Geographic Mapping Techniques

Jedidiah Harwood[*]
Kurt Wydrinski[†]

11/22/2019

```
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------------------------------------ t
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
## -- Conflicts --------------------------------------------------------------------------- tidyvers
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(maps)
```

```
##
## Attaching package: 'maps'
## The following object is masked from 'package:purrr':
##
##     map
```

Introduction

There are a number of techniques available for creating geographic maps within R. Techniques using both the `maps`, `ggplot` and other `tidyverse` packages are demonstrated in the following sections.

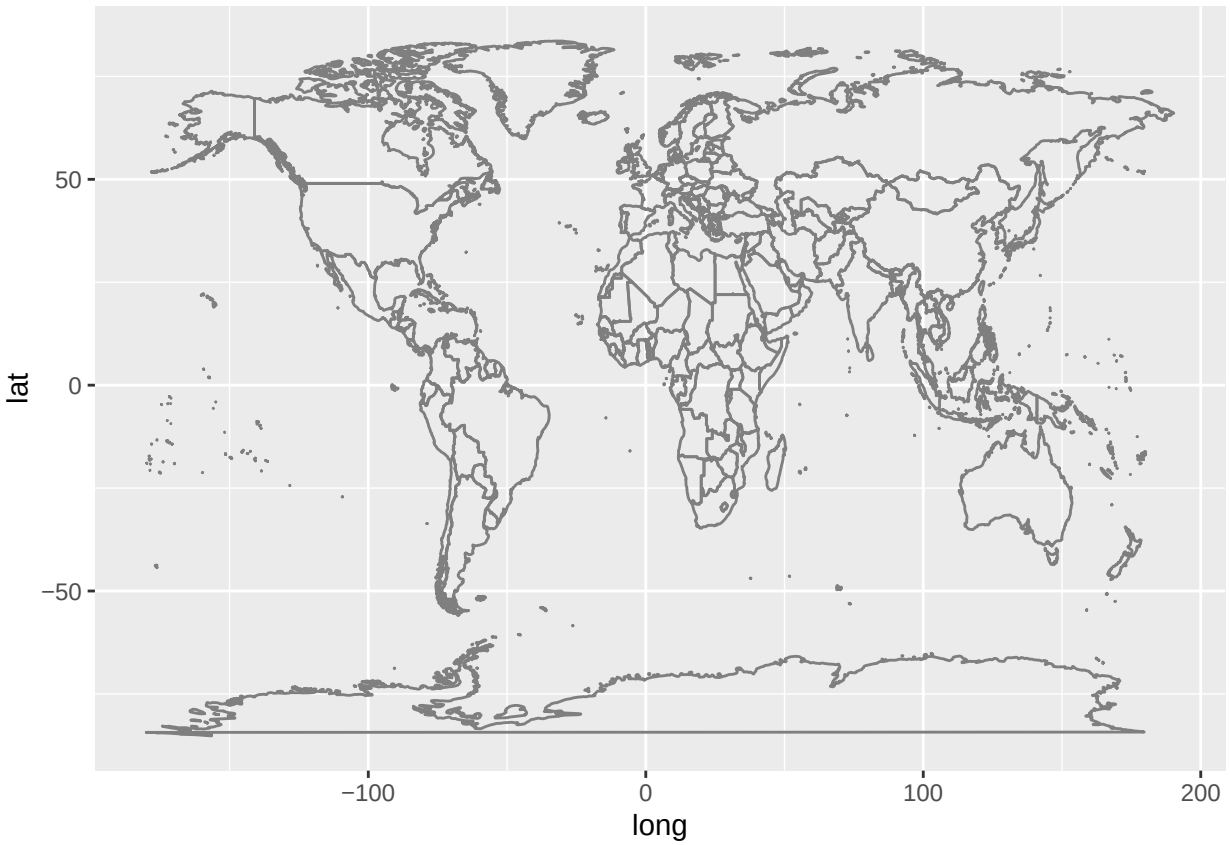
Basic Techniques

World Maps

The code below demonstrates the simplest world map. By default, borders are drawn 1 unit thick in a medium grey color. Also notice that the map is centered on the Prime Meridian at 0° longitude.

