MATH 118: Notes I

custom colors, labels

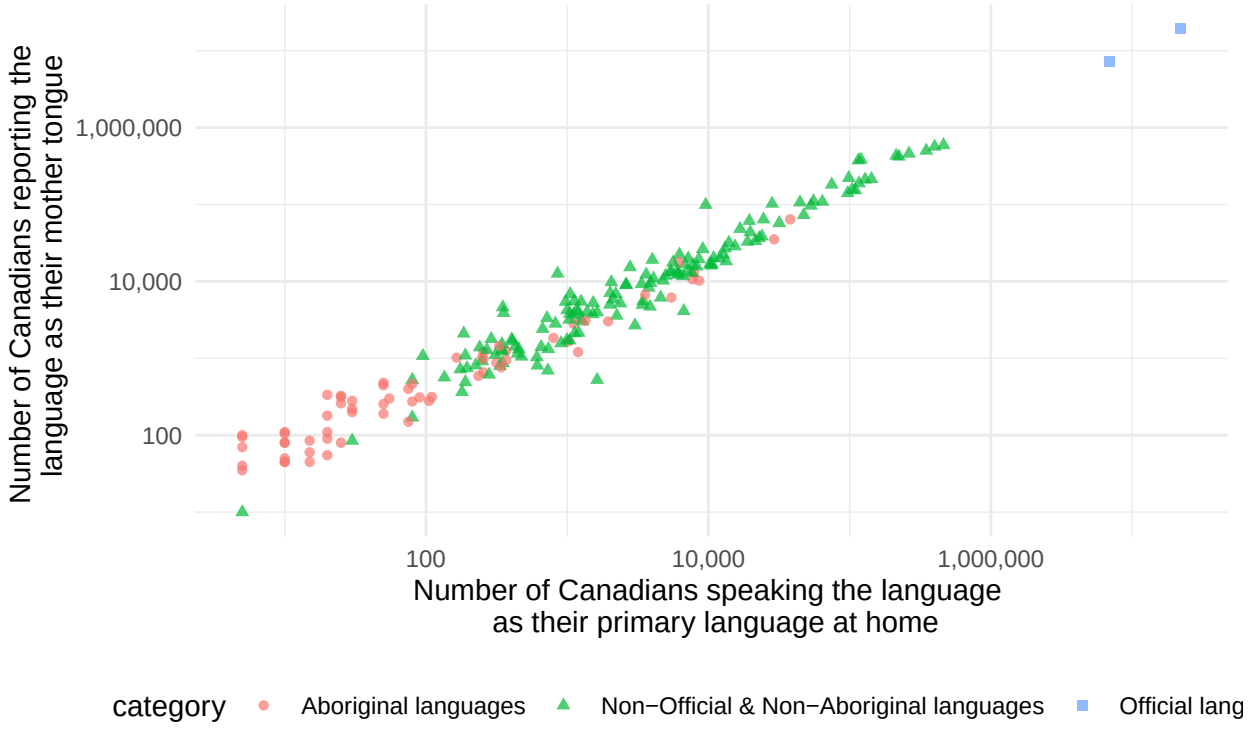
Before we start, check out: https://native-land.ca/

```
#LOAD PACKAGES
library(tidyverse)
library(kableExtra)

#LOAD DATA SETS
can_lang <- read.csv("https://raw.githubusercontent.com/ttimbers/canlang/master/inst/extdata/can_lang.cs
```

