Choropleth maps with tricolore

Jonas Schöley

2018-09-13

Here I demonstrate how to use the `tricolore` library to generate ternary choropleth maps using both `ggplot2` and `leaflet`.

The data

```
library(tricolore)
library(dplyr)
#>
#> Attaching package: 'dplyr'
#> The following objects are masked from 'package:stats':
#>
#>     filter, lag
#> The following objects are masked from 'package:base':
#>
#>     intersect, setdiff, setequal, union

as_tibble(euro_example)
#> # A tibble: 312 x 9
#>    id    name       ed_0to2 ed_3to4 ed_5to8  lf_pri lf_sec lf_ter geometry
#>    <chr> <chr>        <dbl>   <dbl>   <dbl>   <dbl>  <dbl>  <dbl> <list>
#>  1 AT11  Burgenlan…   0.165   0.557   0.279 0.0442   0.268  0.682 <S3: XY>
#>  2 AT12  Niederöst…   0.147   0.551   0.302 0.0562   0.244  0.700 <S3: XY>
#>  3 AT13  Wien         0.169   0.432   0.399 0.00518  0.143  0.852 <S3: XY>
#>  4 AT21  Kärnten      0.106   0.6     0.294 0.0566   0.265  0.671 <S3: XY>
#>  5 AT22  Steiermark   0.14    0.586   0.274 0.0610   0.292  0.647 <S3: XY>
#>  6 AT31  Oberöster…   0.157   0.553   0.291 0.0623   0.331  0.606 <S3: XY>
#>  7 AT32  Salzburg     0.138   0.547   0.315 0.0415   0.249  0.704 <S3: XY>
#>  8 BE31  Prov. Bra…   0.163   0.315   0.522 0        0.148  0.842 <S3: XY>
#>  9 BE32  Prov. Hai…   0.312   0.388   0.3   0.0170   0.204  0.779 <S3: XY>
#> 10 BE33  Prov. Liè…   0.301   0.365   0.334 0.0121   0.211  0.772 <S3: XY>
#> # ... with 302 more rows
```

The data set `euro_example` contains the administrative boundaries for the European NUTS-2 regions in the column `geometry`. This data can be used to plot a choropleth map of Europe using the `sf` package. Each region is represented by a single row. The name of a region is given by the variable `name` while the respective NUTS-2 geocode is given by the variable `id`. For each region some compositional statistics are available: Variables starting with `ed` refer to the relative share of population ages 25 to 64 by educational attainment in 2016 and variables starting with `lf` refer to the relative share of workers by labor-force sector in the European NUTS-2 regions 2016.

Take the first row of the data set as an example: in the Austrian region of “Burgenland” (`id` = `AT11`) 16.5% of the population aged 25–64 had attained an education of “Lower secondary or less” (`ed_0to2`), 55.7% attained “upper secondary” education (`ed_3to4`), and 27.9% attained “tertiary” education. In the very same region 4.4% of the labor-force works in the primary sector, 26.8% in the secondary and

68.2% in the tertiary sector.

The education and labor-force compositions are *ternary*, i.e. made up from three elements, and therefore can be color-coded as the weighted mixture of three primary colors, each primary mapped to one of the three elements. Such a color scale is called a *ternary balance scheme*[1]. This is what `tricolore` does.

`ggplot2` for ternary choropleth maps

Here I show how to create a choropleth map of the regional distribution of education attainment in Europe 2016 using `ggplot2`.

1. Using the `Tricolore()` function, color-code each educational composition in the `euro_example` data set and add the resulting vector of hex-srgb colors as a new variable to the data frame. Store the color key seperately.

```
# color-code the data set and generate a color-key
tric <- Tricolore(euro_example, p1 = 'ed_0to2', p2 = 'ed_3to4', p3 = 'ed_5to8')
#> Warning: Ignoring unknown aesthetics: z
```

