Data Visualization with Esquisse

Esquisse Package

```
# install.packages("esquisse")
library(esquisse)
```

Esquisse Package

The `esquisse` package is helpful for getting used to creating plots in R.

It is an interactive tool to help you in RStudio.

It's super **nifty**!

First, get some data..

We can use the CO heat-related ER visits dataset. This dataset contains information about the number and rate of visits for heat-related illness to ERs in Colorado from 2011-2022, adjusted for age.

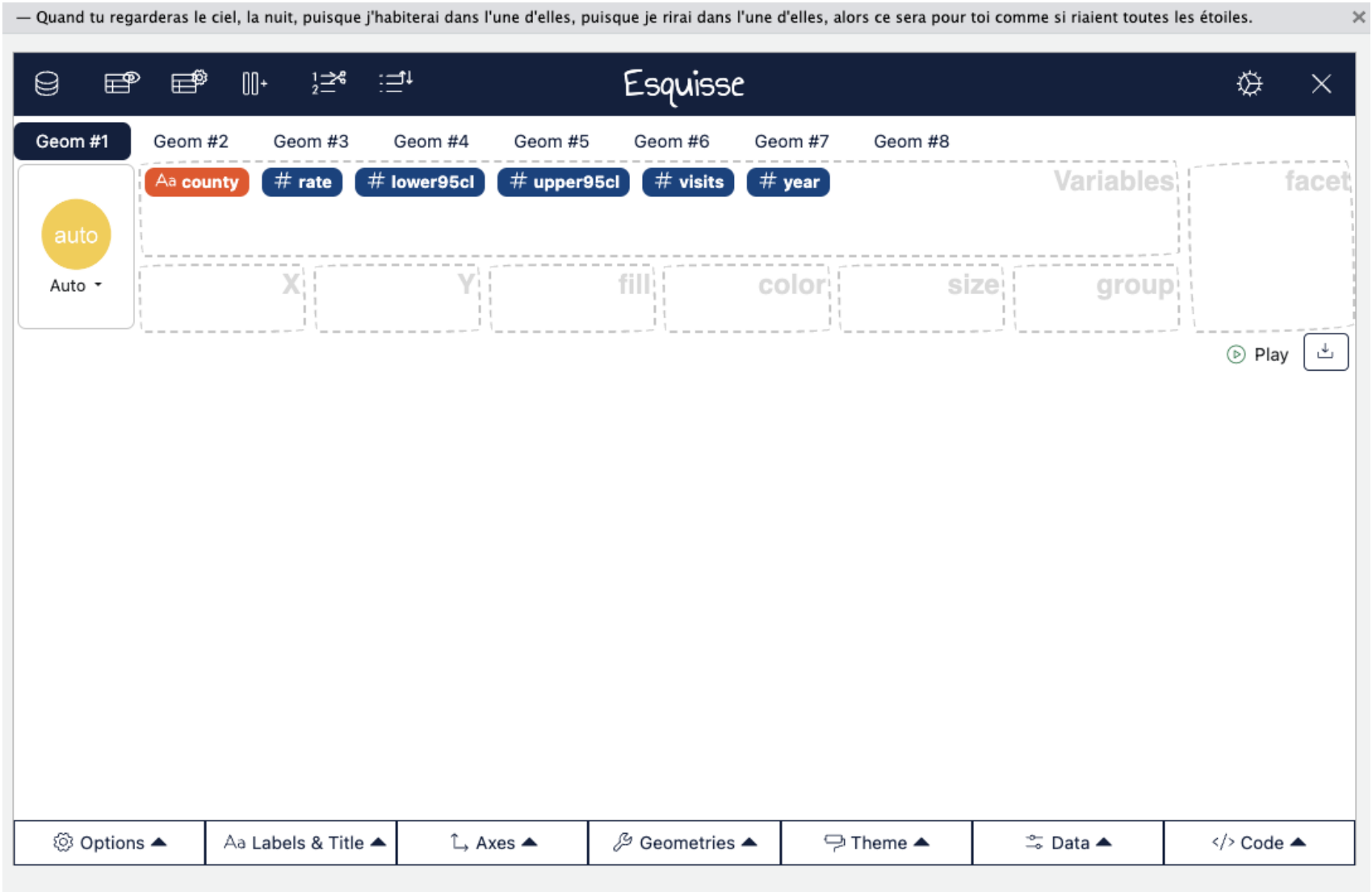
```
er <-
  read_csv("https://daseh.org/data/CO_ER_heat_visits.csv")

head(er)
```

```
## # A tibble: 6 × 6
##   county  rate lower95cl upper95cl visits  year
##   <chr>  <dbl>     <dbl>     <dbl>  <dbl> <dbl>
## 1 Adams   6.73     NA         9.24     29  2011
## 2 Adams   4.84      2.85     NA        23  2012
## 3 Adams   6.84      4.36      9.31     31  2013
## 4 Adams   3.08      1.71      4.85     15  2014
## 5 Adams   3.36      1.89      5.23     16  2015
## 6 Adams   8.85      6.12     11.6      42  2016
```