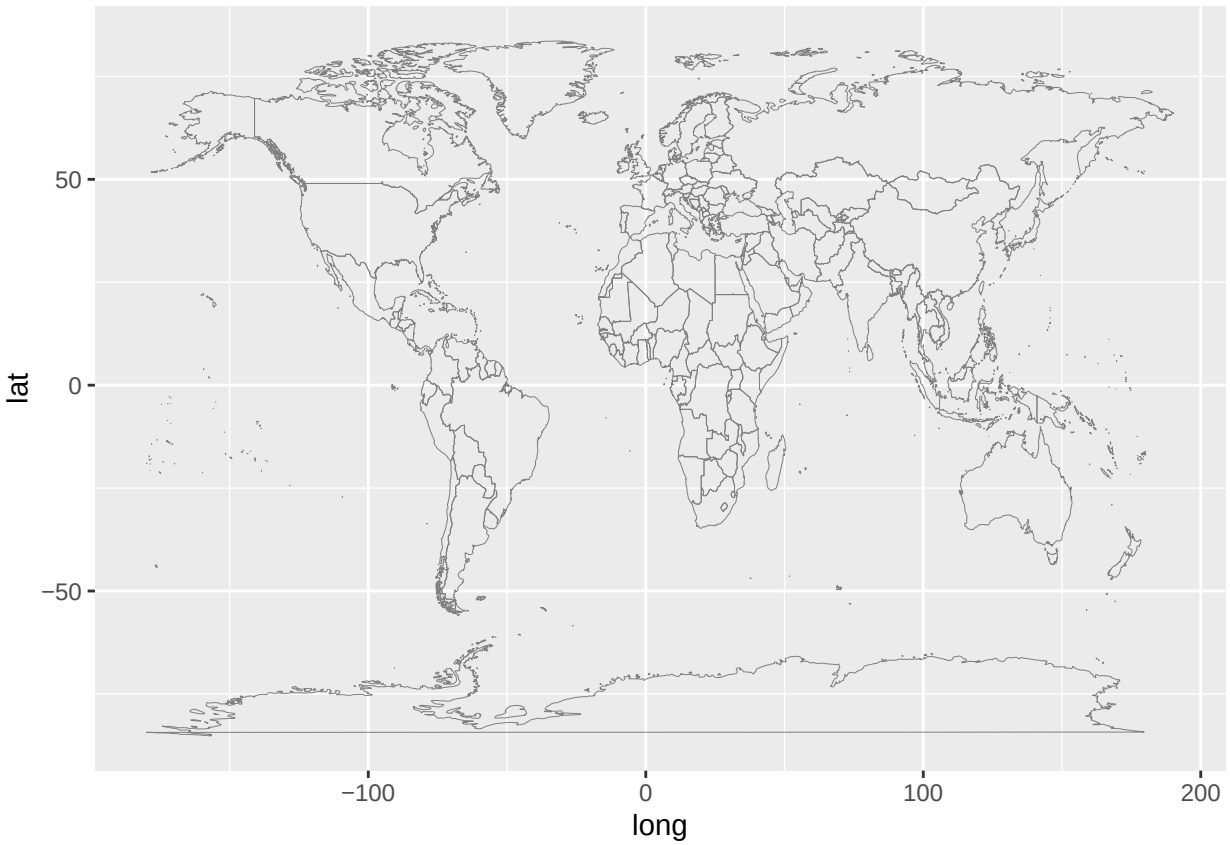
```
ggplot() +
  borders()
```

[*]California State University - East Bay
[†]California State University - East Bay