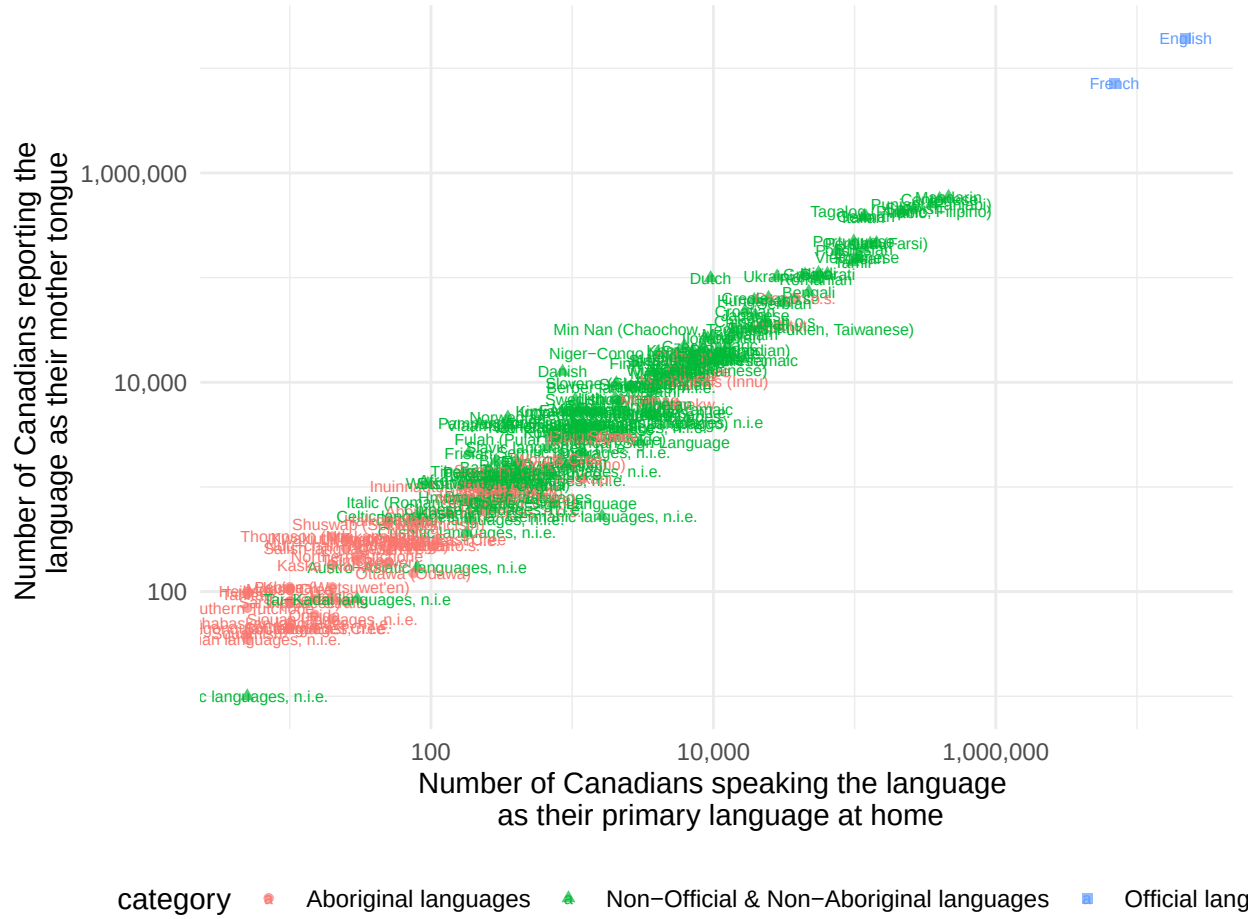
```
plot1 <- can_lang %>%
  filter(most_at_home >0) %>%
  filter(mother_tongue >0) %>%
ggplot( aes(x=most_at_home, y=mother_tongue, color=category, shape=category)) +
  geom_point(alpha=0.7) +
  scale_y_log10(name = "Number of Canadians reporting the \n language as their mother tongue",
                labels = scales::comma) +
  scale_x_log10(name = "Number of Canadians speaking the language \n as their primary language at home"
                labels = scales::comma) +
  theme_minimal() +
  theme(legend.position="bottom")

plot1
```



Adding Labels (all points)

```
plot1 + geom_text(aes(label=language), size=2)
```



Um... a bit messy.

We could play with `geom_text` options: `nudge_x` and `nudge_y` but realistically this graph is just too crowded for all these labels to make sense on this graph.