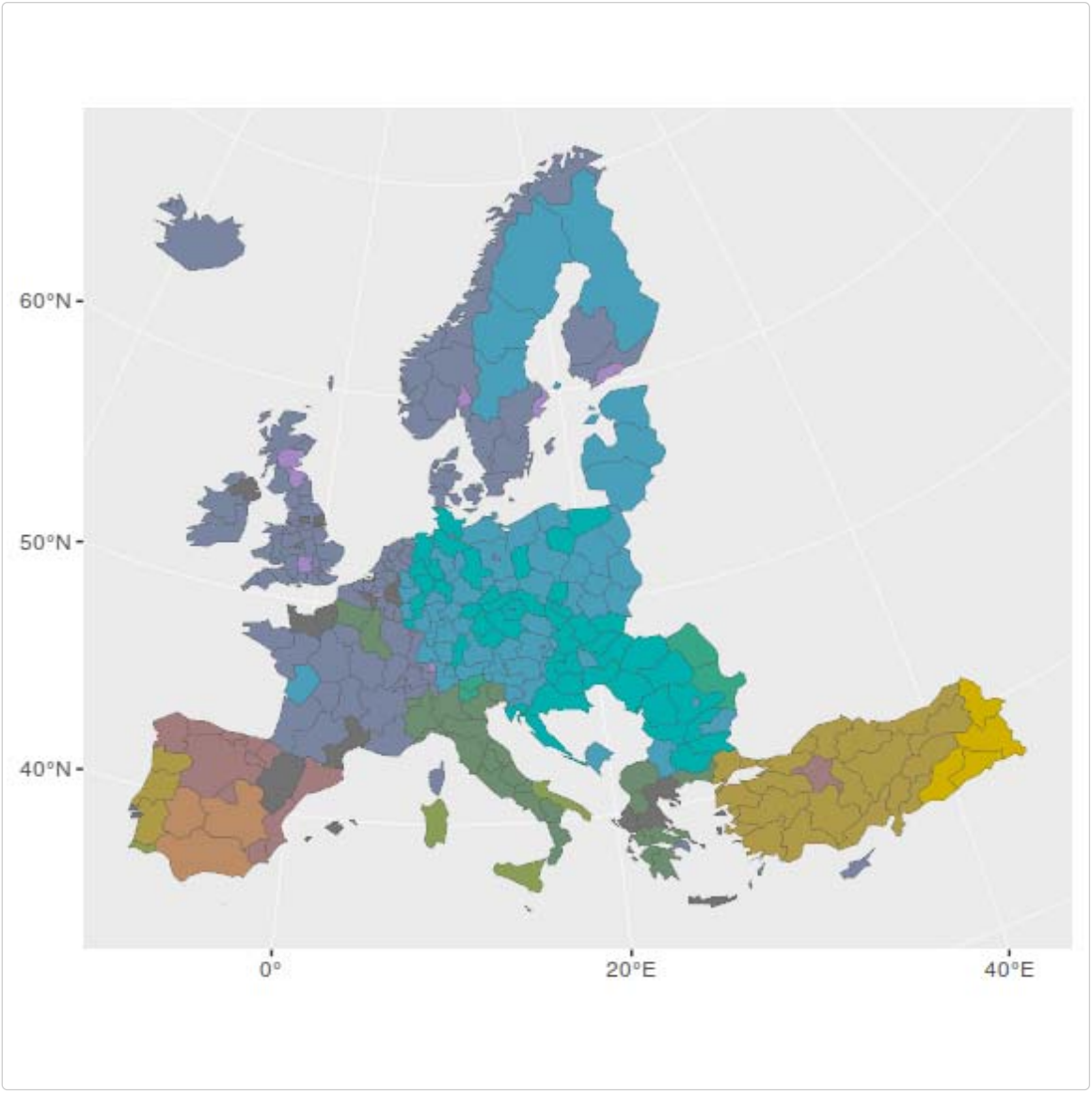
`tric` contains both a vector of color-coded compositions (`tric$rgb`) and the corresponding color key (`tric$key`). We add the vector of colors to the map-data.

```
# add the vector of colors to the `euro_example` data
euro_example$rgb <- tric$rgb
```

2. Using `ggplot2` and the joined color-coded education data and geodata, plot a ternary choropleth map of education attainment in the European regions. Add the color key to the map.

