

A Journey Into The World of Tidyverse

Coffee, Cookie and Coding (C³) Workshop supported by the
Public Health Data Science and Data Equity team

Howard Baik, M.S. and Shelby Golden, M.S.

Dec 2nd, 2024



Howard Baik, M.S.

- Worked 1.5 years as a Software Development Engineer on R packages and Shiny applications.
- Received a Masters in Biostatistics from the University of Washington in 2023.



Shelby Golden, M.S.

- Worked 7 years as a Molecular Biologist and Biochemist.
- Received a Masters in Applied Computational Mathematics from Johns Hopkins University in 2024.

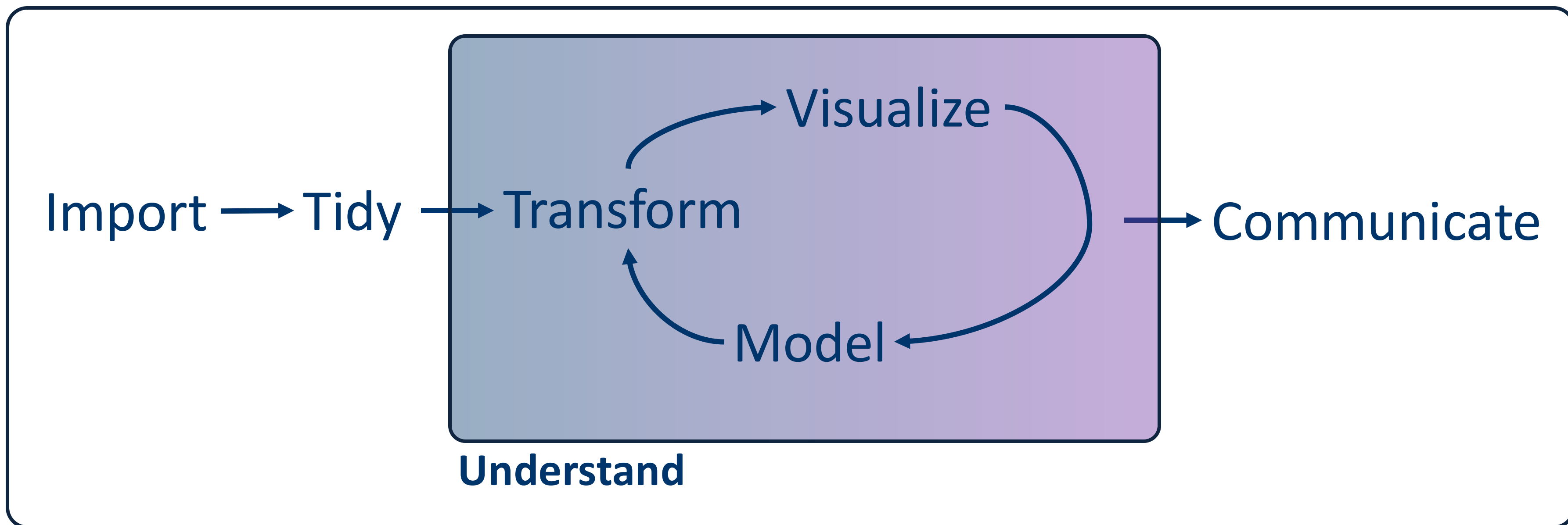
Today's Learning Objectives

- 01** Introduction to the tidyverse (~ 5 minutes)
- 02** Learn how to apply tidyr, dplyr, and stringr (~ 25 minutes)
- 03** Worked Through Example: Make an interpretable plot using ggplot2 in Posit Cloud (~ 20 minutes)

Our Choice Resources

- Yale's Center for Research Computing workshop [Tidying Data](#) by [Benjamin Evans](#)
- [Learn the tidyverse](#) webpage of resource by tidyverse
- [dplyr](#) package documentation and cheat sheets by tidyverse
- [tidyr](#) package documentation and cheat sheets by tidyverse
- [stringr](#) package documentation and cheat sheets by tidyverse

Welcome to the Tidyverse



Program

[R for Data Science \(2e\) - Introduction Figure 1](#) by Hadley Wickham, Mine Çetinkaya-Rundel, and Garrett Golemund. Accessed November 15th, 2024.



Commonly Used Core Packages

dplyr Tools for transforming data: i.e. `filter()`, `arrange()`, and `mutate()`.

tidyr Tools for tidying data: i.e. `pivot_wider()`, `pivot_longer()`, and `group_by()`.

stringr Tools to manage character strings: i.e. `str_c()`, `str_detect()`, and `str_replace()`.





[2019 COPSS Presidents' Award Winner Hadley Wickham](#)
Accessed November 18th, 2024.

