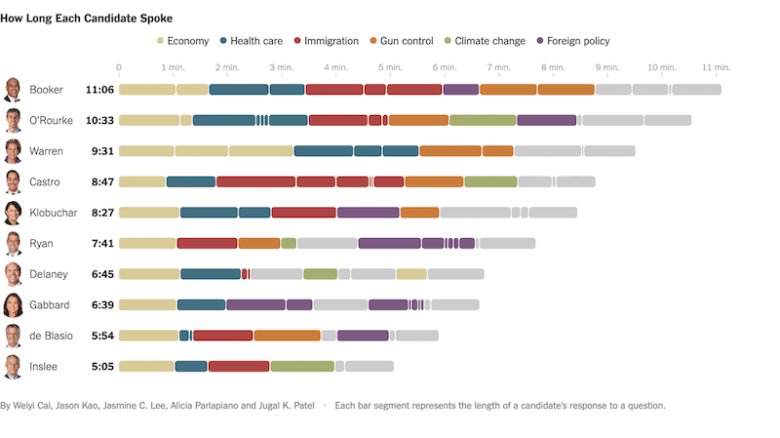
The first U.S. Democratic debates of the 2020 election season were held over two nights this past week due to the daft number of candidates running for POTUS. The spiffy [@NYTgraphics](http://twitter.com/NYTgraphics) folks took the tallies of time spent blathering by each speaker/topic and made rounded rectangle segmented bar charts ordered by the time the blathering was performed (these aren’t really debates, they’re loosely-prepared for performances) which I have dubbed “chicklet” charts due to a vague resemblance to the semi-popular gum/candy.

You can see each day’s live, javascript-created NYTimes charts here:

* <https://www.nytimes.com/interactive/2019/admin/100000006581096.embedded.html>
* <https://www.nytimes.com/interactive/2019/admin/100000006584572.embedded.html>

and this is a PNG snapshot of one of them:

[](https://i2.wp.com/rud.is/b/wp-content/uploads/2019/06/nytimes.png?ssl=1)

I liked the chicklet aesthetic enough to make a new {ggplot2} geom\_chicklet() to help folks make them. To save some blog bytes, you can read how to install the package over at <https://cinc.rud.is/web/packages/ggchicklet/>.

**Making Chicklet Charts**

Since the [@NYTimes](http://twitter.com/NYTimes) chose to use javascript to make their chart they also kinda made the data available (view the source of both of the aforelinked URLs) which I’ve wrangled a bit and put into the {ggchicklet} package. We’ll use it to progress from producing basic bars to crunching out chicklets and compare all the candidates across both days.

While the [@Nytimes](http://twitter.com/Nytimes) chart(s) provide a great deal of information, most media outlets focused on how much blather time each candidate managed to get. We do not need anything fancier than a bar chart or table to do that:

library(hrbrthemes)

library(ggchicklet)

library(tidyverse)

data("debates2019")