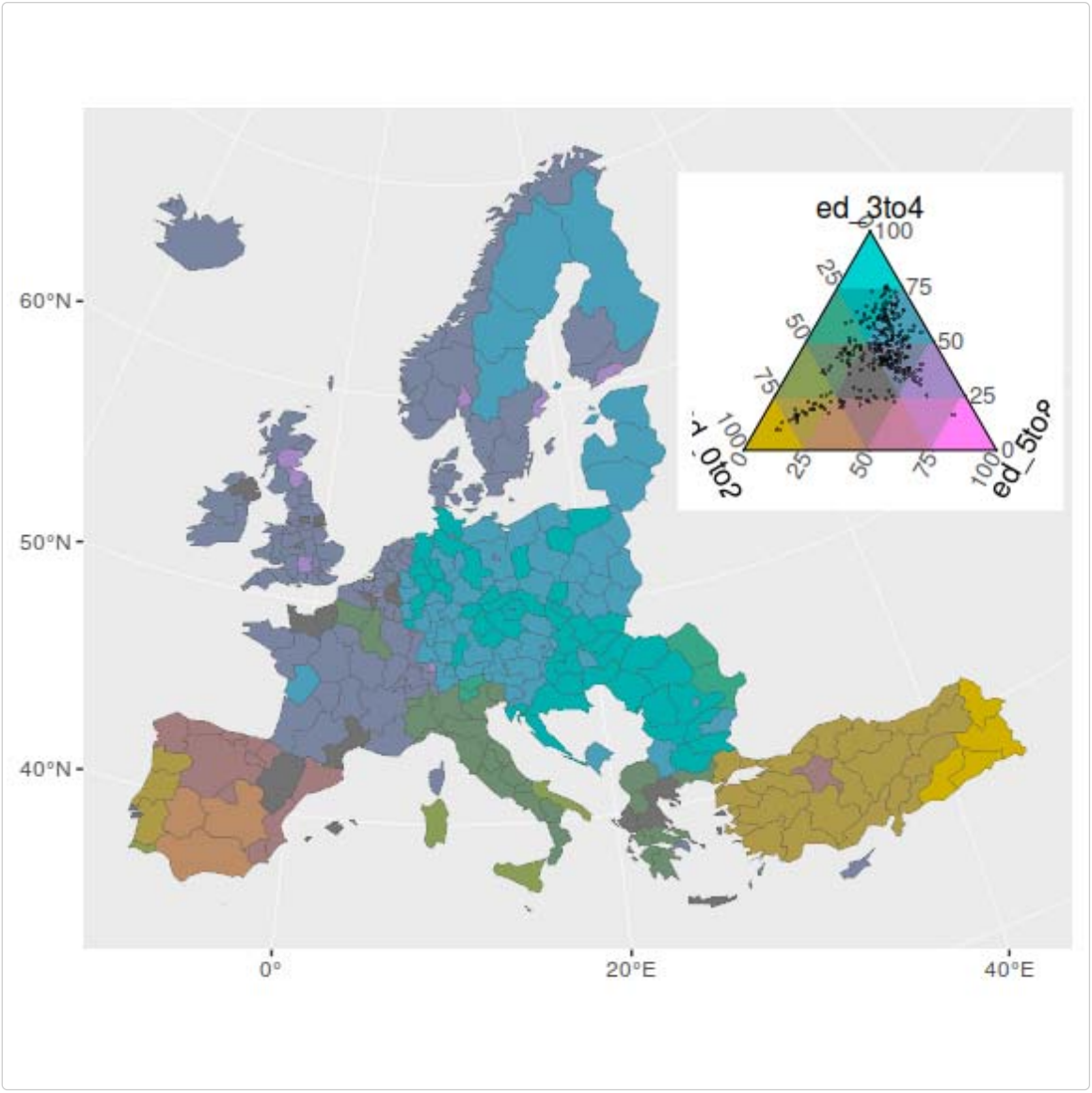
The secret ingredient is `scale_fill_identity()` to make sure that each region is colored according to the value in the `rgb` variable of `euro_educ_map`.

```
library(ggplot2)

plot_educ <-
  # using sf dataframe `euro_example`...
  ggplot(euro_example) +
  # ...draw a polygon for each region...
  geom_sf(aes(fill = rgb), size = 0.1) +
  # ...and color each region according to the color code in the variable `rgb`
  scale_fill_identity()

plot_educ
```