Dr. Hadley Wickham

- Founder of tidyverse and leads the current team of collaborators.
- Chief scientist at Posit (formerly Rstudio).
- Adjunct professor at the University of Auckland, Stanford University, and Rice University.

In 2019 he was awarded the International COPSS Presidents' Award for

“ developing and implementing an impressively comprehensive computational infrastructure for data analysis through R software...”

– Citation on the COPSS plaque

The pipe, |>

Solving complex problems require combining multiple dplyr verbs with the pipe, |>

The pipe takes the thing on its left and passes it along to function on the right.

For example, $x \mid\> f(y)$ is equivalent to $f(x,y)$.

The easiest way to pronounce the pipe is “then”.

Source: <https://r4ds.hadley.nz/data-transform.html#dplyr-basics>

Background of Dataset

Dataset imported from COVID-19 Data Repository by the Center for Systems Science and Engineering (CSSE) at JHU.

We will be using time series data of COVID-19 death counts in the US. (1/22/2020 ~3/9/2023)

This raw dataset will be transformed using tidyverse packages.

A tibble: 10 × 6

	Country_Region	Province_State	Admin2	`6/11/21`	`6/12/21`	`6/13/21`
	<chr>	<chr>	<chr>	<dbl>	<dbl>	<dbl>
1	US	Connecticut	Fairfield	2201	2201	2201
2	US	Connecticut	Hartford	2431	2431	2431
3	US	Connecticut	Litchfield	298	298	298
4	US	Connecticut	Middlesex	371	371	371
5	US	Connecticut	New Haven	2127	2127	2127
6	US	Connecticut	New London	450	450	450
7	US	Connecticut	Out of CT	0	0	0
8	US	Connecticut	Tolland	187	187	187
9	US	Connecticut	Unassigned	1	1	1
10	US	Connecticut	Windham	195	195	195

Data Dictionary

- **Country_Region** : Represents the country, in this case, the United States.
- **Province_State** : Indicates the state within the country.
- **Admin2** : Specifies the county within the state.
- **date** : The date on which the data was recorded.
- **daily_count**: The number of COVID-19 deaths reported on that day.

```
# A tibble: 678 × 5
  Country_Region Province_State Admin2    date    daily_count
  <chr>          <chr>      <chr>    <date>      <dbl>
1 US            Connecticut New Haven 2021-05-01         0
2 US            Connecticut New Haven 2021-05-02         0
3 US            Connecticut New Haven 2021-05-03         4
4 US            Connecticut New Haven 2021-05-04         4
5 US            Connecticut New Haven 2021-05-05         1
6 US            Connecticut New Haven 2021-05-06         5
7 US            Connecticut New Haven 2021-05-07         1
8 US            Connecticut New Haven 2021-05-08         0
9 US            Connecticut New Haven 2021-05-09         0
10 US           Connecticut New Haven 2021-05-10         8
# i 668 more rows
# i Use `print(n = ...)` to see more rows
```

Import dataset

```
> library(tidyverse)
> covid19_death_url <-  
"https://raw.githubusercontent.com/CSSEGISandData/COVID-19/refs/heads/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_deaths_US.csv"  
  
> df <- read_csv(file = covid19_death_url)
```


Introducing tidy



tidyr helps you create tidy data

Tidy data is data where:

1. Each variable is a column
2. Each observation is a row
3. Each cell is one value

Describes a standard way of storing data that is used wherever possible throughout the tidyverse.

Source: <https://tidyr.tidyverse.org/>

tidyr helps you create tidy data

```
# A tibble: 6,780 × 5
  Country_Region Province_State Admin2    date    daily_count
  <chr>          <chr>        <chr>    <date>      <dbl>
1 US            Connecticut Fairfield 2021-05-01      0
2 US            Connecticut Fairfield 2021-05-02      0
3 US            Connecticut Fairfield 2021-05-03      2
4 US            Connecticut Fairfield 2021-05-04      1
5 US            Connecticut Fairfield 2021-05-05      0
6 US            Connecticut Fairfield 2021-05-06      1
7 US            Connecticut Fairfield 2021-05-07      4
8 US            Connecticut Fairfield 2021-05-08      0
9 US            Connecticut Fairfield 2021-05-09      0
10 US           Connecticut Fairfield 2021-05-10      1
# i 6,770 more rows
# i Use `print(n = ...)` to see more rows

# A tibble: 6,780 × 5
  Country_Region Province_State Admin2    date    daily_count
  <chr>          <chr>        <chr>    <date>      <dbl>
1 US            Connecticut Fairfield 2021-05-01      0
2 US            Connecticut Fairfield 2021-05-02      0
3 US            Connecticut Fairfield 2021-05-03      2
4 US            Connecticut Fairfield 2021-05-04      1
5 US            Connecticut Fairfield 2021-05-05      0
6 US            Connecticut Fairfield 2021-05-06      1
7 US            Connecticut Fairfield 2021-05-07      4
8 US            Connecticut Fairfield 2021-05-08      0
9 US            Connecticut Fairfield 2021-05-09      0
10 US           Connecticut Fairfield 2021-05-10      1
# i 6,770 more rows
# i Use `print(n = ...)` to see more rows
```