Suppose we are only interested in the Cree languages - a dialect of Algonquin languages. There are several listed in this dataset:

- Plains Cree
- Moose Cree
- Swampy Cree (sometimes divided into Eastern and Western Swampy Cree, but not in this dataset)
- Woods Cree
- Northern East Cree
- Southern East Cree
- Atikamekw
- Montagnais (Innu)
- Naskapi
- Cree, n.o.s. (not otherwise specified)

See them on a map here:

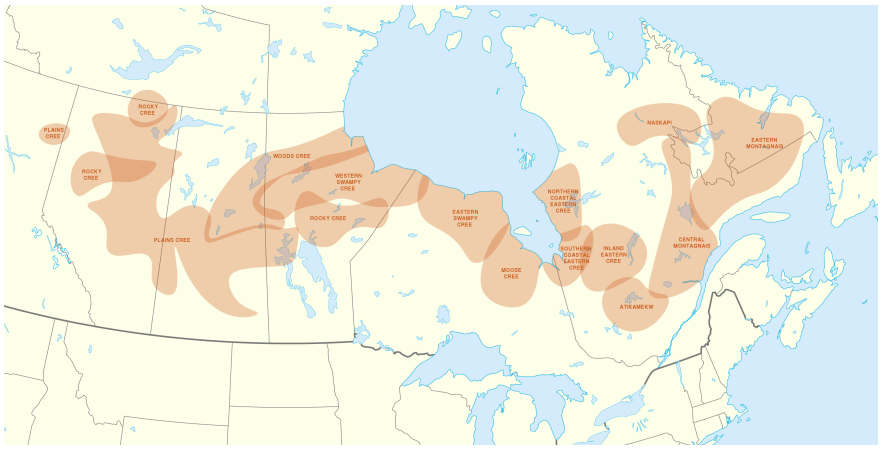


Figure 1: By Noahedits - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=84799257

As with many aboriginal languages in Canada, there is a loss of the Cree languages among many speakers over the last few centuries.

Let's say we only want to label on the graph the Cree languages on above plot. We need to create a new column which only contains the Cree languages, and is blank for all the other languages.

Adding Labels (subset of points)

```
#ifelse(condition, if condition is met do this, else do this)

can_lang <- can_lang %>%
  mutate(language_cree = ifelse(language == "Plains Cree"|
                                  language =="Moose Cree"|
                                  language == "Swampy Cree"|
                                  language == "Woods Cree"|
                                  language =="Northern East Cree"|
                                  language =="Southern East Cree"|
                                  language=="Atikamekw"|
                                  language == "Montagnais (Innu)"|
                                  language =="Naskapi"|
                                  language=="Cree, n.o.s.",language,""))

#check it out in the can_lang dataset what we have done!
can_lang %>%
  slice(120:130) %>%
  kbl() %>%
  kable_styling(latex_options="scale_down") %>%
  kable_styling(latex_options = "HOLD_position")
```

