Topic 6. Visualization Conditional Variation - Part II

Josh Clinton

9/27/2021

Plotting Multiple Variables Over Time (Time-Series)

- Can we plot support for Biden and support for Trump separately over time (on the same plot)?

```
load(file="data/Pres2020.PV.Rdata")
election.day <- as.Date("11/3/2020", "%m/%d/%Y")

Pres2020.PV <- Pres2020.PV %>%
                mutate(EndDate = as.Date(Pres2020.PV$EndDate, "%
                      StartDate = as.Date(Pres2020.PV$StartDate,
                      DaysToED = as.numeric(election.day - EndDa
                      margin = Biden - Trump)
```

“Stretch” Extensions

- Comparing the change in `margin` over time for multiple election years?
- Comparing the support for candidates (`Biden` and `Trump`) in multiple states?
- Comparing the support for candidates according to different types of polls?
- Comparing the support for presidential candidates relative to senatorial and gubernatorial candidates in the same state?
- Comparing the deaths/cases per capita over time (and also by county/state)?
- Comparing the performance of an NBA team/player in several dimensions over time?

First, define the canvas!

```
BidenTrumpplot <- ggplot(Pres2020.PV)  +
  labs(title="% Biden and Trump in 2020 National Popular Vote Po
  labs(y = "Pct. Support") +
  labs(x = "Poll Ending Date")
```

Blank scale!

```
BidenTrumpplot
```

% Biden and Trump in 2020 National Popular Vote Polls Over Time
Pct. Support
Poll Ending Date

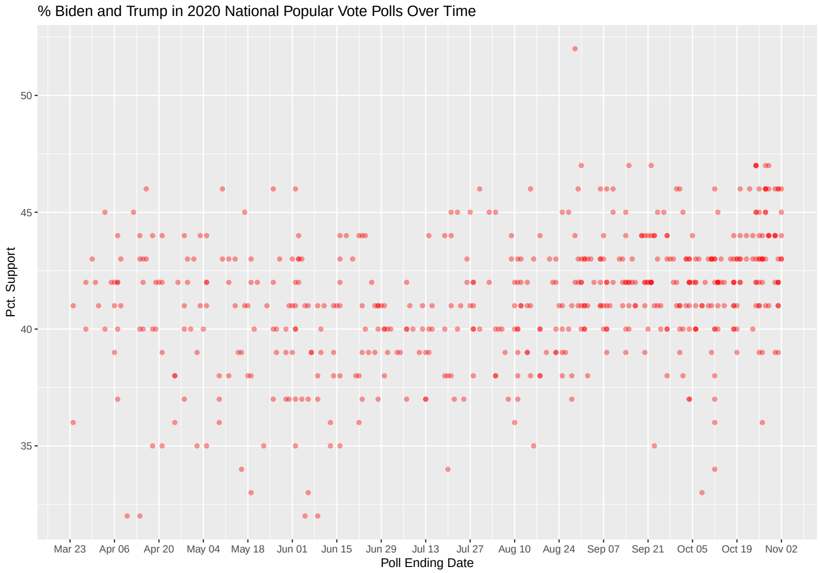
Now, add the points for Trump

```r
BidenTrumpplot <- BidenTrumpplot +
  geom_point(aes(x = EndDate, y = Trump),
             color = "red", alpha=.4)  +
  scale_x_date(date_breaks = "2 week", date_labels = "%b %d")
```

- Note the use of aes in geom_point()!

What do you have?

```
BidenTrumpplot
```



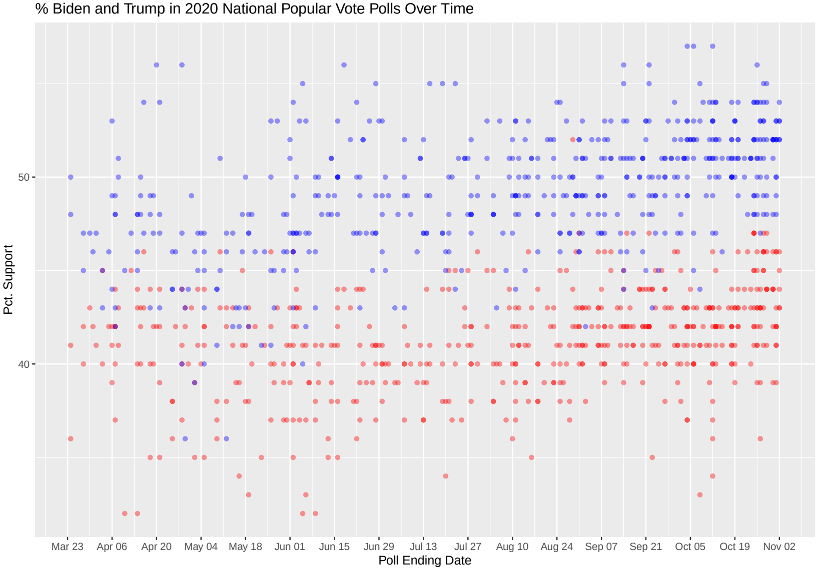
Now, add the points for Biden

```
BidenTrumpplot <- BidenTrumpplot +
  geom_point(aes(x = EndDate, y = Biden),
             color = "blue", alpha=.4)
```