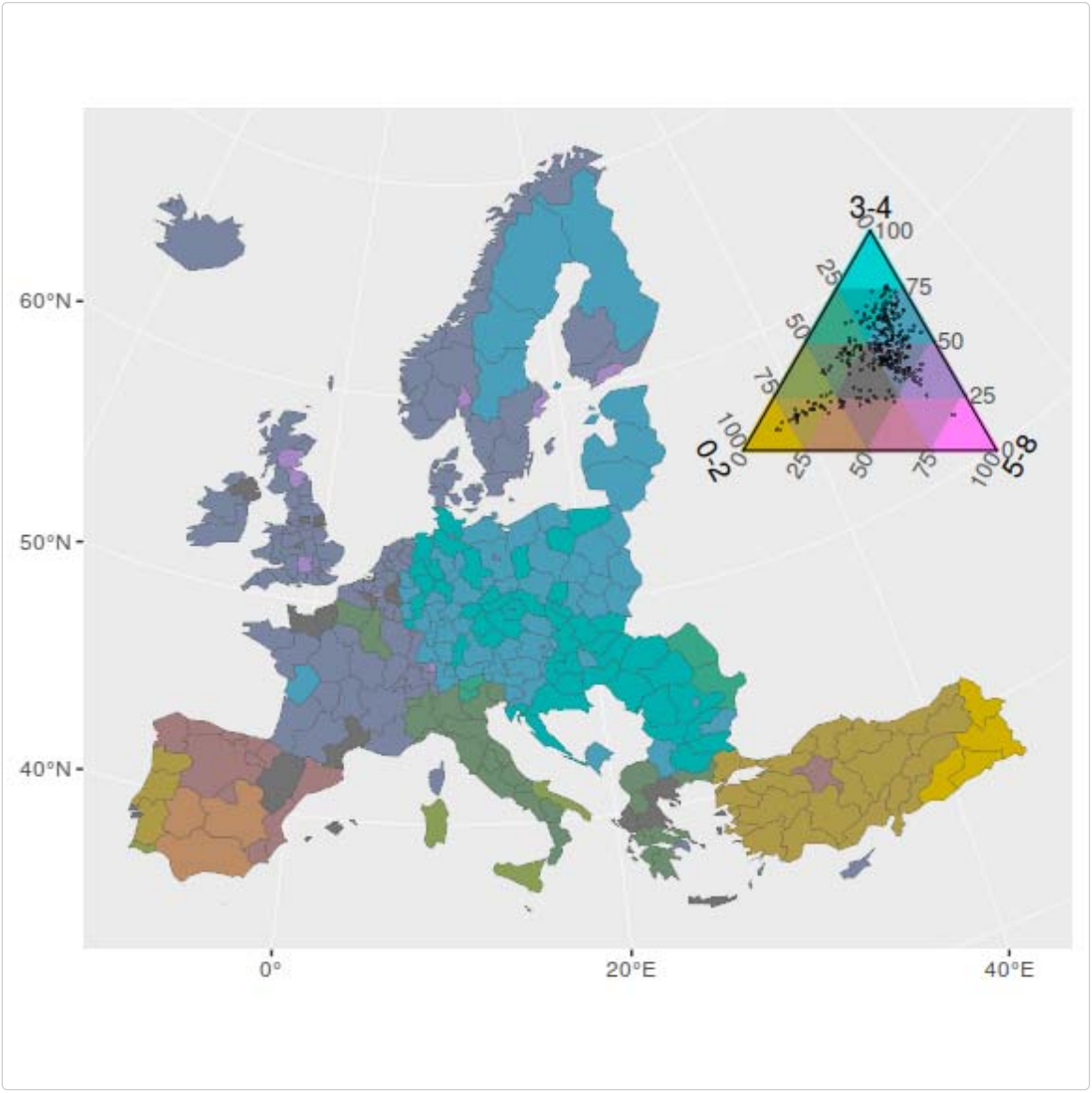
Using `annotation_custom()` and `ggplotGrob` we can add the color key produced by `Tricolore()` to the map. Internally, the color key is produced with the `ggtern` package. In order for it to render correctly we need to load `ggtern` *after* loading `ggplot2`. Don't worry, the `ggplot2` functions still work.

```
library(ggtern)
#> --
#> Remember to cite, run citation(package = 'ggtern') for further info.
#> --
#>
#> Attaching package: 'ggtern'
#> The following objects are masked from 'package:ggplot2':
#>
#>     %+%, aes, annotate, calc_element, ggplot, ggplotGrob,
#>     ggplot_build, ggplot_gtable, ggsave, layer_data, theme,
#>     theme_bw, theme_classic, theme_dark, theme_gray, theme_light,
#>     theme_linedraw, theme_minimal, theme_void
plot_educ +
  annotation_custom(
    ggplotGrob(tric$key),
    xmin = 55e5, xmax = 75e5, ymin = 8e5, ymax = 80e5
  )
#> Warning: Removed 1 rows containing missing values (geom_point).
```