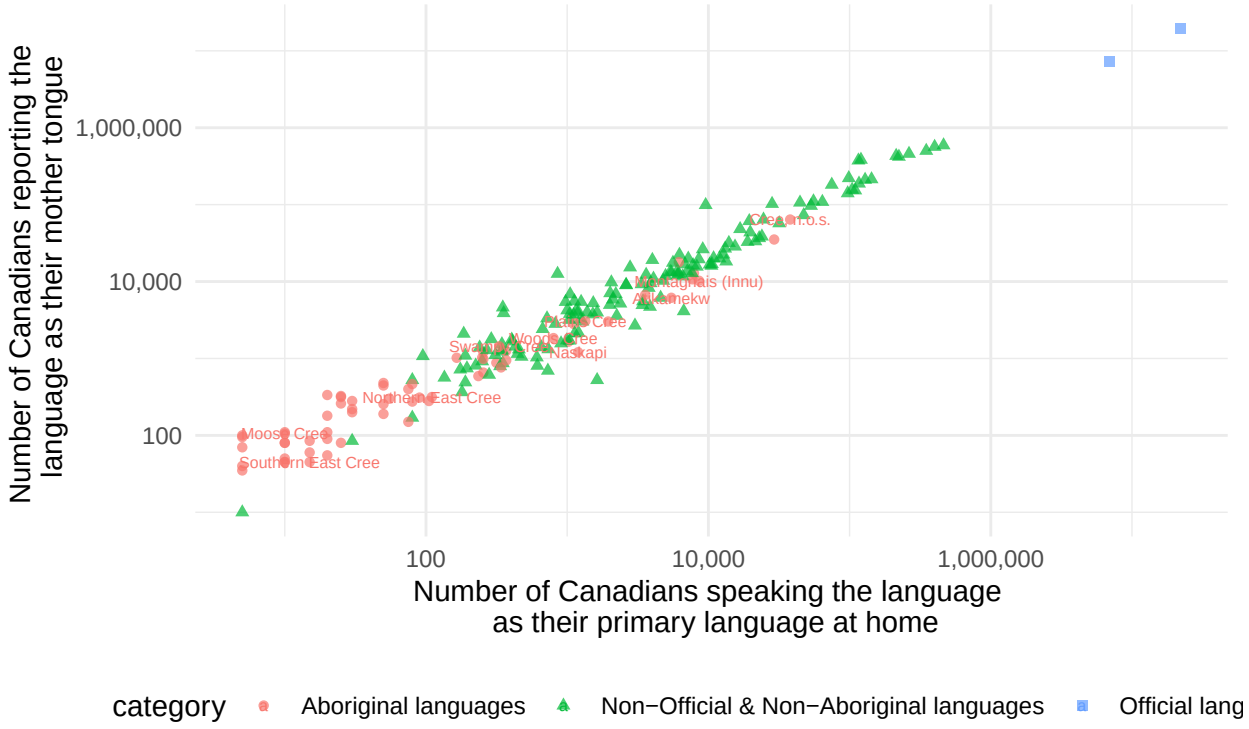
category	language	mother_tongue	most_at_home	most_at_work	lang_known	language_cree
Aboriginal languages	Mi'kmaq	6690	3565	915	9025	
Aboriginal languages	Michif	465	80	10	1210	
Non-Official & Non-Aboriginal languages	Min Dong	1230	345	30	1045	
Non-Official & Non-Aboriginal languages	Min Nan (Chaochow, Teochow, Fukien, Taiwanese)	31800	13965	565	42840	
Aboriginal languages	Mohawk	985	255	30	2415	
Non-Official & Non-Aboriginal languages	Mongolian	1575	905	10	2095	
Aboriginal languages	Montagnais (Innu)	10235	8585	2055	11445	Montagnais (Innu)
Aboriginal languages	Moose Cree	105	10	0	195	Moose Cree
Aboriginal languages	Naskapi	1205	1195	370	1465	Naskapi
Non-Official & Non-Aboriginal languages	Nepali	18275	13375	195	21385	
Non-Official & Non-Aboriginal languages	Niger-Congo languages, n.i.e.	19135	4010	30	40760	

Using `geom_text`

Note that we need to recreate the plot since the dataset `can_lang` has changed (we added column `language_cree`)