- `ggplot` will now rescale y-axis to fit both `Trump` and `Biden`

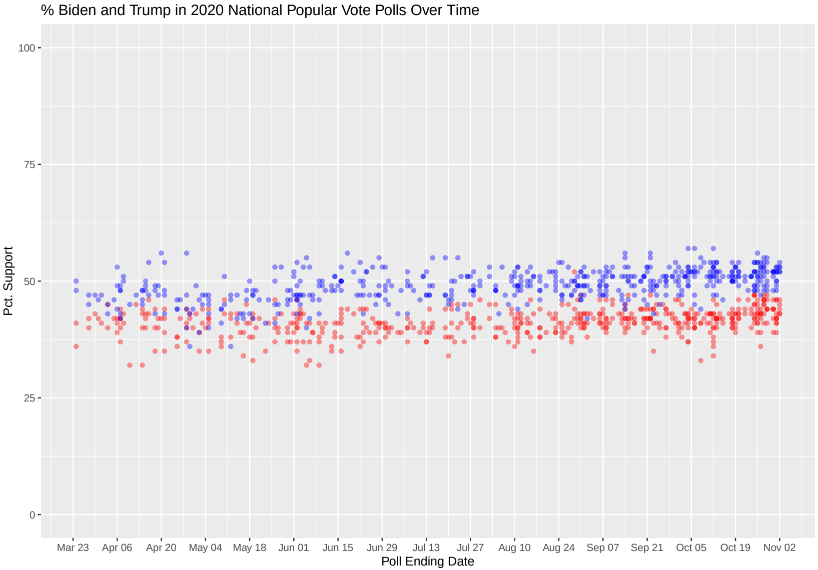
Adding Biden

```
BidenTrumpplot
```



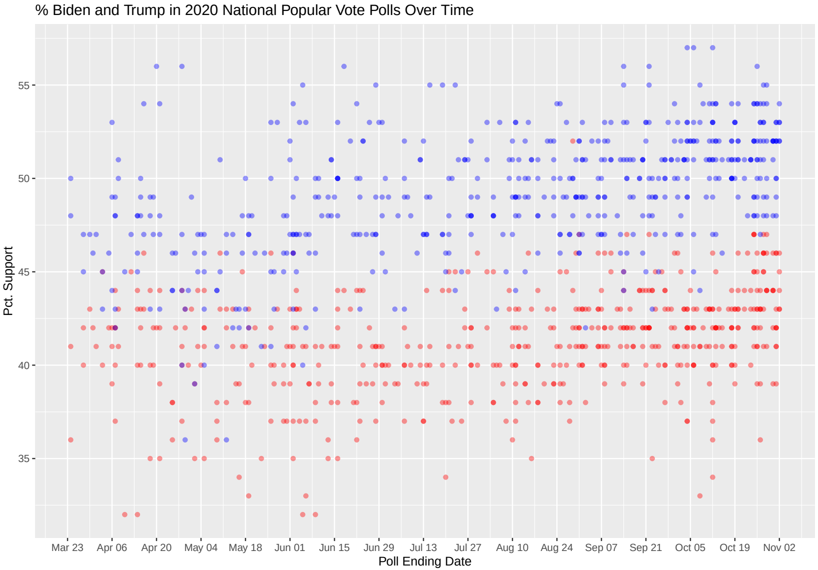
Set the Axis?

```
BidenTrumpplot + ylim(0,100)
```



