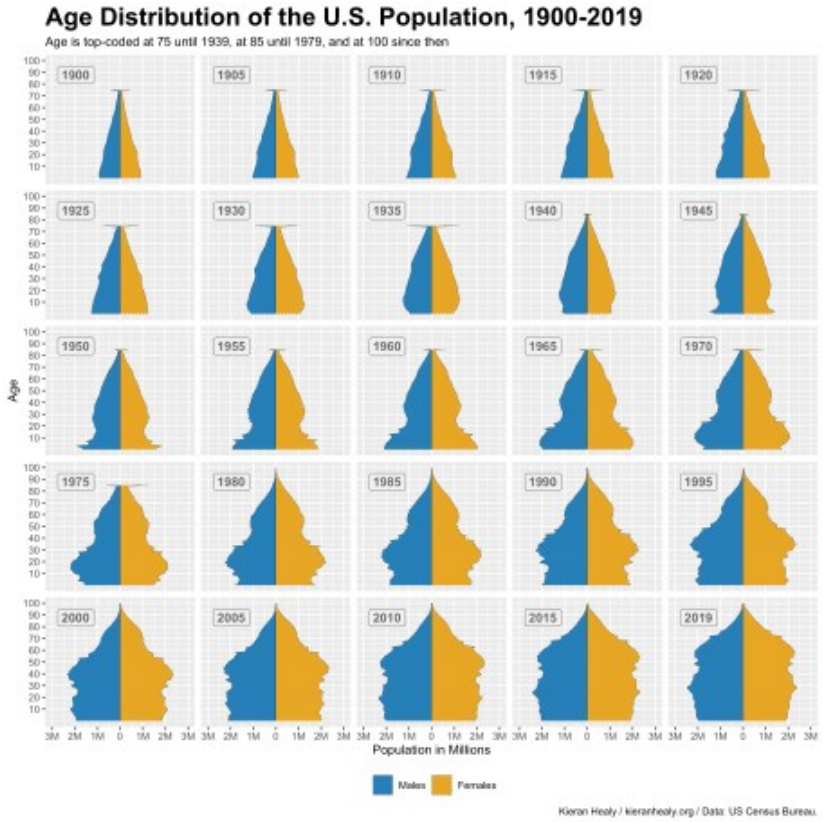
.



A small multiple plot of selected population pyramids

Instead of an animation, let's make the less-flashy but, frankly, in all likelihood more useful small multiple plot seen here. With the package installed we can produce it as follows:

```
 1
 2 library(tidyverse)
 3 library(uscenpops)
 4
 5 uscenpops
 6 #> # A tibble: 10,520 x 5
 7 #>     year   age     pop   male female
 8 #>
 9 #>  1  1900     0 1811000 919000 892000
10 #>  2  1900     1 1835000 928000 907000
11 #>  3  1900     2 1846000 932000 914000
12 #>  4  1900     3 1848000 932000 916000
13 #>  5  1900     4 1841000 928000 913000
14 #>  6  1900     5 1827000 921000 906000
15 #>  7  1900     6 1806000 911000 895000
16 #>  8  1900     7 1780000 899000 881000
17 #>  9  1900     8 1750000 884000 866000
18 #> 10  1900     9 1717000 868000 849000
19 #> # … with 10,510 more rows
20
```

That's what the dataset looks like. We'll lengthen it, calculate a relative frequency (that we won't use in this particular plot) and add a base value that we'll use for the ribbon boundaries below.

```

pop_pyr <- uscenpops %>% select(year, age, male, female) %>%
  pivot_longer(male:female, names_to = "group", values_to = "count") %>%
  group_by(year, group) %>%
  mutate(total = sum(count),
         pct = (count/total)*100,
         base = 0)

pop_pyr

#> # A tibble: 21,040 x 7
#> # Groups:   year, group [240]
#>     year   age group   count    total   pct  base
#>
#>  1  1900     0 male   919000 38867000  2.36     0
#>  2  1900     0 female 892000 37227000  2.40     0
#>  3  1900     1 male   928000 38867000  2.39     0
#>  4  1900     1 female 907000 37227000  2.44     0
#>  5  1900     2 male   932000 38867000  2.40     0
#>  6  1900     2 female 914000 37227000  2.46     0
#>  7  1900     3 male   932000 38867000  2.40     0
#>  8  1900     3 female 916000 37227000  2.46     0
#>  9  1900     4 male   928000 38867000  2.39     0
#> 10  1900     4 female 913000 37227000  2.45     0
#> # … with 21,030 more rows

```

Next we set up some little vectors of labels and colors, and then make a mini-dataframe of what we'll use as labels in the plot area, rather than using the default strip labels in `facet_wrap()`.

```

## Axis labels
mbreaks <- c("1M", "2M", "3M")

## colors
pop_colors <- c("#E69F00", "#0072B2")

## In-plot year labels
dat_text <- data.frame(
  label =  c(seq(1900, 2015, 5), 2019),
  year  =  c(seq(1900, 2015, 5), 2019),
  age = rep(95, 25),
  count = rep(-2.75e6, 25)
)

```

As before, the trick to making the pyramid is to set all the values for one category (here, males) to negative numbers.

```
pop_pyr$count[pop_pyr$group == "male"] <- -pop_pyr$count[pop_pyr$group == "male"]

p <- pop_pyr %>%
  filter(year %in% c(seq(1900, 2015, 5), 2019)) %>%
  ggplot(mapping = aes(x = age, ymin = base,
                       ymax = count, fill = group))

p + geom_ribbon(alpha = 0.9, color = "black", size = 0.1) +
  geom_label(data = dat_text,
             mapping = aes(x = age, y = count,
                           label = label), inherit.aes = FALSE,
             vjust = "inward", hjust = "inward",
             fontface = "bold",
             color = "gray40",
             fill = "gray95") +
  scale_y_continuous(labels = c(rev(mbreaks), "0", mbreaks),
                     breaks = seq(-3e6, 3e6, 1e6),
                     limits = c(-3e6, 3e6)) +
  scale_x_continuous(breaks = seq(10, 100, 10)) +
  scale_fill_manual(values = pop_colors, labels = c("Females", "Males")) +
  guides(fill = guide_legend(reverse = TRUE)) +
  labs(x = "Age", y = "Population in Millions",
       title = "Age Distribution of the U.S. Population, 1900-2019",
       subtitle = "Age is top-coded at 75 until 1939, at 85 until 1979, and at 100
since then",
       caption = "Kieran Healy / kieranhealy.org / Data: US Census Bureau.",
       fill = "") +
  theme(legend.position = "bottom",
        plot.title = element_text(size = rel(2), face = "bold"),
        strip.background = element_blank(),
        strip.text.x = element_blank()) +
  coord_flip() +
  facet_wrap(~ year, ncol = 5)
```

The calls to `geom_ribbon()` and `geom_label()` draw the actual plots, and everything else is just a little attention to detail in order to make it come out nicely.