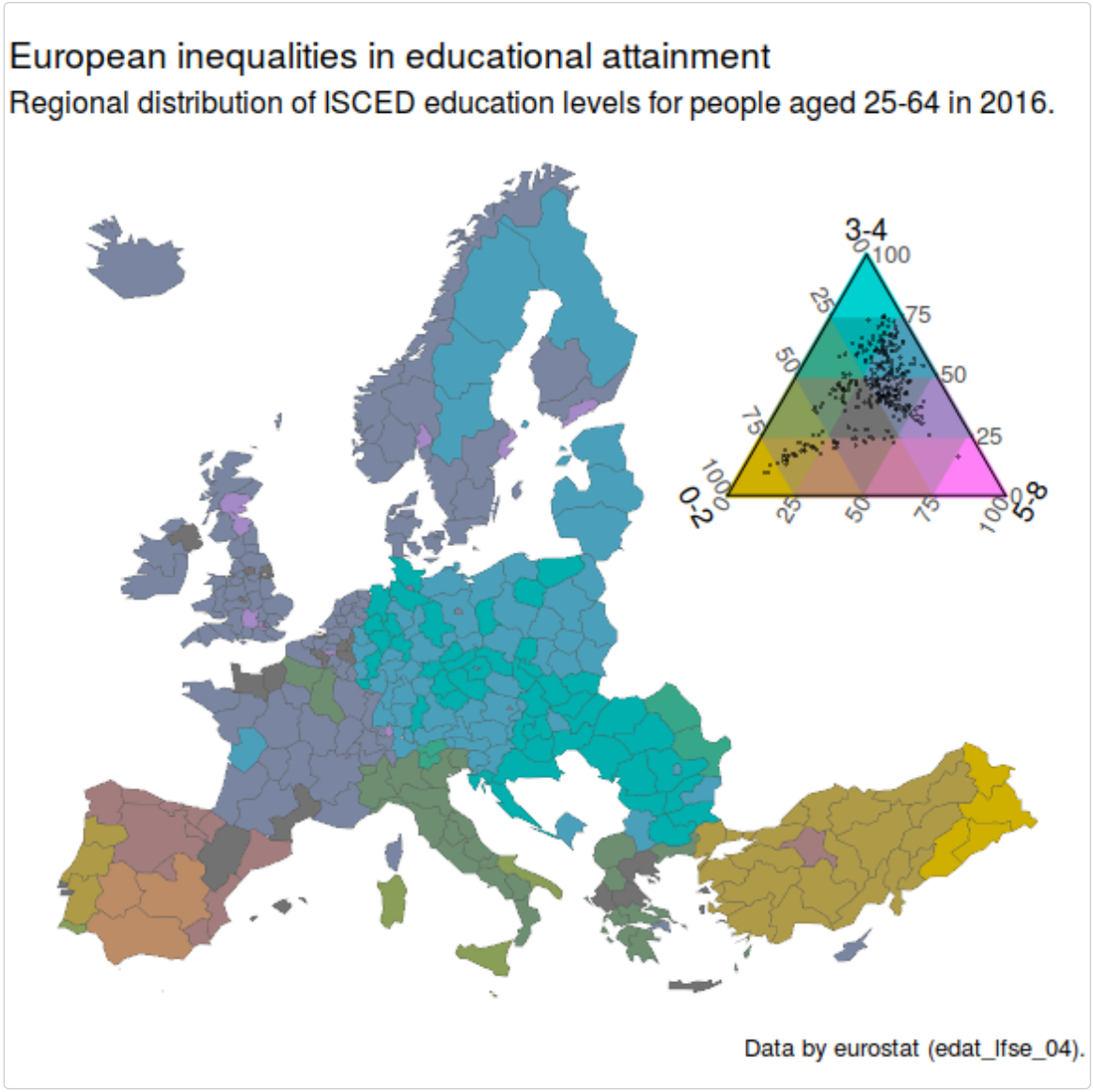
Because the color key behaves just like a `ggplot2` plot we can change it to our liking.

```
plot_educ <-
  plot_educ +
  annotation_custom(
    ggplotGrob(tric$key +
                 theme(plot.background = element_rect(fill = NA, color = NA)) +
                 labs(L = '0-2', T = '3-4', R = '5-8')),
    xmin = 55e5, xmax = 75e5, ymin = 8e5, ymax = 80e5
  )
#> Warning: Removed 1 rows containing missing values (geom_point).
plot_educ
```



Some final touches…

```
plot_educ +
  theme_void() +
  coord_sf(datum = NA) +
  labs(title = 'European inequalities in educational attainment',
       subtitle = 'Regional distribution of ISCED education levels for people aged 25-64 in
2016.',
       caption = 'Data by eurostat (edat_lfse_04).')
```