Starting a plot

Using the `esquisser()` function you can start creating a plot for a `data.frame` or `tibble`. That's it!

```
esquisser(er)
```

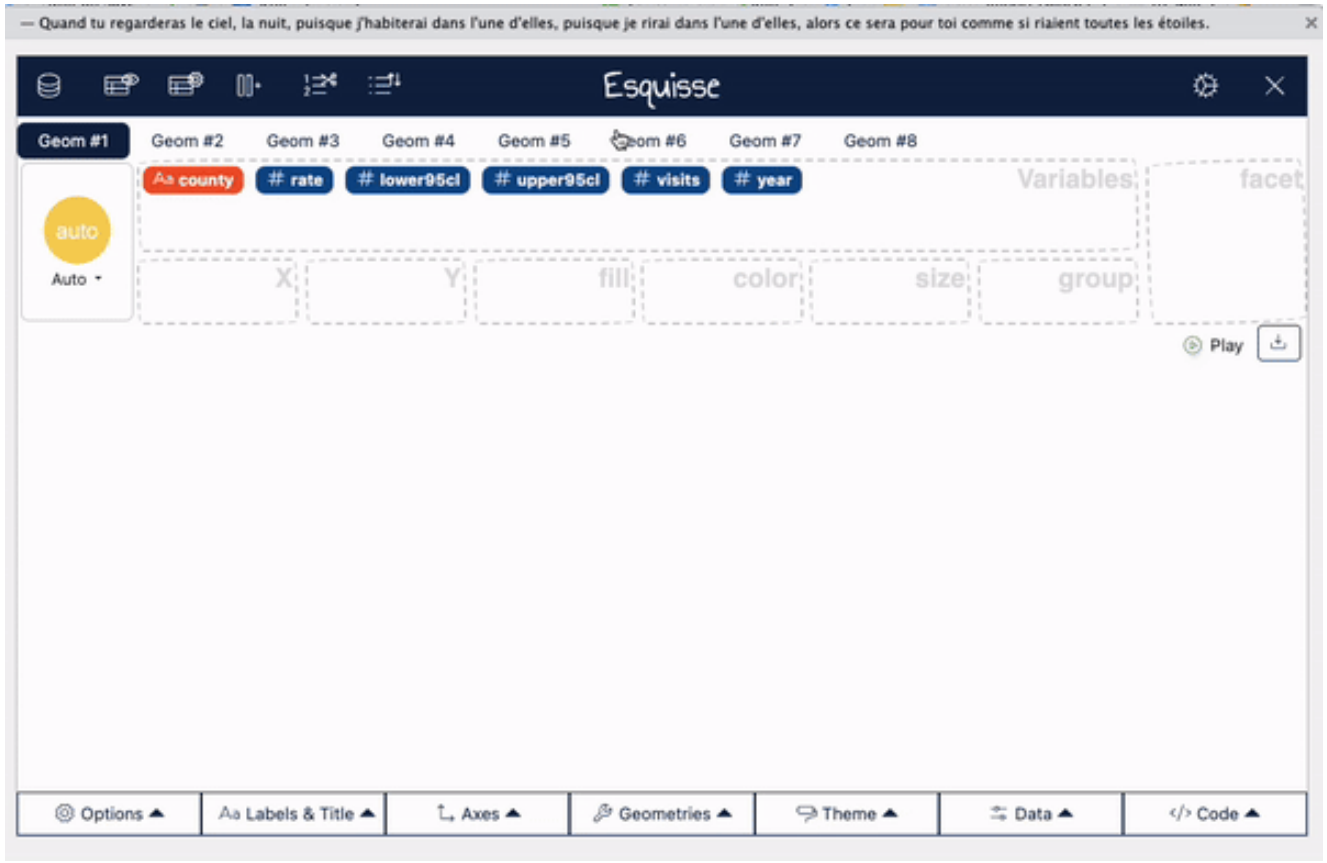


Show the plot in the browser

```