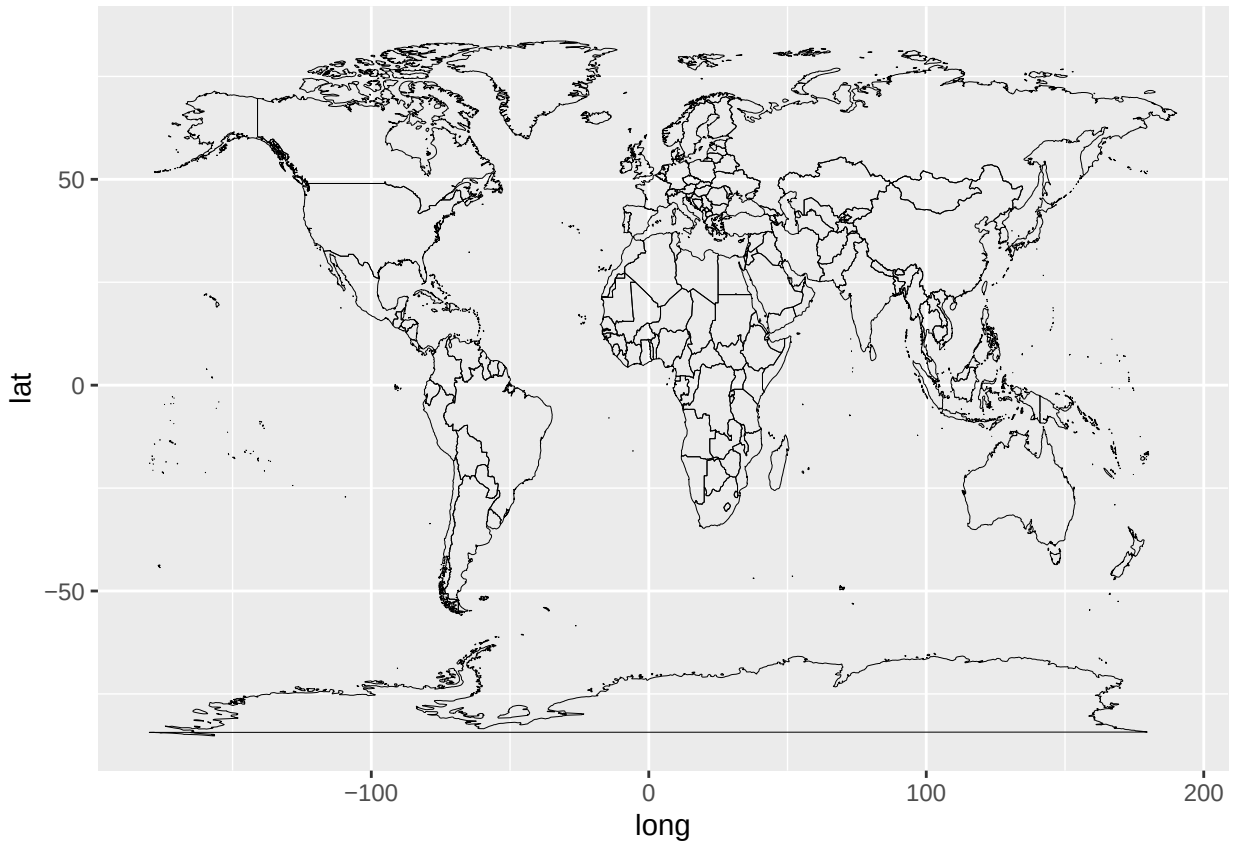
The simple map is just that, simple but it's not very appealing. Adding the `size` graphic aesthetic parameter to the borders call, thinner border line can be used in which improves the appeal of the map.

```
ggplot() +
  borders(size = 0.1)     # thin borders
```