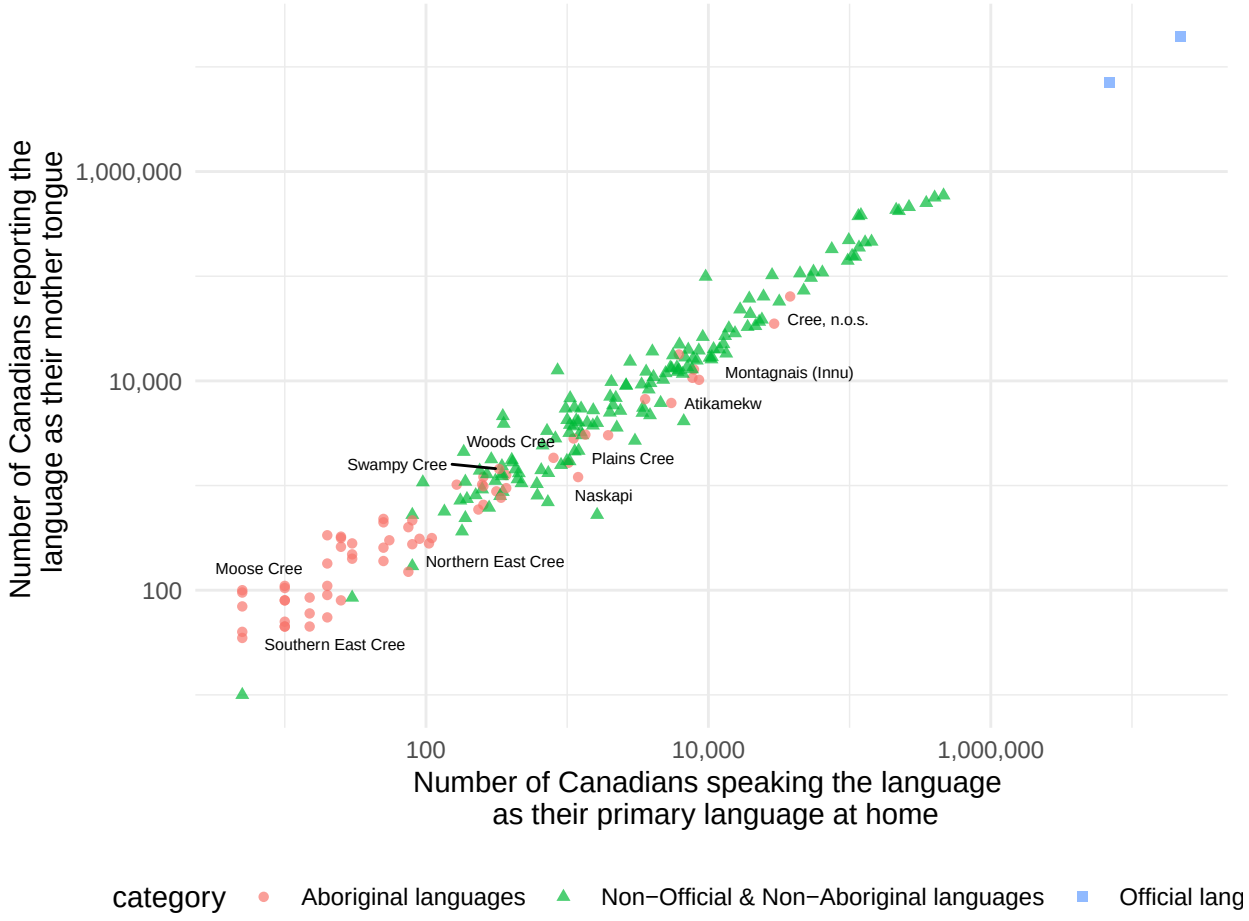
```
plot1 <- can_lang %>%
  filter(most_at_home >0) %>%
  filter(mother_tongue >0) %>%
ggplot( aes(x=most_at_home, y=mother_tongue, color=category, shape=category)) +
  geom_point(alpha=0.7) +
  scale_y_log10(name = "Number of Canadians reporting the \n language as their mother tongue",labels = s
  scale_x_log10(name = "Number of Canadians speaking the language \n as their primary language at home"
  theme_minimal() +
  theme(legend.position="bottom")

plot1 + geom_text(aes(label=language_cree), size=2)
```