esquisse::esquisser(er, viewer = "browser")
```

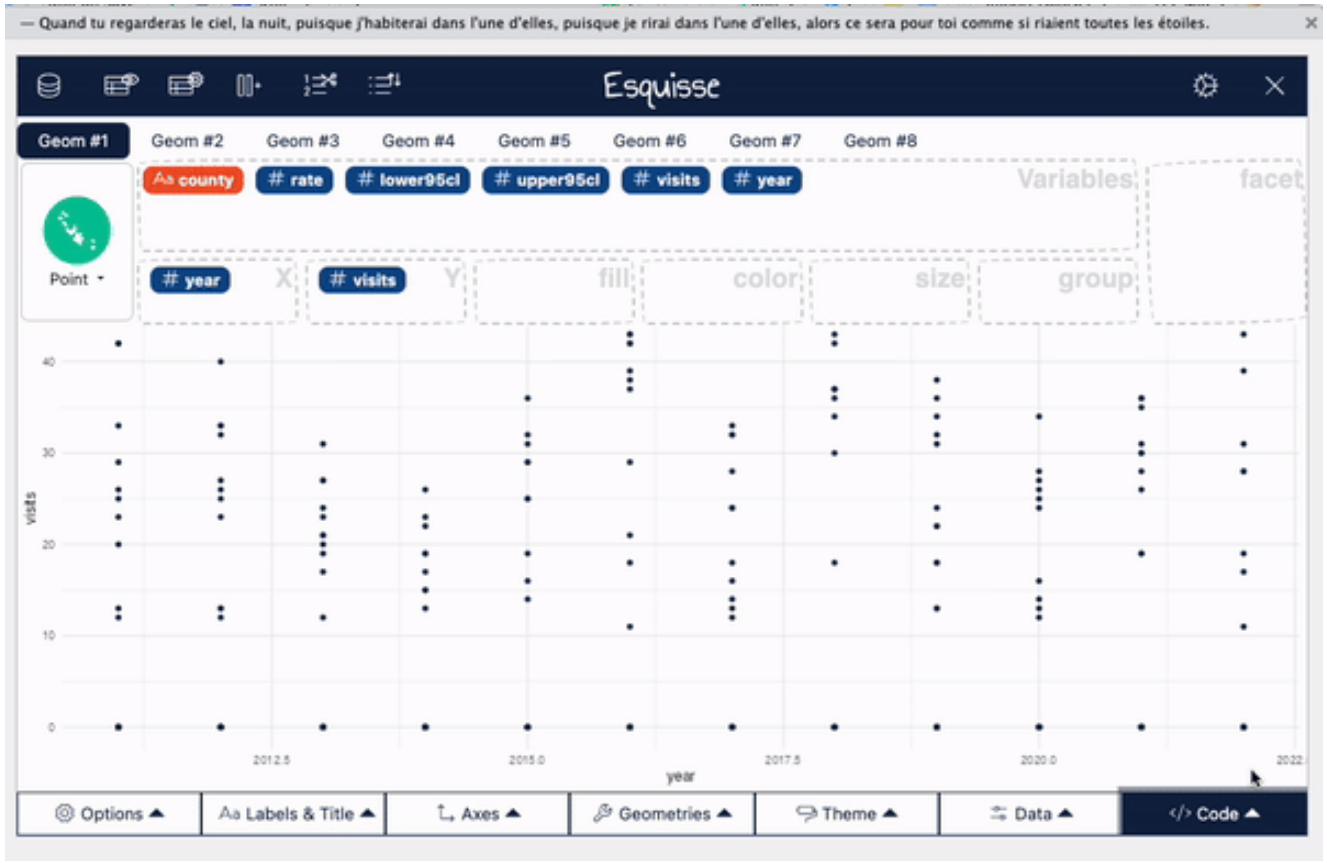
Select Variables

To select variables you can drag and drop variables to the respective axis that you would like the variable to be plotted on.



