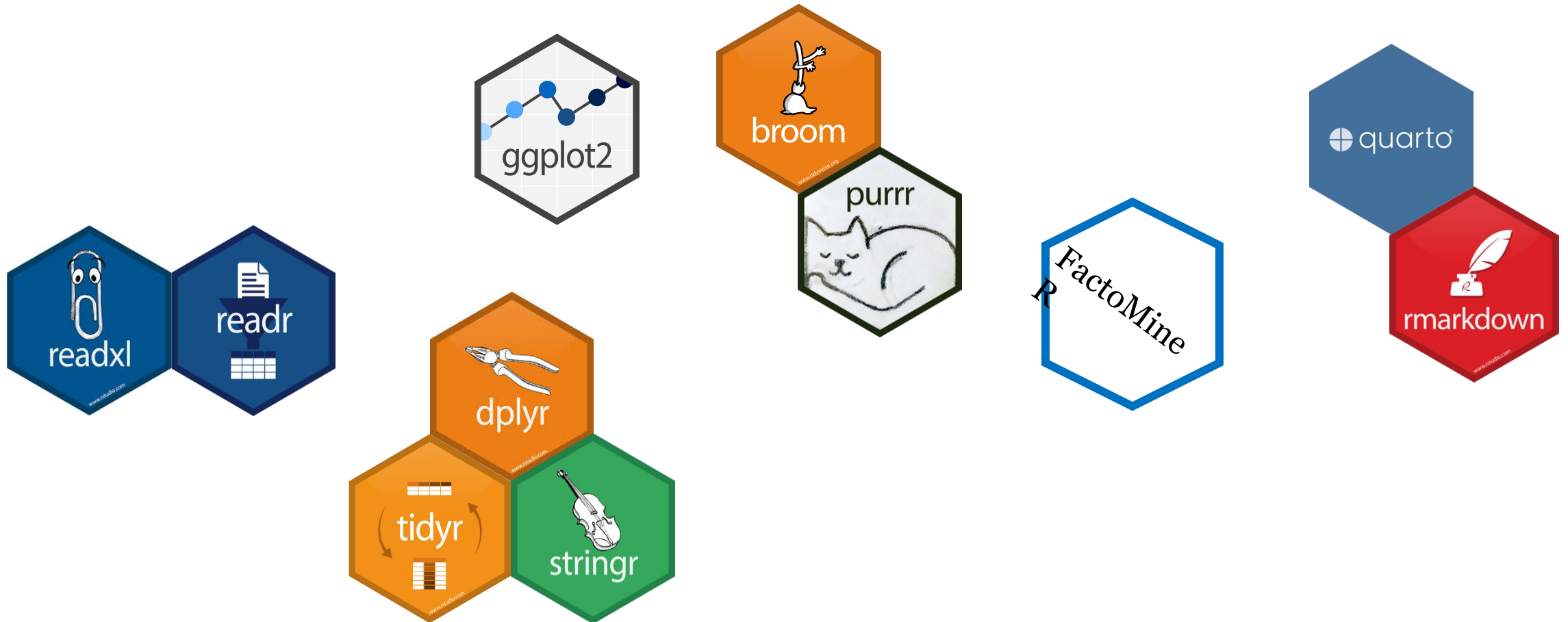
Comments?

RESOURCES

- R Cheat Sheets
- R Books
 - R for Data Science
 - R Cookbook
 - R Graphics Cookbook
- R Tutorials
 - [Carpentries Open Science](#)



R PACKAGES USED IN THIS WORKSHOP



TIDYVERSE