Variables

Values

Observations

tidyr helps you create tidy data

A tibble: 6,780 × 5

Country_Region	Province_State	Admin2	date	daily_count
<chr>	<chr>	<chr>	<date>	<dbl>
1 US	Connecticut	Fairfield	2021-05-01	0
2 US	Connecticut	Fairfield	2021-05-02	0
3 US	Connecticut	Fairfield	2021-05-03	2
4 US	Connecticut	Fairfield	2021-05-04	1
5 US	Connecticut	Fairfield	2021-05-05	0
6 US	Connecticut	Fairfield	2021-05-06	1
7 US	Connecticut	Fairfield	2021-05-07	4
8 US	Connecticut	Fairfield	2021-05-08	0
9 US	Connecticut	Fairfield	2021-05-09	0
10 US	Connecticut	Fairfield	2021-05-10	1

i 6,770 more rows
i Use `print(n = ...)` to see more rows

A tibble: 6,780 × 5

Country_Region	Province_State	Admin2	date	daily_count
<chr>	<chr>	<chr>	<date>	<dbl>
1 US	Connecticut	Fairfield	2021-05-01	0
2 US	Connecticut	Fairfield	2021-05-02	0
3 US	Connecticut	Fairfield	2021-05-03	2
4 US	Connecticut	Fairfield	2021-05-04	1
5 US	Connecticut	Fairfield	2021-05-05	0
6 US	Connecticut	Fairfield	2021-05-06	1
7 US	Connecticut	Fairfield	2021-05-07	4
8 US	Connecticut	Fairfield	2021-05-08	0
9 US	Connecticut	Fairfield	2021-05-09	0
10 US	Connecticut	Fairfield	2021-05-10	1

i 6,770 more rows
i Use `print(n = ...)` to see more rows

Variables

Values

Observations

Pivot data from wide to long

```
df |>
  pivot_longer(
    cols = "05/03/21":"05/05/21",
    names_to = "date",
    values_to = "daily_count")
```

Arguments in pivot_longer():

- `cols` = columns that need to be pivoted
- `names_to` = variable storing pivoted columns
- `values_to` = variable storing cell values

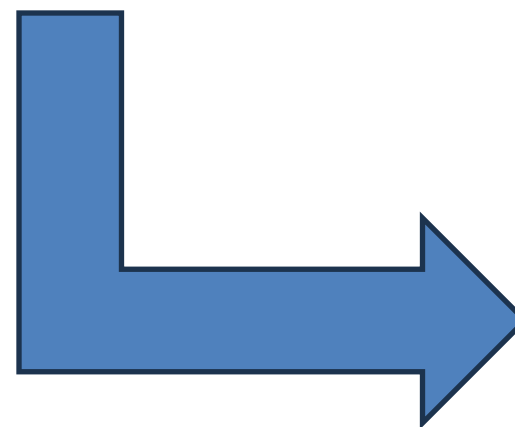
Pivot longer

Admin2	05/03/21	05/04/21	05/05/21
New Haven	4	4	1
New London	1	0	2
Fairfield	2	1	0

```
df |>
  pivot_longer(
    cols = "05/03/21":"05/05/21",
    names_to = "date",
    values_to = "daily_count")
```

Column headings (05/03/21, 05/04/21, 05/05/21) become values in new column, **date**

Values in **Admin2** column need to be repeated



Cell values became values in new column, **daily_count**

Admin2	date	daily_count
New Haven	05/03/21	4
New Haven	05/04/21	4
New Haven	05/05/21	1
New London	05/03/21	1
New London	05/04/21	0
New London	05/05/21	2
Fairfield	05/03/21	2
Fairfield	05/04/21	1
Fairfield	05/05/21	0

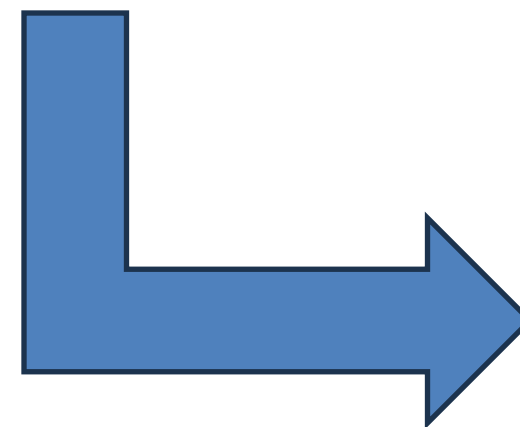
Pivot longer

Admin2	05/03/21	05/04/21	05/05/21
New Haven	4	4	1

```
df |>
  pivot_longer(
    cols = "05/03/21":"05/05/21",
    names_to = "date",
    values_to = "daily_count")
```