count(debates2019, speaker, wt=elapsed, sort=TRUE) %>%

mutate(speaker = fct\_reorder(speaker, n, sum, .desc=FALSE)) %>%

mutate(speaker = fct\_inorder(speaker) %>% fct\_rev()) %>%

ggplot(aes(speaker,n)) +

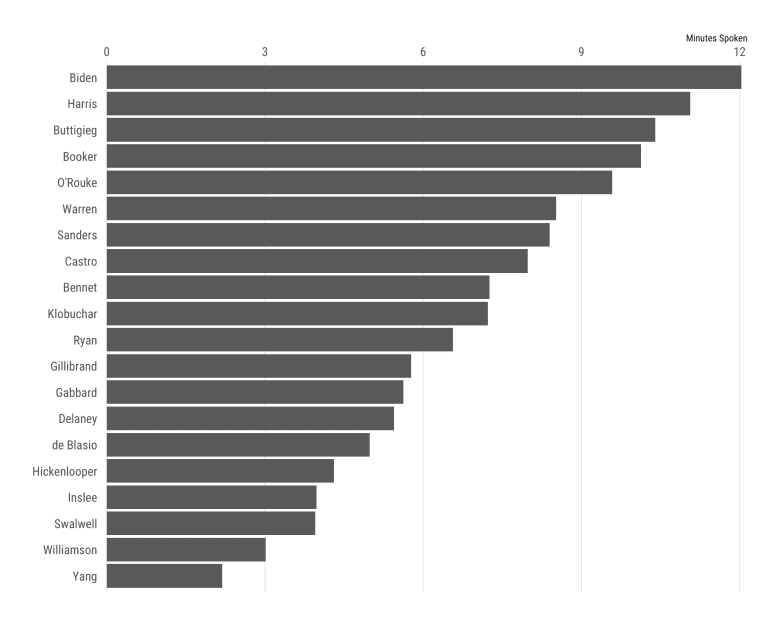
geom\_col() +

scale\_y\_comma(position = "right") +

coord\_flip() +

labs(x = NULL, y = "Minutes Spoken") +

theme\_ipsum\_rc(grid="X")

[](https://i0.wp.com/rud.is/b/wp-content/uploads/2019/06/chicklet-01-1.png?ssl=1)

If we want to see the same basic view but include how much time each speaker spent on each topic, we can also do that without much effort:

count(debates2019, speaker, topic, wt=elapsed, sort=TRUE) %>%

mutate(speaker = fct\_reorder(speaker, n, sum, .desc=FALSE)) %>%

ggplot(aes(speaker, n , fill = topic)) +

geom\_col() +

scale\_y\_comma(position = "right") +

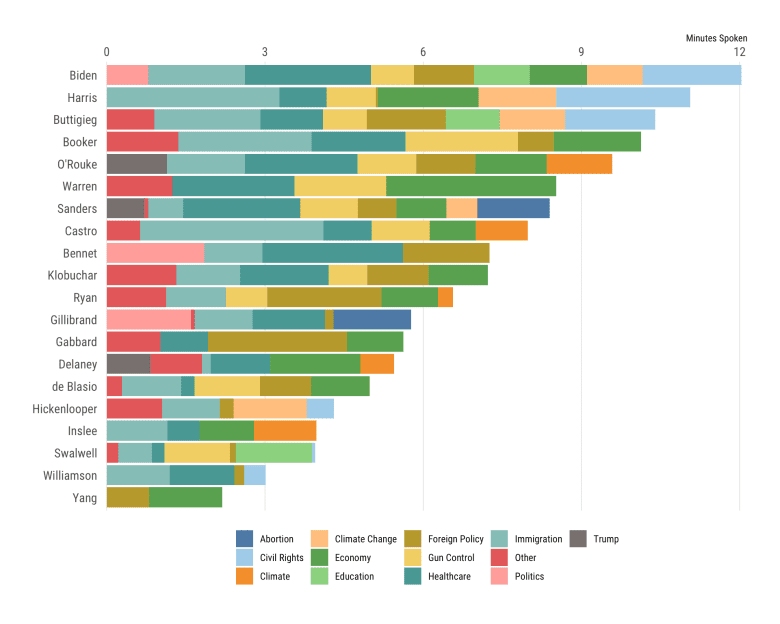
ggthemes::scale\_fill\_tableau("Tableau 20", name = NULL) +

coord\_flip() +

labs(x = NULL, y = "Minutes Spoken") +

theme\_ipsum\_rc(grid="X") +

theme(legend.position = "bottom")

[](https://i2.wp.com/rud.is/b/wp-content/uploads/2019/06/chicklet-02-1.png?ssl=1)

By default geom\_col() is going to use the fill aesthetic to group the bars and use the default sort order to stack them together.

We can also get a broken out view by not doing the count() and just letting the segments group together and use a white bar outline to keep them distinct:

debates2019 %>%

mutate(speaker = fct\_reorder(speaker, elapsed, sum, .desc=FALSE)) %>%

ggplot(aes(speaker, elapsed, fill = topic)) +

geom\_col(color = "white") +

scale\_y\_comma(position = "right") +

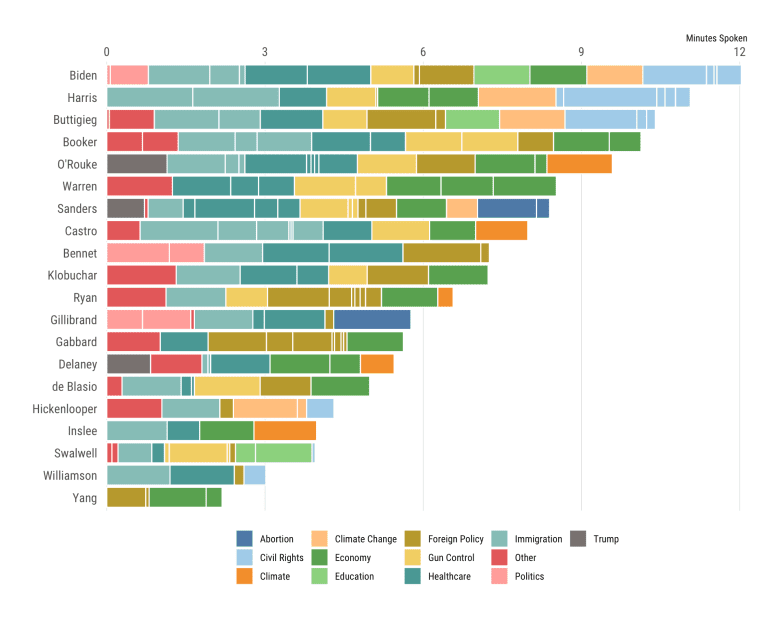
ggthemes::scale\_fill\_tableau("Tableau 20", name = NULL) +

coord\_flip() +

labs(x = NULL, y = "Minutes Spoken") +

theme\_ipsum\_rc(grid="X") +

theme(legend.position = "bottom")

[](https://i1.wp.com/rud.is/b/wp-content/uploads/2019/06/chicklet-03-1.png?ssl=1)

While I liked the rounded rectangle aesthetic, I also really liked how the [@nytimes](http://twitter.com/nytimes) ordered the segments by when the topics occurred during the debate. For other types of chicklet charts you don’t need to grouping variable to be a time-y whime-y column, just try to use something that has a sane ordering characteristic to it:

debates2019 %>%

mutate(speaker = fct\_reorder(speaker, elapsed, sum, .desc=FALSE)) %>%

ggplot(aes(speaker, elapsed, group = timestamp, fill = topic)) +

geom\_col(color = "white", position = position\_stack(reverse = TRUE)) +

scale\_y\_comma(position = "right") +

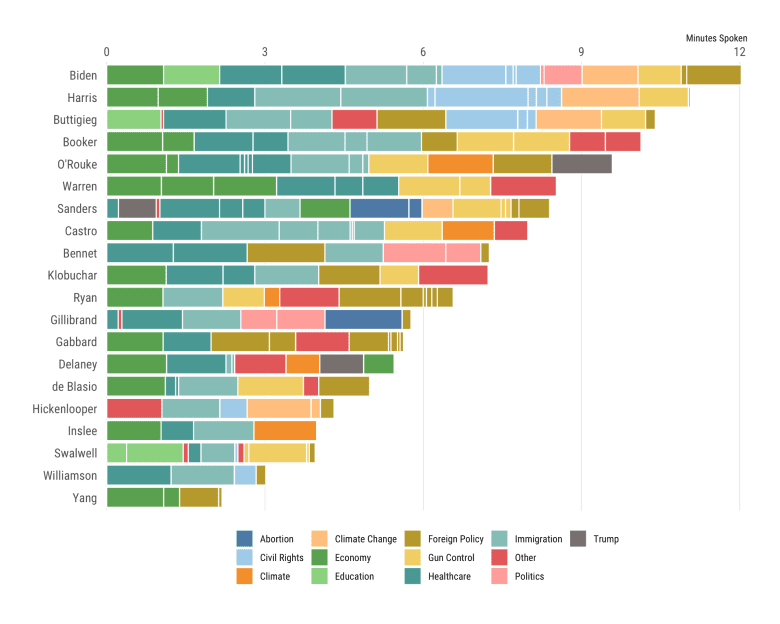
ggthemes::scale\_fill\_tableau("Tableau 20", name = NULL) +

coord\_flip() +

labs(x = NULL, y = "Minutes Spoken") +

theme\_ipsum\_rc(grid="X") +

theme(legend.position = "bottom")

[](https://i2.wp.com/rud.is/b/wp-content/uploads/2019/06/chicklet-04-1.png?ssl=1)

That last chart is about as far as you could go to reproduce the [@nytimes](http://twitter.com/nytimes) look-and-feel without jumping through some serious gg-hoops.

