Constructing a bar plot

The goal here is to use the data from the Stata file `IOT.dta` to create a horizontal bar graph comparing two variables (domestic and foreign), grouped by a categorical variable (producers).

To accomplish this, we will use the following packages:

```
here::i_am("material/Session-03-Example-Barplot.qmd") # Adjust to your case
library(here)
library(haven) # to import stata files
library(dplyr) # to do data manipulation
library(tidyr) # to use do data wrangling
library(ggplot2) # for visualization
```

We first have a look at the data set:

```
iot_raw <- haven::read_dta(here("material/data/IOT.dta"))
iot_raw <- as_tibble(iot_raw)
head(iot_raw)
```

```
# A tibble: 6 x 6
   year producer supplier producers          domestic foreign
  <dbl> <chr>    <chr>    <dbl+lbl>             <dbl>   <dbl>
1  2009 10       10       10 [Food products]     80.0    20.0
2  2009 10       11       10 [Food products]     80.0    20.0
3  2009 10       12       10 [Food products]     80.0    20.0
4  2009 10       13       10 [Food products]     80.0    20.0
5  2009 10       14       10 [Food products]     80.0    20.0
6  2009 10       15       10 [Food products]     80.0    20.0
```

Note that some of the data has a somehow weird data type, which is due to the specific features of the Stata data. In most cases this is not a problem, but if you work with the particular

variable, its a good idea to transform it into a data type you know how to handle. In our case, this concerns the variable `producers`. Since we are less interested in the numeric coding, but just want to keep the labels, we use `haven::as_factor()` together with `as.characer()` because working with character vectors is easier than with factors.

Also, we can make the data tidy, to get one variable `production`, and one variable about the `origin` of the produced goods:

```
iot_tidy <- iot_raw %>%
  mutate(
    producers = as_factor(producers),
    producers = as.character(producers)
    ) %>%
  filter(year==2009) %>%
  pivot_longer(
    cols = c("domestic", "foreign"),
    names_to = "origin",
    values_to = "production")
head(iot_tidy)
```

```
# A tibble: 6 x 6
   year producer supplier producers     origin   production
  <dbl> <chr>    <chr>    <chr>         <chr>         <dbl>
1  2009 10       10       Food products domestic       80.0
2  2009 10       10       Food products foreign        20.0
3  2009 10       11       Food products domestic       80.0
4  2009 10       11       Food products foreign        20.0
5  2009 10       12       Food products domestic       80.0
6  2009 10       12       Food products foreign        20.0
```

Input-Output tables contain information about the flows of intermediate goods from one actor to another. This is not relevant for our case, so we sum up the values for producers and suppliers, and group by sector and year:

```
iot_tidy_agg <- iot_tidy %>%
  summarise(
    production=sum(production),
    .by = c("year", "producers", "origin"))
head(iot_tidy_agg)
```

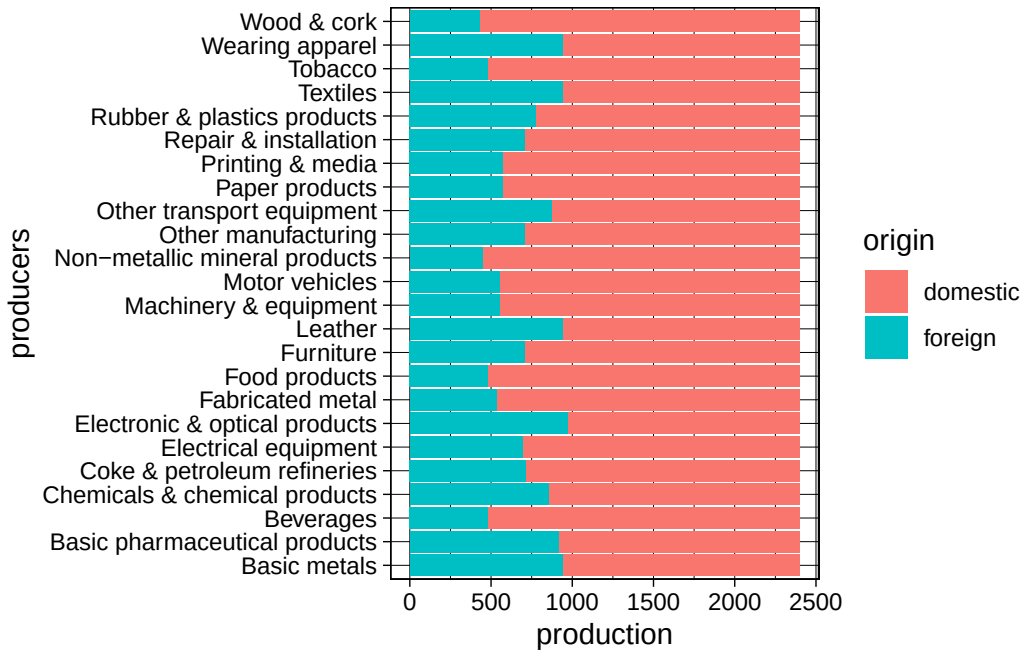
```
# A tibble: 6 x 4
```

```
  year producers     origin   production
 <dbl> <chr>         <chr>         <dbl>
1  2009 Food products domestic      1919.
2  2009 Food products foreign        481.
3  2009 Beverages     domestic      1919.
4  2009 Beverages     foreign        481.
5  2009 Tobacco       domestic      1919.
6  2009 Tobacco       foreign        481.
```

- Which years?
- What are the scales?