Column headings (05/03/21, 05/04/21, 05/05/21) become values in new column, **date**

Values in **Admin2** column need to be repeated



Cell values became values in new column, **daily_count**

Admin2	date	daily_count
New Haven	05/03/21	4
New Haven	05/04/21	4
New Haven	05/05/21	1

Pivot data from long to wide

```
df |>  
  pivot_wider(  
    names_from = date,  
    values_from = daily_count)
```

Arguments in pivot_wider():

- `names_from` = column with new variable names
- `values_from` = column with values for new variables

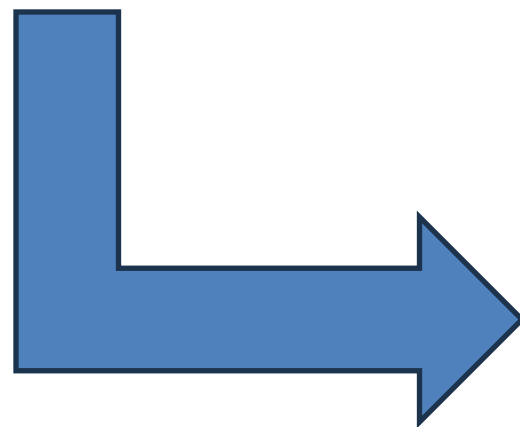
Pivot wider

```
df |>
  pivot_wider(
    names_from = date,
    values_from = daily_count)
```

Admin2	date	daily_count
New Haven	05/03/21	4
New Haven	05/04/21	4
New Haven	05/05/21	1
New London	05/03/21	1
New London	05/04/21	0
New London	05/05/21	2
Fairfield	05/03/21	2
Fairfield	05/04/21	1
Fairfield	05/05/21	0

Values in **date** column are widened

Values in **daily_count**
become the cell values



Admin2	05/03/21	05/04/21	05/05/21
New Haven	4	4	1
New London	1	0	2
Fairfield	2	1	0

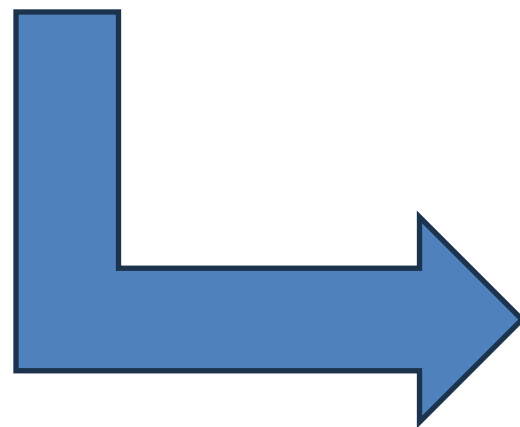
Pivot wider

```
df |>
  pivot_wider(
    names_from = date,
    values_from = daily_count)
```

Admin2	date	daily_count
New Haven	05/03/21	4
New Haven	05/04/21	4
New Haven	05/05/21	1

Values in **date** column are widened

Values in **daily_count**
become the cell values



Admin2	05/03/21	05/04/21	05/05/21
New Haven	4	4	1

Introducing dplyr



Grammar of Data Manipulation

dplyr provides a consistent set of verbs for data manipulation:

- `select()` picks variables based on their names.
- `mutate()` adds new variables that are functions of existing variables.
- `filter()` picks rows based on their values.
- `group_by()` allows you to perform any operation “by group”
- `summarise()` reduces multiple values down to a summary.

select() picks variables based on their name

```
df |> select(date, daily_count)
```

```
# A tibble: 678 × 2
```

	date	daily_count
	<date>	<dbl>
1	2021-05-01	0
2	2021-05-02	0
3	2021-05-03	4
4	2021-05-04	4
5	2021-05-05	1
6	2021-05-06	5
7	2021-05-07	1
8	2021-05-08	0
9	2021-05-09	0
10	2021-05-10	8

```
# i 668 more rows
```

```
# i Use `print(n = ...)` to see more rows
```

mutate() adds new variables

```
df |>
  mutate(
    county_state = paste0(Admin2, ",", Province_State))
```

```
# A tibble: 678 × 6
```