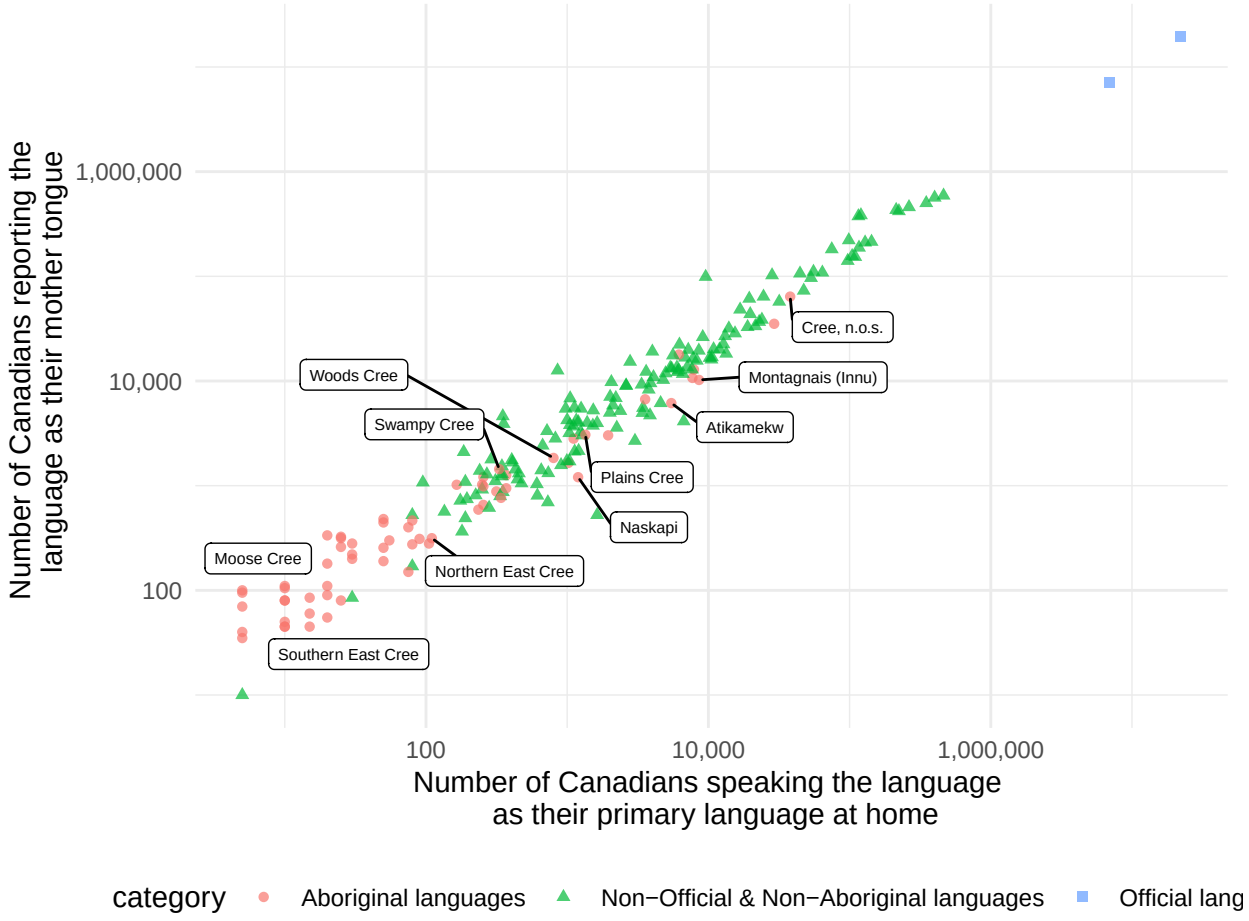
Using `geom_text_repel` (requires `ggrepel` package)

```
library(ggrepel)
plot1 + geom_text_repel(aes(label=language_cree), size=2, color="black", max.overlaps=50)
```



Using `geom_label_repel` (requires `ggrepel` package)