Then we take a first very rough look at the figure that would come out from the data. Since we want to create a barplot we use the geom `geom_bar` and since we wish to draw the absolute values from the data we use `stat = "identity"`:

```
ggplot(
  data = iot_tidy_agg,
  mapping = aes(y=producers, x=production, fill=origin)
  ) +
  geom_bar(stat = "identity") +
  theme_linedraw()
```



This is not too bad and shows that the visualization makes sense. But what if we do want

to order the appearance by the value of domestic production? Unfortunately, this is not so straightforward because I do not know how to order by only the value of one group. But we can do this by first creating the order separately, and then create an ordered factor variable:

```
order_producers <- iot_tidy_agg %>%
  filter(origin=="domestic") %>%
  arrange(production) %>%
  pull(producers)
head(order_producers) # a vector of producers ordered by domestic output
```

```
[1] "Electronic & optical products" "Basic metals"
[3] "Textiles"                      "Wearing apparel"
[5] "Leather"                       "Basic pharmaceutical products"
```

Then sort the producers by `order_producers`:

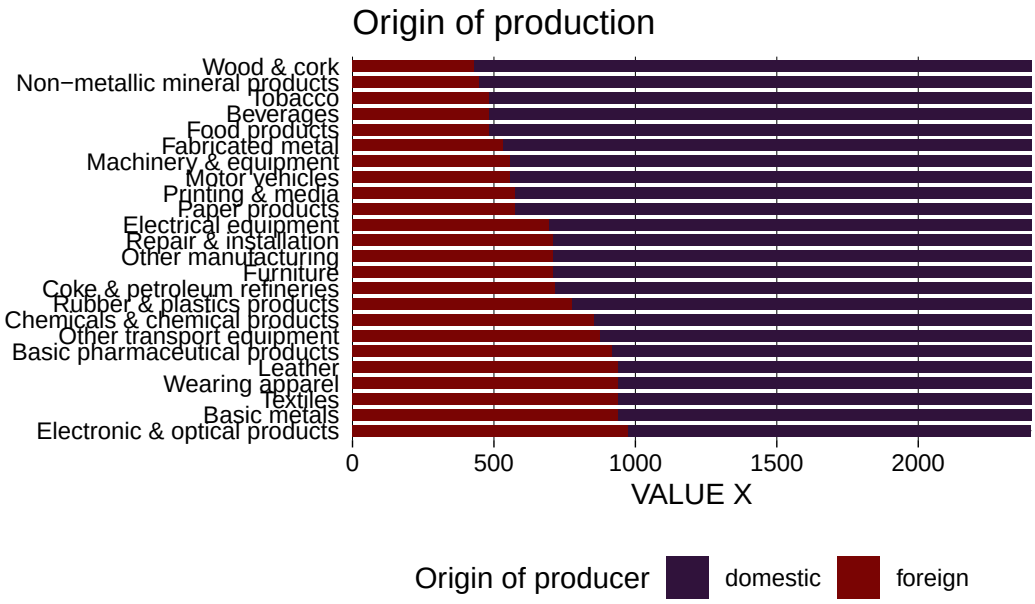
```
iot_tidy_agg <- iot_tidy_agg %>%
  mutate(producers = factor(producers, levels = order_producers))
head(iot_tidy_agg)
```

```
# A tibble: 6 x 4
   year producers     origin   production
  <dbl> <fct>         <chr>         <dbl>
1  2009 Food products domestic      1919.
2  2009 Food products foreign        481.
3  2009 Beverages     domestic      1919.
4  2009 Beverages     foreign        481.
5  2009 Tobacco       domestic      1919.
6  2009 Tobacco       foreign        481.
```

The following code then adds a number of additional specifications that make it prettier:

```
iot_plot <- ggplot(
  data = iot_tidy_agg,
  mapping = aes(
    y=producers,
    x=production,
    fill=origin)
  ) +
  scale_x_continuous(expand = expansion()) +
```

```
  scale_fill_viridis_d(option = "H", name = "Origin of producer") +
  geom_bar(stat = "identity", width = 0.75) +
  labs(
    title = "Origin of production",
    x = "VALUE X", caption = "Data: Destatis (2024)."
  ) +
  theme_linedraw() +
  theme(
    legend.position = "bottom",
    panel.grid.minor.x = element_blank(),
    axis.title.y = element_blank(),
    axis.ticks.y = element_blank(),
    panel.border = element_blank())
iot_plot
```



Now you can save the plot and control its aspect ratio:

```
ggsave(
  plot = iot_plot,
  filename = here("material/Day3-IOT-plot.pdf"),
  width = 6, height = 5)
```