	Country_Region	Province_State	Admin2	date	daily_count	county_state
	<chr>	<chr>	<chr>	<date>	<dbl>	<chr>
1	US	Connecticut	New Haven	2021-05-01	0	New Haven,Connecticut
2	US	Connecticut	New Haven	2021-05-02	0	New Haven,Connecticut
3	US	Connecticut	New Haven	2021-05-03	4	New Haven,Connecticut
4	US	Connecticut	New Haven	2021-05-04	4	New Haven,Connecticut
5	US	Connecticut	New Haven	2021-05-05	1	New Haven,Connecticut
6	US	Connecticut	New Haven	2021-05-06	5	New Haven,Connecticut
7	US	Connecticut	New Haven	2021-05-07	1	New Haven,Connecticut
8	US	Connecticut	New Haven	2021-05-08	0	New Haven,Connecticut
9	US	Connecticut	New Haven	2021-05-09	0	New Haven,Connecticut
10	US	Connecticut	New Haven	2021-05-10	8	New Haven,Connecticut

```
# i 668 more rows
```

```
# i Use `print(n = ...)` to see more rows
```

filter() picks rows

```
df |>  
  filter(date == "2021-05-03")
```

```
# A tibble: 1 × 5
```

	Country_Region	Province_State	Admin2	date	daily_count
	<chr>	<chr>	<chr>	<date>	<dbl>
1	US	Connecticut	New Haven	2021-05-03	4

group_by() performs operation “by group”

```
df |>
```

```
  group_by(Province_State, date) |>
```

```
  summarise(daily_count = sum(daily_count))
```

```
# A tibble: 678 × 3
```

```
# Groups:   Province_State [1]
```

	Province_State	date	daily_count
	<chr>	<date>	<dbl>
1	Connecticut	2021-05-01	0
2	Connecticut	2021-05-02	0
3	Connecticut	2021-05-03	15
4	Connecticut	2021-05-04	5
5	Connecticut	2021-05-05	7
6	Connecticut	2021-05-06	7
7	Connecticut	2021-05-07	6
8	Connecticut	2021-05-08	0
9	Connecticut	2021-05-09	0
10	Connecticut	2021-05-10	17

```
# i 668 more rows
```

```
# i Use `print(n = ...)` to see more rows
```

summarise() performs summary operations

Introducing stringr



stringr contains a set of functions for manipulating and interpreting strings.

str_c() Join discrete strings into one. Can specify spacers.

str_detect() Find pattern match within strings.

str_length() Counts the code points, or characters, in a string.

str_replace() Replace the first match of a pattern in a string.

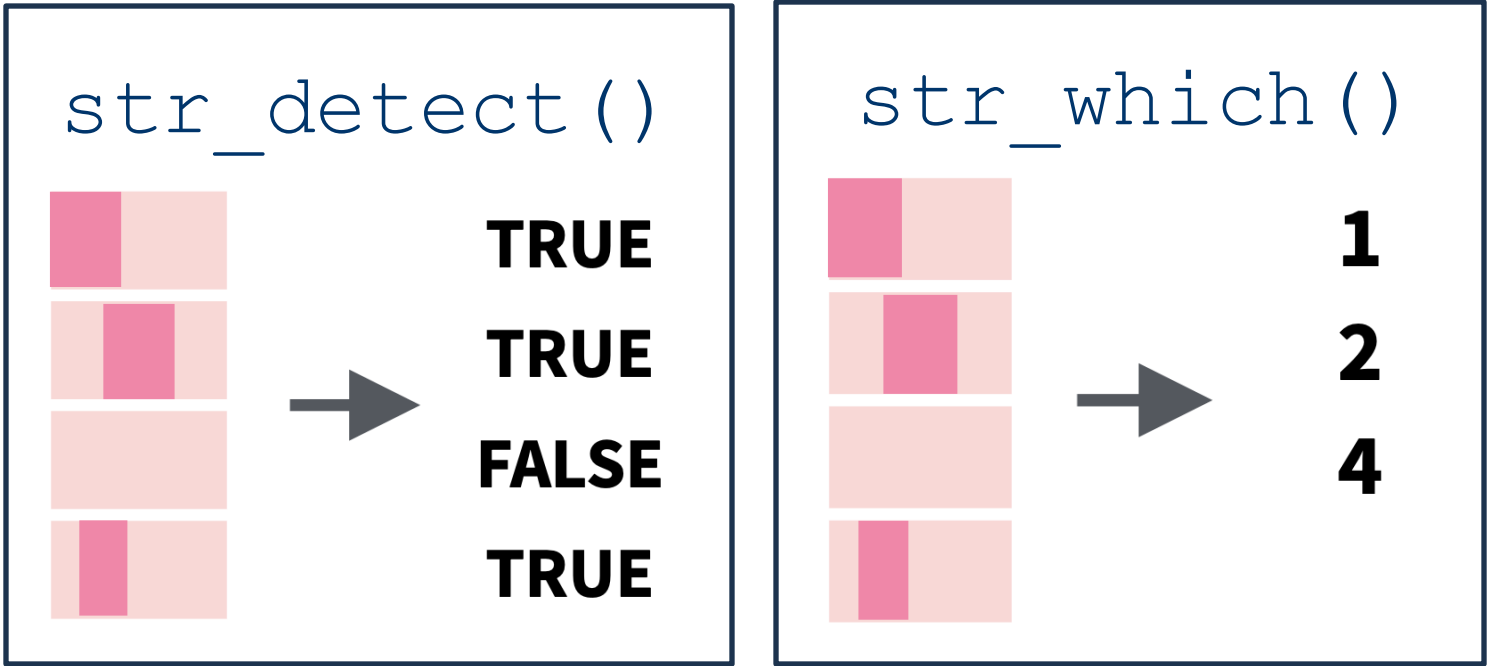
str_lower() Convert strings to lower case.