Find code

To select variables you can drag and drop variables to the respective axis that you would like the variable to be plotted on.

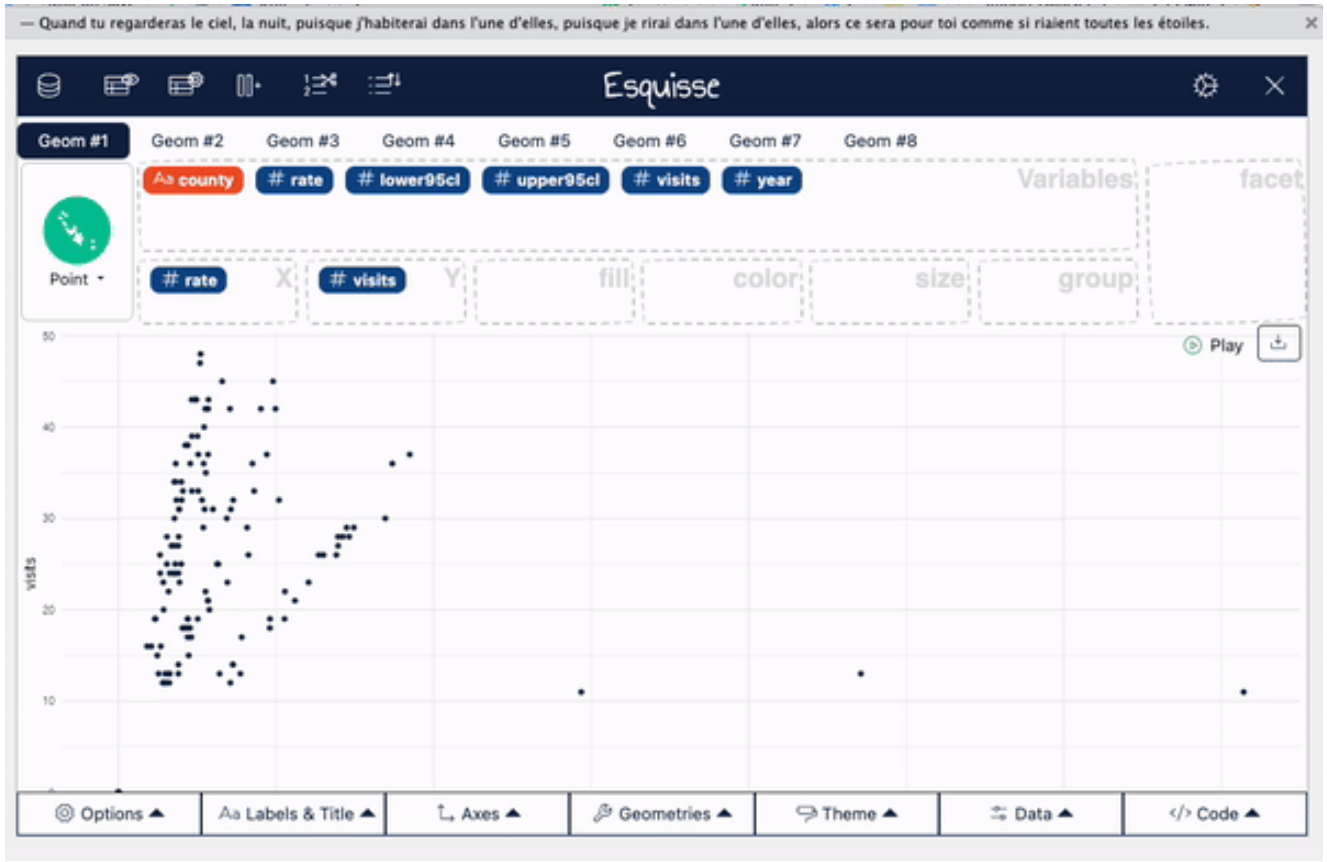


Change plot type

`esquisse` automatically assumes a plot type, but you might want to change this.