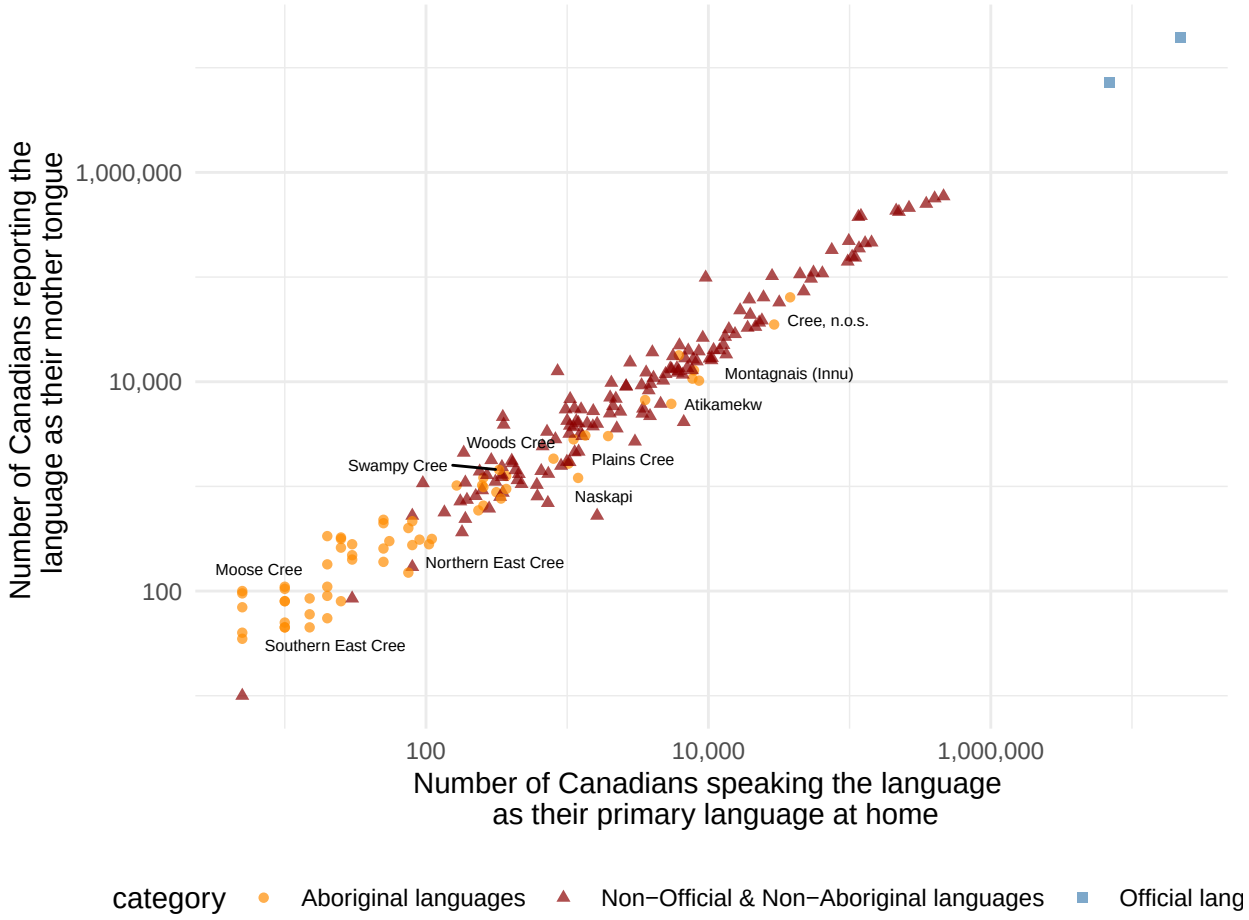
```
plot1 + geom_label_repel(aes(label=language_cree), size=2, color="black", max.overlaps=50)
```



Customizing Colors

Using Custom Colors

```
plot1 +
  scale_color_manual(values=c("darkorange","darkred","steelblue")) +
  geom_text_repel(aes(label=language_cree), size=2, color="black", max.overlaps=50)
```