`leaflet` for ternary choropleth maps

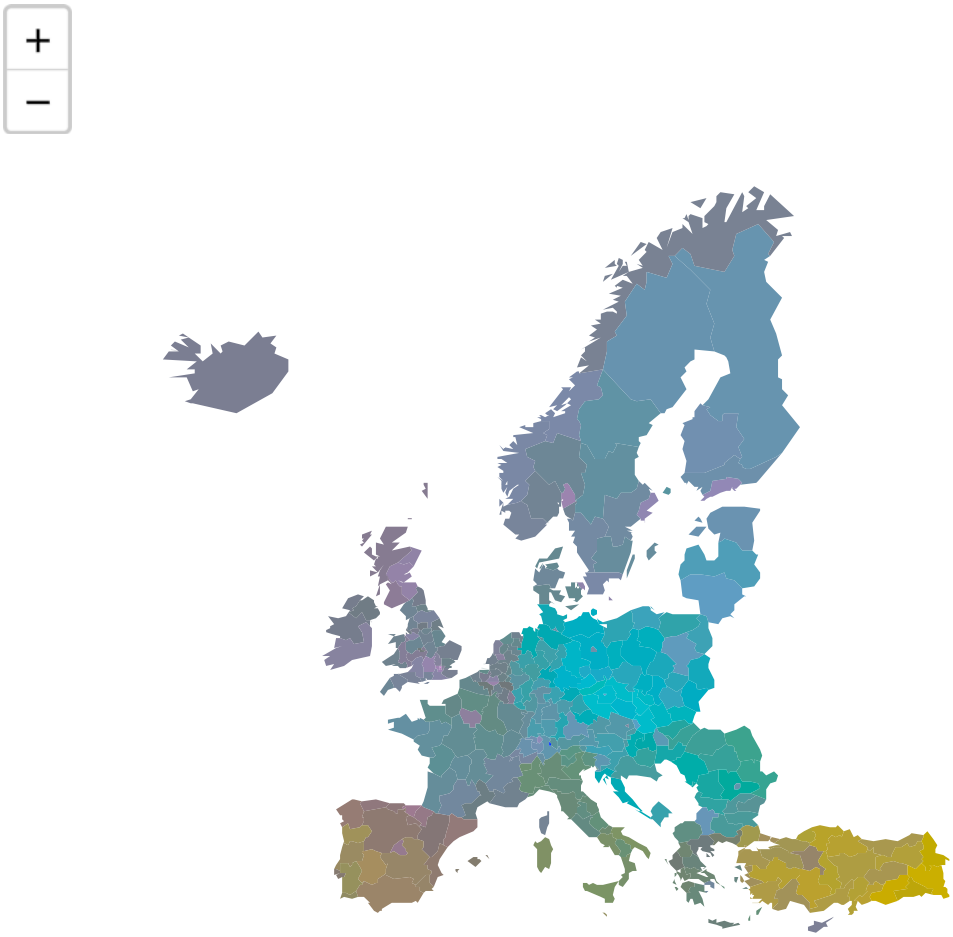
The `ggplot2` example above is easily adapted to `leaflet`. This time I use a continuous color scale.

```
# color-code the data set and generate a color-key
tric <- Tricolore(euro_example, p1 = 'ed_0to2', p2 = 'ed_3to4', p3 = 'ed_5to8',
                  breaks = Inf)
#> Warning: Ignoring unknown aesthetics: z

# add the vector of colors to the `euro_example` data
euro_example$rgb <- tric$rgb
```

`leaflet` requires geodata in spherical coordinates (longitude-latitude format). Therefore I reproject the data to a suitable crs using the `sf` package.

```
library(sf)
#> Linking to GEOS 3.5.1, GDAL 2.1.2, proj.4 4.9.3
library(leaflet)

euro_example %>%
```

```
  st_transform(crs = 4326) %>%
  leaflet() %>%
  addPolygons(smoothFactor = 0.1, weight = 0,
              fillColor = euro_example$rgb,
              fillOpacity = 1)
```