Add Facets

Facets create multiple plots based on the different values of a variable.



Add size

Sometimes it is useful to change the way points are plotted so that size represents a variable. This can especially be helpful if you need your plot to be black and white.

Add color

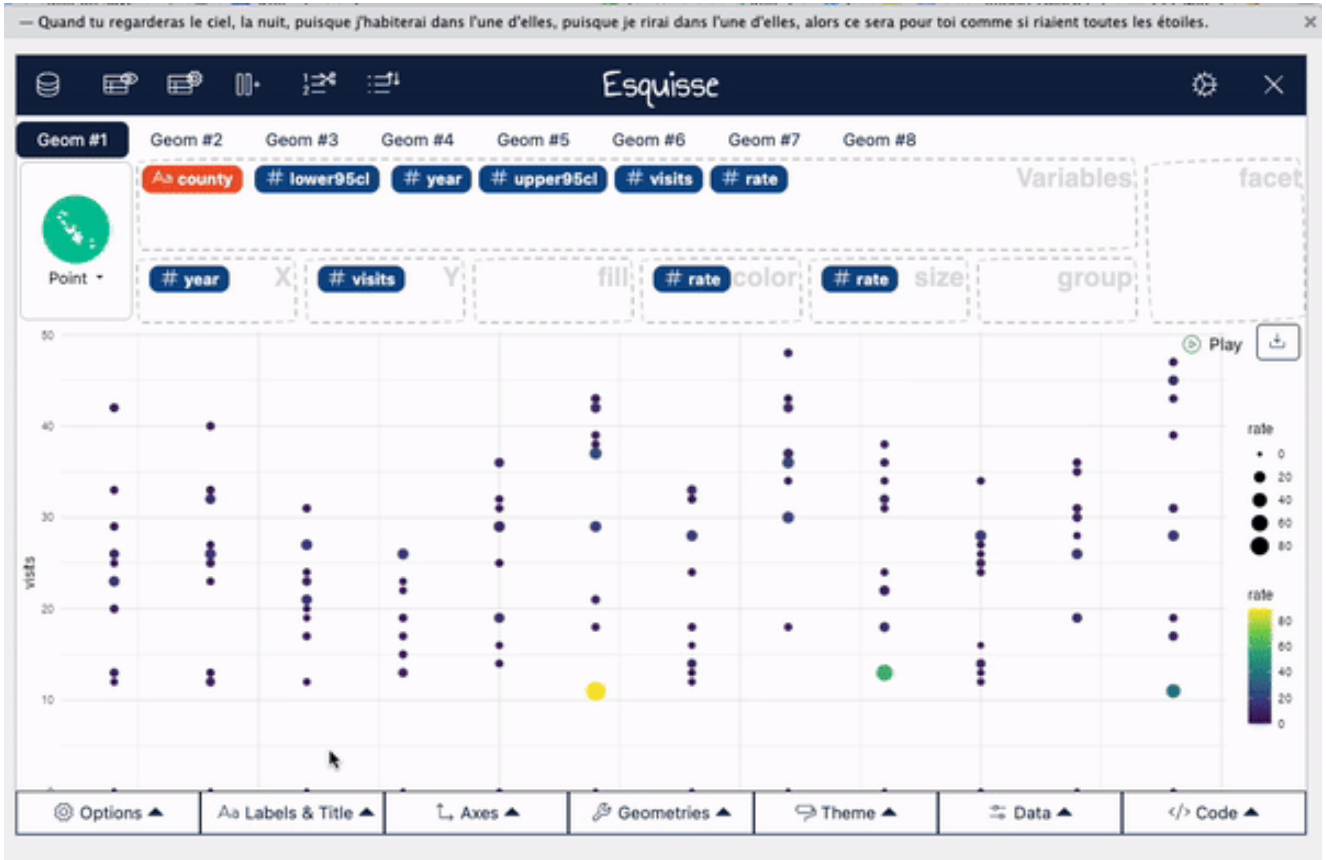
For plots with points use the color region to change coloring according to a variable. (use “fill” for bar plots)

Appearance

You can change the overall appearance with “Geometries” and “Theme”.