str_upper() Convert strings to upper case.

str_title() Convert strings to tittle case.

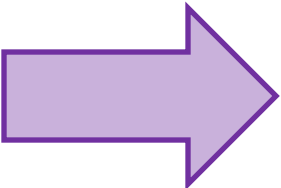
str_sort() Sorts the character vector.

Two functions that find rows where a string match is found. Reports as a Boolean or index.

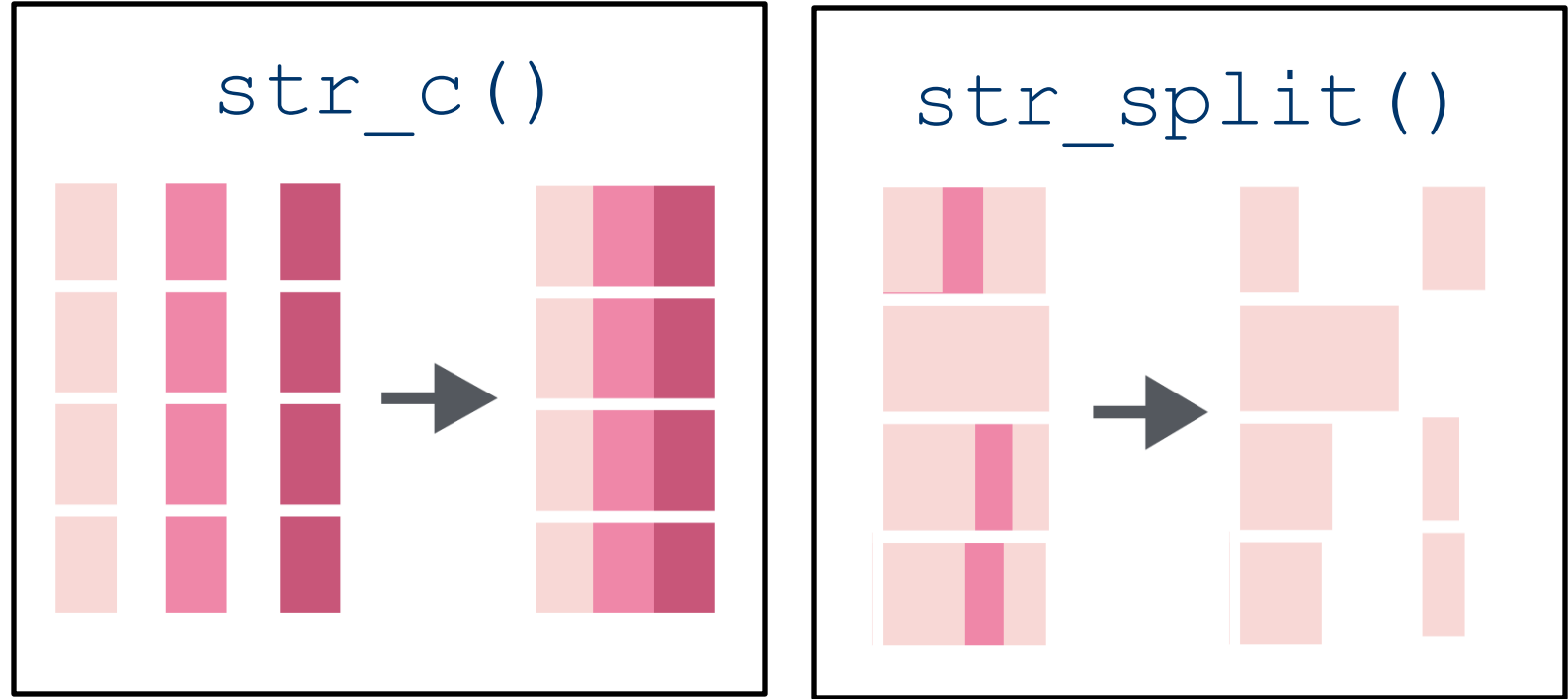


```
df[!str_detect(df$Province_State, "Princess"), ]  
# or  
df[-str_which(df$Province_State, "Princess"), ]
```

Province_State	Deaths_Count_Cumulative
Connecticut	5995
Diamond Princess	0
Florida	21673
Georgia	10958
Grand Princess	3



Province_State	Deaths_Count_Cumulative
Connecticut	5995
Florida	21673
Georgia	10958



Two functions that generate a new string by joining discrete strings or splitting a composite.