I had made a rounded rectangle hidden geom to make rounded-corder tiles for the [{statebins} package](https://git.rud.is/hrbrmstr/statebins) so making a version of ggplot2::geom\_col() (which I also added to {ggplot2}) was pretty straightforward. There are some key differences in the defaults of geom\_chicklet():

* a “white” stroke for the chicklet/segment (geom\_col() has NA for the stroke)
* automatic reversing of the group order (geom\_col() uses the standard sort order)
* radius setting of unit(3, "px") (change this as you need)
* chicklet legend geom (b/c they aren’t bars or points)

You likely just want to see it in action, so here it is without further adieu:

debates2019 %>%

mutate(speaker = fct\_reorder(speaker, elapsed, sum, .desc=FALSE)) %>%

ggplot(aes(speaker, elapsed, group = timestamp, fill = topic)) +

geom\_chicklet(width = 0.75) +

scale\_y\_continuous(

expand = c(0, 0.0625),

position = "right",

breaks = seq(0, 14, 2),

labels = c(0, sprintf("%d min.", seq(2, 14, 2)))

) +

ggthemes::scale\_fill\_tableau("Tableau 20", name = NULL) +

coord\_flip() +

labs(

x = NULL, y = NULL, fill = NULL,

title = "How Long Each Candidate Spoke",

subtitle = "Nights 1 & 2 of the June 2019 Democratic Debates",

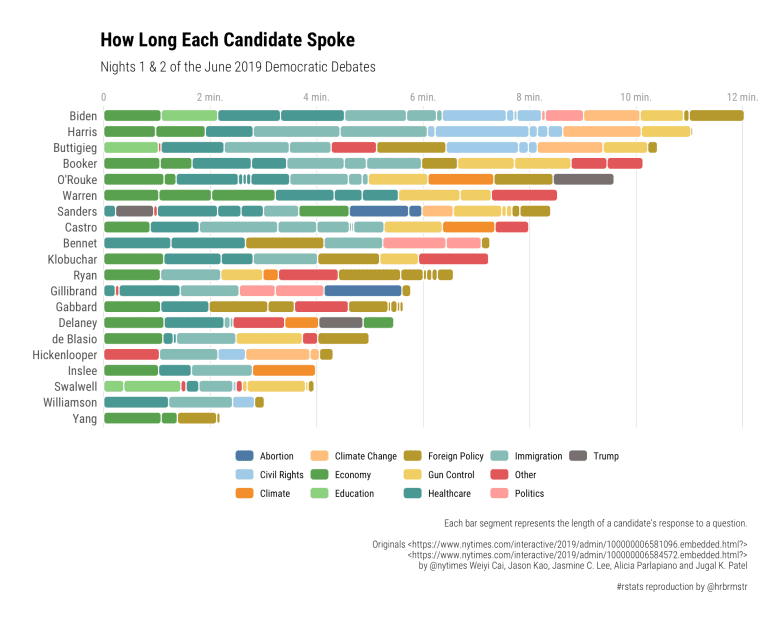
caption = "Each bar segment represents the length of a candidate’s response to a question.\n\nOriginals \n\nby [@nytimes](http://twitter.com/nytimes) Weiyi Cai, Jason Kao, Jasmine C. Lee, Alicia Parlapiano and Jugal K. Patel\n\n#rstats reproduction by [@hrbrmstr](http://twitter.com/hrbrmstr)"

) +

theme\_ipsum\_rc(grid="X") +

theme(axis.text.x = element\_text(color = "gray60", size = 10)) +

theme(legend.position = "bottom")

[](https://i2.wp.com/rud.is/b/wp-content/uploads/2019/06/chicklet-06-1.png?ssl=1)

Yes, I upped the ggplot2 tweaking a bit to get closer to the [@nytimes](http://twitter.com/nytimes) (FWIW I like the Y gridlines, YMMV) but didn’t have to do much else to replace geom\_col() with geom\_chicket(). You’ll need to play with the segment width value depending on the size of your own, different plots to get the best look (just like you do with any other geom).