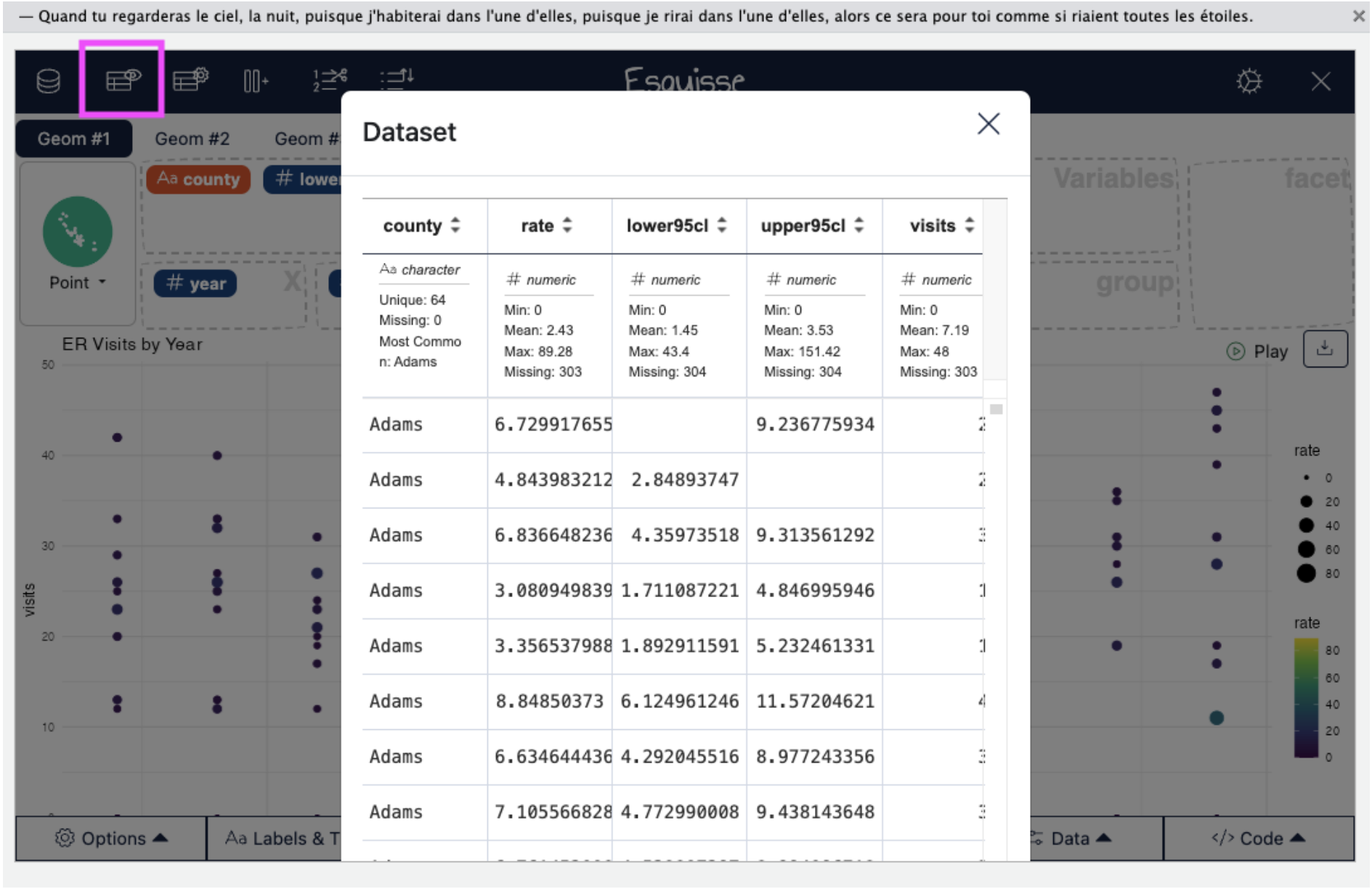
Change titles

To change titles on your plot, use the “Labels & Titles” tab.