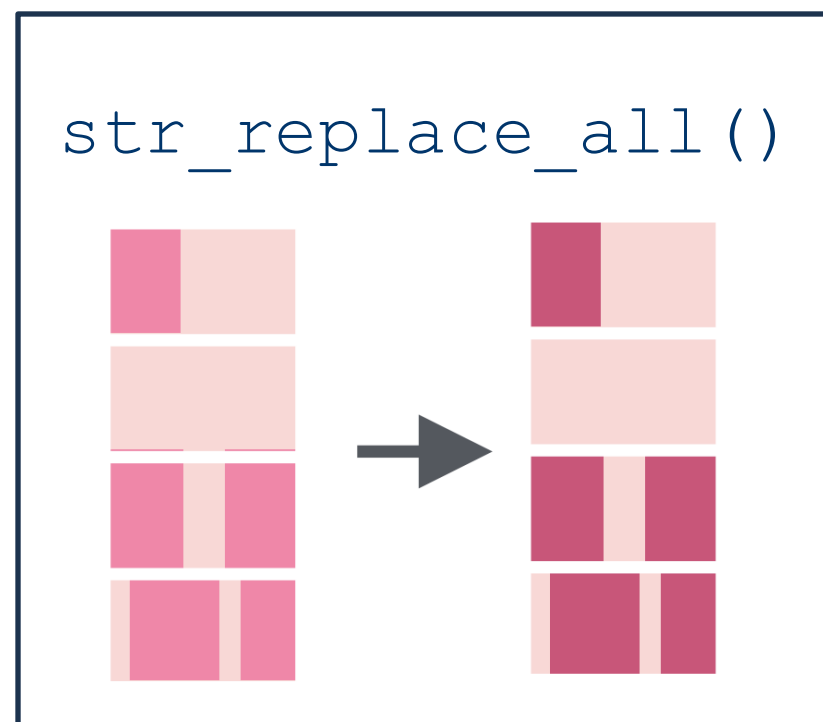
```
new <- str_c(
  df_filtered$Province_State,
  df_filtered$Country_Region,
  sep = ", ")
```

or

```
new <- str_split(
  df_filtered$Combined_Key,
  ",", simplify = TRUE,
  n = 2)[, 2]
```

Province_State	Country_Region	new
Montana	US	Montana, US
Oregon	US	Oregon, US
Hawaii	US	Hawaii, US

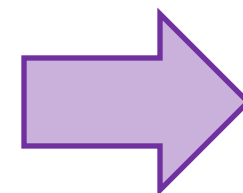
Combined_Key	new
Cheyenne, Colorado, US	Colorado, US
McMinn, Tennessee, US	Tennessee, US
Accomack, Virginia, US	Virginia, US



Find string matches and replace those substrings with a new string.

```
df_filtered[, "Province_State"] <- str_replace(df_filtered[,  
  "Province_State"], "Virgin Islands", "US Virgin Islands")  
# followed by  
df_filtered[, "Combined_Key"] <- str_replace(df_filtered[,  
  "Combined_Key"], "Virgin Islands", "US Virgin Islands")
```

Province_State	Combined_Key
Virginia	Virginia, US
Virgin Islands	Virgin Islands, US
Texas	Texas, US



Province_State	Combined_Key
Virginia	Virginia, US
US Virgin Islands	US Virgin Islands, US
Texas	Texas, US

Worked Through Example on Posit Cloud

Worked Through Example

Dataset Information:

- The dataset contains COVID-19 death counts in the US
- Covers the period from January 22, 2020, to March 9, 2023.
- Same dataset used in these slides

Getting Started:

- Visit this [link](#) and login to your Posit Cloud account

After the workshop:

- Posit Cloud workspace will be archived – Make sure to “Export” your work.

Quarto



Overview:

- Document where you can run R code and weave text in the same file
- **Next-generation version** of R Markdown from Posit

Features:

- Editors available: Can use either “Source” mode or “Visual” mode
- Shortcuts available for inserting code chunks and running code

Appendix

Glossary

Import Data Loading data from a stored file, database, or application programming interface (API) into the R environment.

Tidy Data Formatting data into a consistent structure without anomalies. Each column represents a variable, and each row represents an observation.