Astute, intrepid readers will note that the above chart has all the topics whereas the [@nytimes](http://twitter.com/nytimes) just has a few. We can do the grouping of non-salient topics into an “Other” category with forcats::fct\_other() and make a manual fill scale from the values ~~stolen from~~used in homage from the [@nytimes](http://twitter.com/nytimes):

debates2019 %>%

mutate(speaker = fct\_reorder(speaker, elapsed, sum, .desc=FALSE)) %>%

mutate(topic = fct\_other(

topic,

c("Immigration", "Economy", "Climate Change", "Gun Control", "Healthcare", "Foreign Policy"))

) %>%

ggplot(aes(speaker, elapsed, group = timestamp, fill = topic)) +

geom\_chicklet(width = 0.75) +

scale\_y\_continuous(

expand = c(0, 0.0625),

position = "right",

breaks = seq(0, 14, 2),

labels = c(0, sprintf("%d min.", seq(2, 14, 2)))

) +

scale\_fill\_manual(

name = NULL,

values = c(

"Immigration" = "#ae4544",

"Economy" = "#d8cb98",

"Climate Change" = "#a4ad6f",

"Gun Control" = "#cc7c3a",

"Healthcare" = "#436f82",

"Foreign Policy" = "#7c5981",

"Other" = "#cccccc"

),

breaks = setdiff(unique(debates2019$topic), "Other")

) +

guides(

fill = guide\_legend(nrow = 1)

) +

coord\_flip() +

labs(

x = NULL, y = NULL, fill = NULL,

title = "How Long Each Candidate Spoke",

subtitle = "Nights 1 & 2 of the June 2019 Democratic Debates",

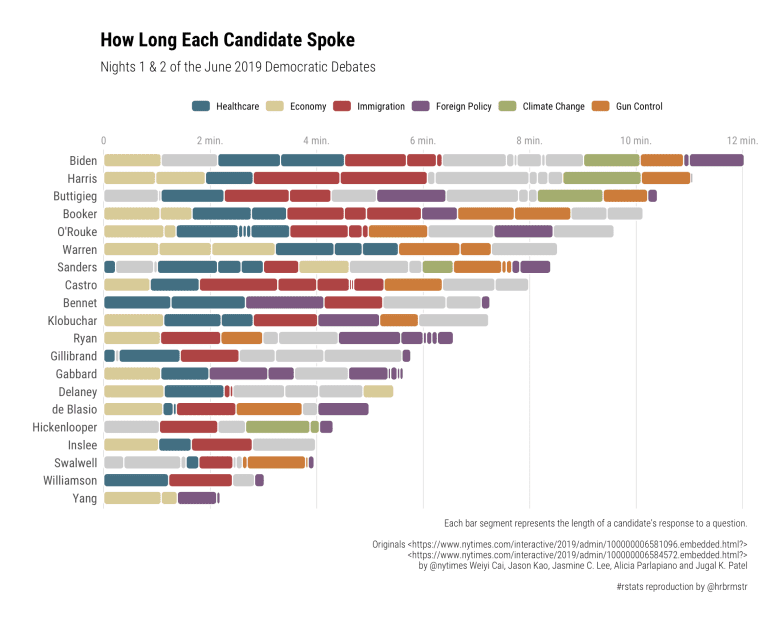
caption = "Each bar segment represents the length of a candidate’s response to a question.\n\nOriginals \n\nby [@nytimes](http://twitter.com/nytimes) Weiyi Cai, Jason Kao, Jasmine C. Lee, Alicia Parlapiano and Jugal K. Patel\n\n#rstats reproduction by [@hrbrmstr](http://twitter.com/hrbrmstr)"

) +

theme\_ipsum\_rc(grid="X") +

theme(axis.text.x = element\_text(color = "gray60", size = 10)) +

theme(legend.position = "top")

[](https://i1.wp.com/rud.is/b/wp-content/uploads/2019/06/chicklet-07-1.png?ssl=1)

**FIN**

Remember, you can find out how to install {ggchicklet} and also where you can file issues or PRs over at <https://cinc.rud.is/web/packages/ggchicklet/>. The package has full documentation, including a vignette, but if any usage help is lacking, definitely file an issue.

If you use the package, don’t hesitate to share your creations in a comment or on Twitter so other folks can see how to use the package in different contexts.