View data

You can also easily view data

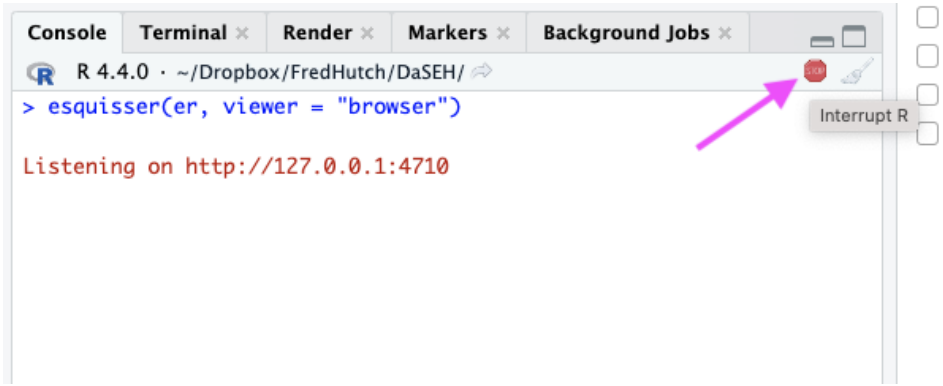


Interrupting Esquisse

You'll need to "interrupt" Esquisse to launch it with a new dataset.

Use the stop button or press ctrl+c to stop the Esquisse app.

If you don't see the stop button, you need to resize your window.



Wide & Long Data ?

Let’s look at why we might want long data using Esquisse.

```
library(tidyverse)
er <- read_csv(file =
    "https://daseh.org/data/CO_ER_heat_visits.csv")
long_er <- er %>%
  filter(county == c("Denver", "Boulder")) %>%
  select(c("county", "year", "visits"))
glimpse(long_er)
```

```
## Rows: 12
## Columns: 3
## $ county <chr> "Boulder", "Boulder", "Boulder", "Boulder", "Boulder", "Boulder…
## $ year   <dbl> 2012, 2014, 2016, 2018, 2020, 2022, 2011, 2013, 2015, 2017, 201…
## $ visits <dbl> 13, 19, 18, 18, 12, 19, 42, 19, 25, 24, 34, 28
```

Wide Data

As a comparison, let's also load a wide version of this dataset.

```
wide_er <- read_csv(file =
    "https://daseh.org/data/CO_heat_er_visits_DenverBoulder_wide.csv")
```

```
## Rows: 2 Columns: 13
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## chr  (1): county
## dbl (12): 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, .
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this mess
```

Wide vs Long Data

```
head(long_er)
```

```
## # A tibble: 6 × 3
##   county   year visits
##   <chr>   <dbl>  <dbl>
## 1 Boulder  2012     13
## 2 Boulder  2014     19
## 3 Boulder  2016     18
## 4 Boulder  2018     18
## 5 Boulder  2020     12
## 6 Boulder  2022     19
```

```
head(wide_er)
```

```
## # A tibble: 2 × 13
##   county  `2011` `2012` `2013` `2014` `2015` `2016` `2017` `2018` `2019` `202
##   <chr>    <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <db
## 1 Boulder   4.03   4.08   3.79   6.29   4.76   5.68   3.51   5.07   3.71   3.
## 2 Denver    7.11   6.79   2.95   3.56   3.84   6.18   3.32   5.81   4.54   4.
## # ℹ 2 more variables: `2021` <dbl>, `2022` <dbl>
```

Make a plot of visit rates by year for different counties

```
esquisser(wide_er) # county as x...? Tricky!
esquisser(long_er) #county as x, visit rate as y, year as fill
```

GUT CHECK!

Why use Esquisse?

A. Explore your data

B. Get a “head start” on your code

C. Both of these!

Some Alternatives to **esquisse**

- `ggquickeda`: https://smouksassi.github.io/ggquickeda/
- `ggraptR`: https://github.com/cargomoose/ggraptR/
- `autoplot` can be helpful for some packages (see this blog post)

Summary

- Use the `esquisser()` function on a dataset
- Use the `viewer = "browser"` argument to launch in your browser.
- Code from Esquisse can copied into code chunks to be generated in the "Plots" pane
- It's easier if your code is in "long" form!

Lab

- Class Website
- Lab
- Day 6 Cheatsheet

Image by Gerd Altmann from Pixabay