Transform Data Usually involves creating a new variable that is a function of other ones (i.e. converting units), subsetting to focus on specific outcomes, or calculating summary statistics.

Wrangle Data The process of tidying and transforming data.

References

Slide 4

1. Ph. D. , B. Evans, "Tidying Data," Yale Center for Research Computing (YCRC). Accessed: Nov. 14, 2024. [Online]. Available: <https://research.computing.yale.edu/training/tidying-data>
2. H. Wickham, M. Çetinkaya-Rundel, and G. Grolemund, "Learn the tidyverse," Tidyverse. Accessed: Nov. 15, 2024. [Online]. Available: <https://www.tidyverse.org/learn/>
3. H. Wickham, R. François, L. Henry, K. Müller, and D. Vaughan, "A Grammar of Data Manipulation • dplyr," Tidyverse. Accessed: Nov. 15, 2024. [Online]. Available: <https://dplyr.tidyverse.org/>
4. H. Wickham, D. Vaughan, and M. Girlich, "Tidy Messy Data • tidyr," Tidyverse. Accessed: Nov. 15, 2024. [Online]. Available: <https://tidyr.tidyverse.org/>
5. H. Wickham, "Simple, Consistent Wrappers for Common String Operations • stringr," Tidyverse. Accessed: Nov. 15, 2024. [Online]. Available: <https://stringr.tidyverse.org/>

References

Slide 6

1. H. Wickham, M. Çetinkaya-Rundel, and G. Grolemund, "Introduction," in R for Data Science (2e), O'Reilly Media. Accessed: Nov. 14, 2024. [Online]. Available: <https://r4ds.hadley.nz/intro>
2. "Tidyverse," Wikipedia. Accessed: Nov. 14, 2024. [Online]. Available: <https://en.wikipedia.org/wiki/Tidyverse>

Slide 7

1. H. Wickham et al., "Welcome to the Tidyverse," Journal of Open Source Software, vol. 4, no. 43, p. 1686, Nov. 2019, doi: 10.21105/JOSS.01686.
2. H. Wickham, "Tidyverse." Accessed: Nov. 14, 2024. [Online]. Available: <https://www.tidyverse.org/>

Slide 8

1. "COPSS Award for Dr. Hadley Wickham," Committee of Presidents of Statistical Societies (COPSS). Accessed: Nov. 17, 2024. [Online]. Available: <https://community.amstat.org/copss/awards/presidents/2018151>

References

Slide 8

2. H. Wickham, "Personal Website." Accessed: Nov. 17, 2024. [Online]. Available: <https://hadley.nz/>
3. "Hadley Wickham," Wikipedia. Accessed: Nov. 17, 2024. [Online]. Available: https://en.wikipedia.org/wiki/Hadley_Wickham
4. "Tidyverse," Wikipedia. Accessed: Nov. 14, 2024. [Online]. Available: <https://en.wikipedia.org/wiki/Tidyverse>

Slides 14-22

1. H. Wickham, D. Vaughan, and M. Girlich, "Tidy Messy Data • tidyr," Tidyverse. Accessed: Nov. 15, 2024. [Online]. Available: <https://tidyr.tidyverse.org/>

Slide 24-28

1. H. Wickham, M. Çetinkaya-Rundel, and G. Grolemund, "Data transformation," in R for Data Science (2e), O'Reilly Media. Accessed: Nov. 14, 2024. [Online]. Available: <https://r4ds.hadley.nz/data-transform>

References

Slide 31-33

2. H. Wickham and L. Vaudor, “stringr Cheat Sheet,” RStudio Cheat Sheets. Accessed: Nov. 15, 2024. [Online]. Available: <https://github.com/rstudio/cheatsheets/blob/main/strings.pdf>

Slides 38

1. H. Wickham and L. Vaudor, “stringr Cheat Sheet,” RStudio Cheat Sheets. Accessed: Nov. 15, 2024. [Online]. Available: <https://github.com/rstudio/cheatsheets/blob/main/strings.pdf>

Evaluation

Feedback Form for Workshop:
Journey Into the Tidyverse




Next DSDE Event

Yale SCHOOL OF PUBLIC HEALTH
Data Science and Data Equity

JOURNEY LECTURE


Jeffrey Townsend, PhD
Elihu Professor of Biostatistics
Professor of Ecology and Evolutionary Biology
Dec 6, 2024 | 12 to 1 pm
LEPH Winslow Auditorium
60 College Street,
New Haven, CT

Speaker's favorite
snack will be served



gorp

RSVP required.



<https://tinyurl.com/dsdejournylecture1>

sph.yale.edu/dsde

@YaleSPH

Public Health Data Science and Data Equity
Yale School of Public Health
60 College Street, New Haven, CT 06510

Yale SCHOOL OF PUBLIC HEALTH