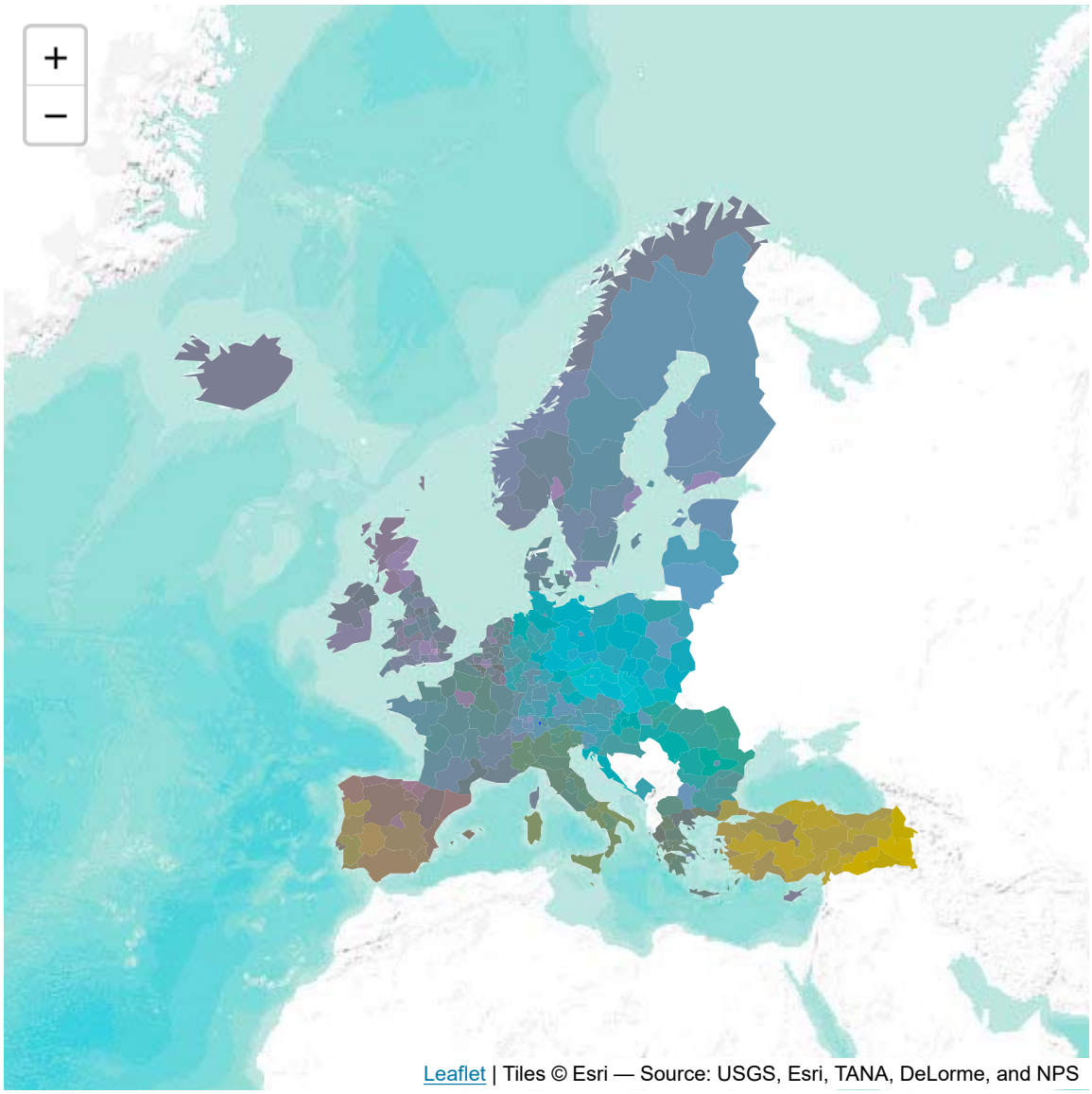


Leaflet

Adding a background map gives geographical context to the map. I also add a mouse pop-up of the actual data.

```
euro_example %>%
  st_transform(crs = 4326) %>%
  leaflet() %>%
  addProviderTiles(providers$Esri.WorldTerrain) %>%
  addPolygons(smoothFactor = 0.1, weight = 0,
              fillColor = euro_example$rgb,
              fillOpacity = 1,
              popup =
                paste0(
                  '<b>', euro_example$name, '</b></br>',
                  'Primary: ',
                  formatC(euro_example$ed_0to2*100,
                          digits = 1, format = 'f'), '%</br>',
```