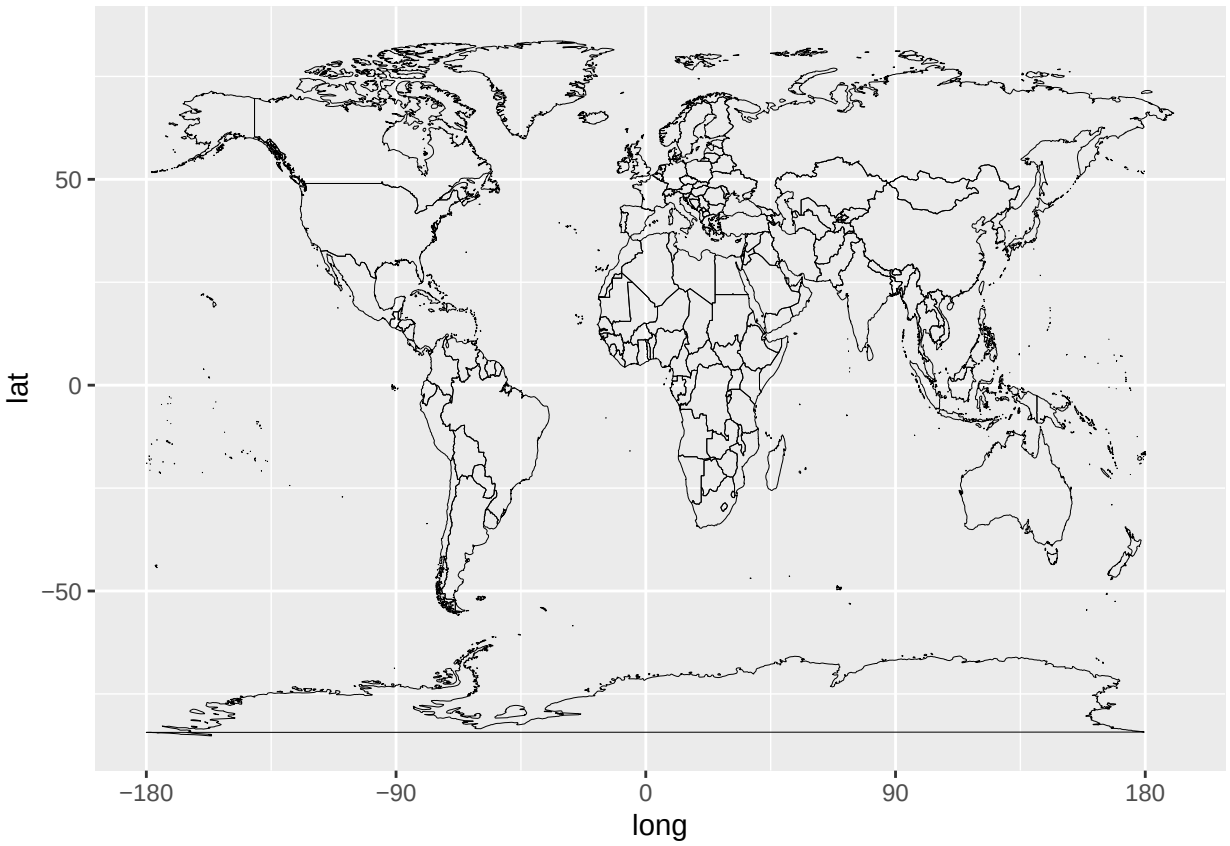


While the thin borders are an improvement, they become harder to see due to the default medium-grey color. Using the `colour` parameter of `borders()` allows the borders to be rendered in different colors. The code below demonstrates using a thin, black border. This fine-line border looks cleaner and more refined than the default border rendering.

```
ggplot() +
  borders(size = 0.1,
          colour = "black")     # thin, black borders
```




```
ggplot() +
  borders(size = 0.1,
          colour = "black") +    # thin, black borders
  scale_x_continuous(breaks = seq(-180, 180, 90))
```


50
0
−50
lat
−180
−90
0
90
180
long