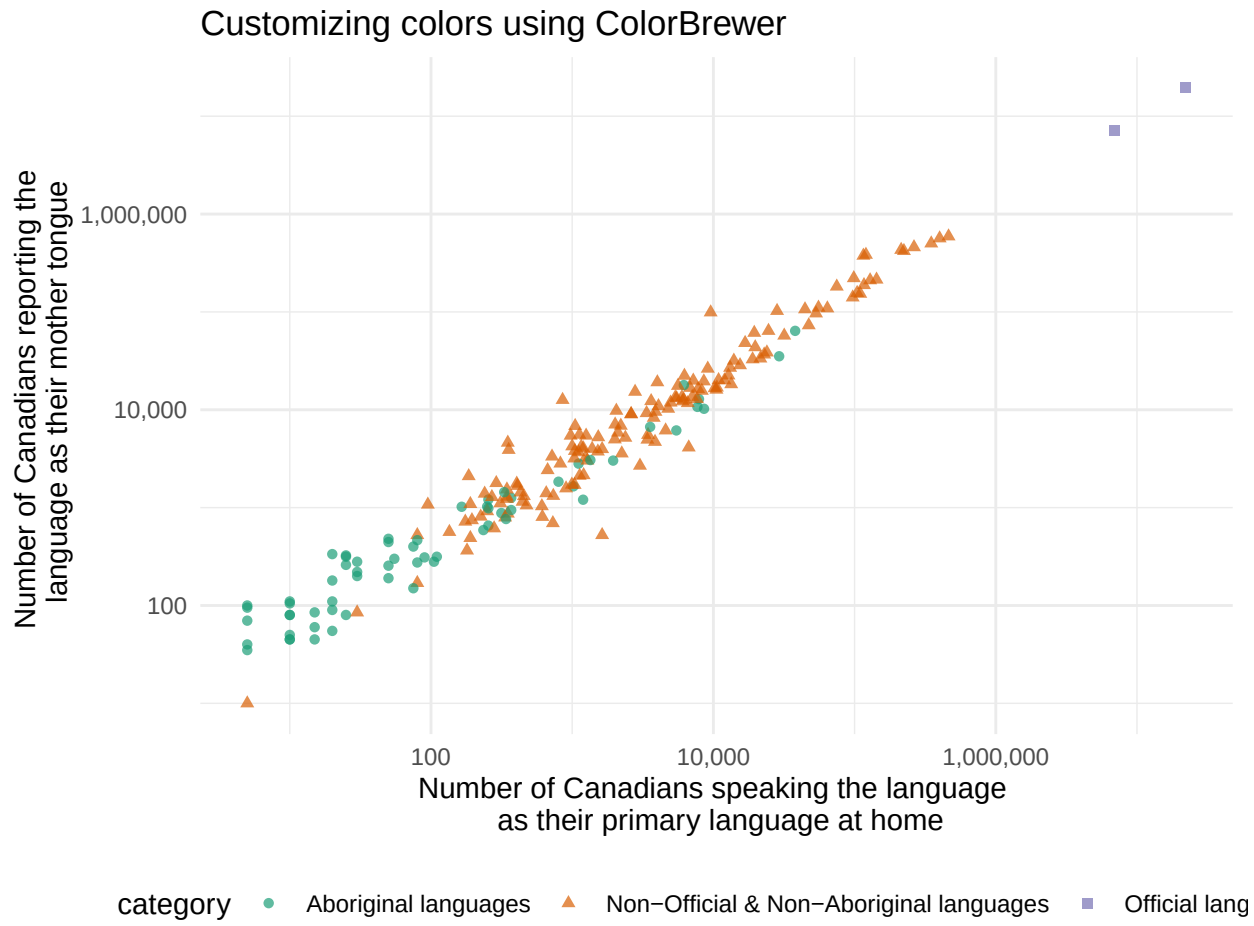


Using Color Brewer

```
library(RColorBrewer)
```

All palettes:

```
plot1 +
  scale_color_brewer(palette = "Dark2") +
  ggtitle("Customizing colors using ColorBrewer")
```



A Note About Color Blindness

It is important to consider how a color palette will look to people with common forms of color blindness: - avoid red-green contrasts - check your plots with systems that simulate color blindness.

```
library(colorBlindness)
displayAllColors(brewer.pal(name = "Set1", n = 8))
```


```
displayAllColors(c("darkorange","darkred","steelblue"))
```