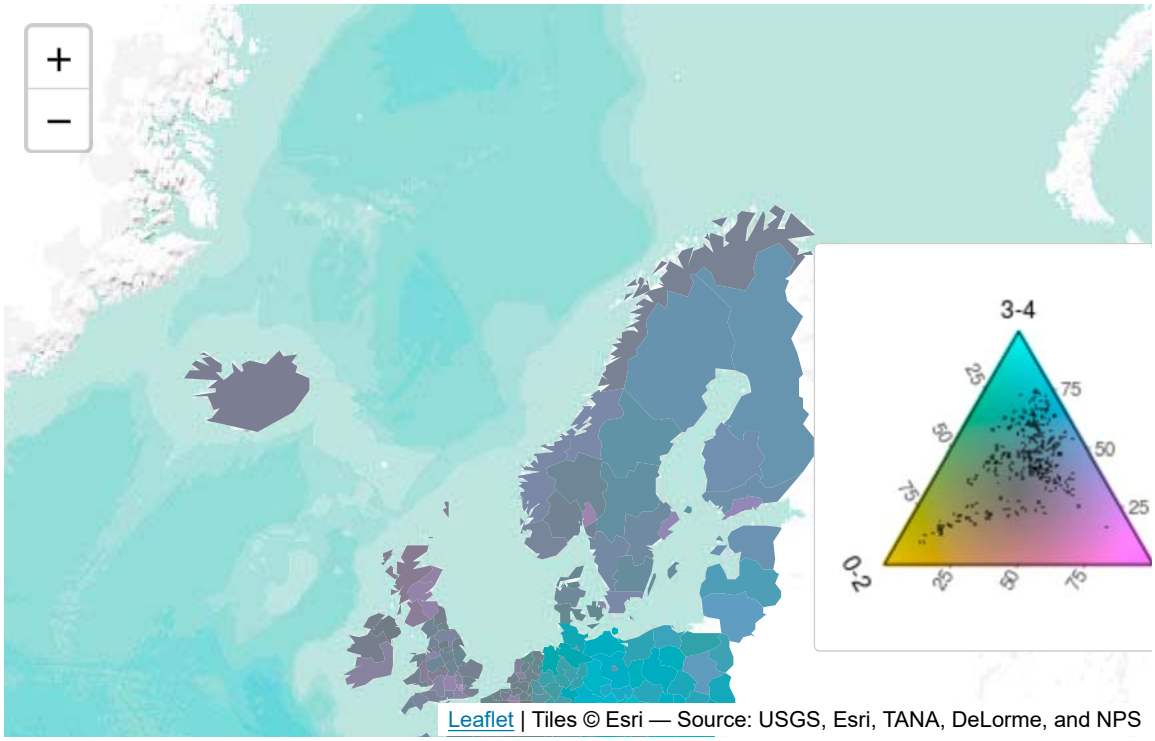
```
                'Secondary: ',
                formatC(euro_example$ed_3to4*100,
                        digits = 1, format = 'f'), '%</br>',
                'Tertiary: ',
                formatC(euro_example$ed_5to8*100,
                        digits = 1, format = 'f'), '%</br>'
              )
  )
```



Adding the legend to the leaflet map requires a bit of a hack.

```
makePlotURI <- function(expr, width, height, ...) {
  pngFile <- shiny::plotPNG(function() { expr }, width = width, height = height, ...)
  on.exit(unlink(pngFile))

  base64 <- httpuv::rawToBase64(readBin(pngFile, raw(1), file.size(pngFile)))
  paste0("", base64)
}

legend_symbol <- makePlotURI({
  print(tric$key +
          theme(plot.background = element_rect(fill = NA, color = NA)) +
```

```
          labs(L = '0-2', T = '3-4', R = '5-8'))
}, 200, 200, bg = "transparent")
#> Warning: Removed 1 rows containing missing values (geom_point).

df <- data.frame(
  lng = 30,
  lat = 70,
  plot = legend_symbol,
  stringsAsFactors = FALSE
)

euro_example %>%
  st_transform(crs = 4326) %>%
  leaflet() %>%
  addProviderTiles(providers$Esri.WorldTerrain) %>%
  addPolygons(smoothFactor = 0.1, weight = 0,
              fillColor = euro_example$rgb,
              fillOpacity = 1,
              popup =
                paste0(
                  '<b>', euro_example$name, '</b></br>',
                  'Primary: ',
                  formatC(euro_example$ed_0to2*100,
                          digits = 1, format = 'f'), '%</br>',
                  'Secondary: ',
                  formatC(euro_example$ed_3to4*100,
                          digits = 1, format = 'f'), '%</br>',
                  'Tertiary: ',
                  formatC(euro_example$ed_5to8*100,
                          digits = 1, format = 'f'), '%</br>'
                )
  ) %>%
  addMarkers(data = df, icon = ~icons(plot))
#> Assuming "lng" and "lat" are longitude and latitude, respectively
```



Literature

Brewer, C. A. (1994). Color Use Guidelines for Mapping and Visualization. In A. M. MacEachren & D. R. F. Taylor (Eds.), Visualization in Modern Cartography (pp. 123–147). Oxford, UK: Pergamon.

Dorling, D. (2012). The Visualization of Spatial Social Structure. Chichester, UK: Wiley. Retrieved from https://sasi.group.shef.ac.uk/thesis/prints.html

---

1. See for example Dorling (2012) and Brewer (1994).↩