For reals

```
BidenTrumpplot + scale_y_continuous(breaks=seq(30,70,by=5))
```



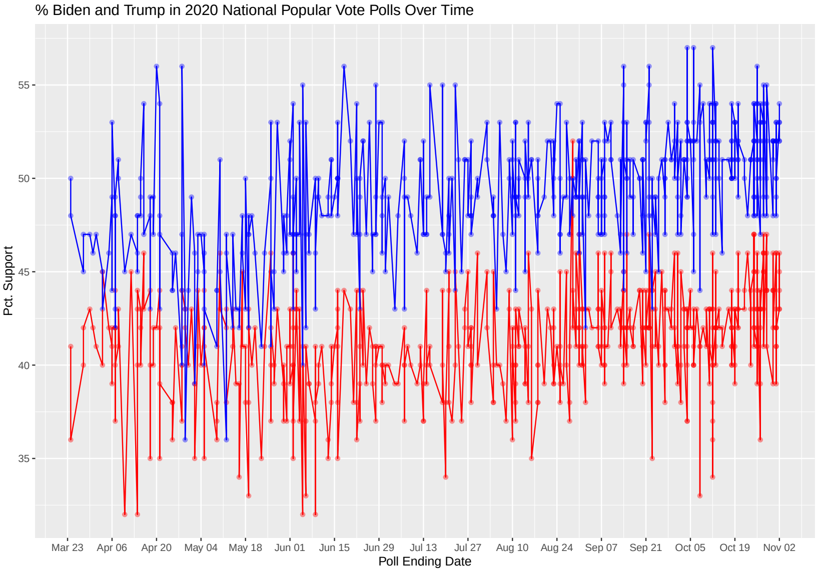
Adding some lines?

```r
BTwithlines <- BidenTrumpplot +
  scale_y_continuous(breaks=seq(30,70,by=5)) +
  geom_line(aes(x = EndDate, y = Trump), color = "red") +
  geom_line(aes(x = EndDate, y = Biden), color = "blue")
```

- We add lines the same way we added points!

But we shouldn't...

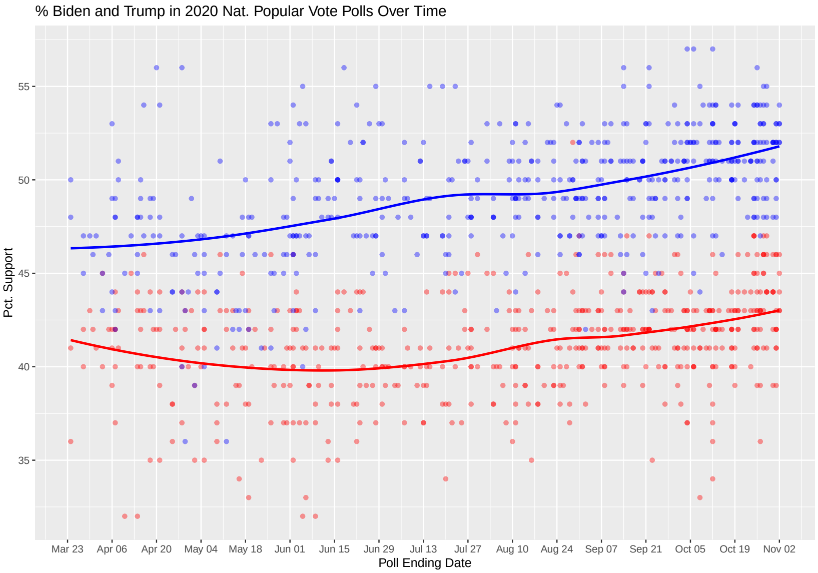
```
BTwithlines
```



Putting it all together

```
BTNational <- ggplot(Pres2020.PV) +
  geom_point(aes(x = EndDate, y = Trump),
             color = "red", alpha = .4) +
  geom_point(aes(x = EndDate, y = Biden),
             color = "blue", , alpha = .4)  +
  geom_smooth(aes(x = EndDate, y = Trump),
              color = "red",se=F) +
  geom_smooth(aes(x = EndDate, y = Biden),
              color = "blue",se=F) +
  labs(title="% Biden and Trump in 2020 Nat. Popular Vote Polls
  labs(y = "Pct. Support") +
  labs(x = "Poll Ending Date") +
  scale_x_date(date_breaks = "2 week", date_labels = "%b %d") +
  scale_y_continuous(breaks=seq(30,70,by=5))
```

BTNational



% Biden and Trump in 2020 Nat. Popular Vote Polls Over Time
Pct. Support
55
50
45
40
35
Mar 23
Apr 06
Apr 20
May 04
May 18
Jun 01
Jun 15
Jun 29
Jul 13
Jul 27
Aug 10
Aug 24
Sep 07
Sep 21
Oct 05
Oct 19
Nov 02
Poll Ending Date

“Smoothing”

- How can we understand the role of “smoothing”?
- Very important because that is the trend we focus on!
- Cannot rely on a default we do not understand.
- We want to summarize the trend by looking at the polling average for all polls conducted in a certain time period (bandwidth).
- Smoothing helps ensure that an outlier does not distort our visualization (and our interpretation of central tendency of the data)
- Smoothing is taking a mean *conditional* on another set of values – here time! – over the range of those values.

ASIDE: Often Very Confusing to People

Suppose you are interested in characterizing the relationship between vaccinations and hospitalizations.

Two possibilities:

1. Conditional on being hospitalized, how many were vaccinated? ($Pr(Vaccinated|Hospitalization)$)
2. Conditional on being vaccinated, how many were hospitalized? ($Pr(Hospitalization|Vaccinated)$)

- How measure 1? How measure 2?

ASIDE: Often Very Confusing to People

Suppose you are interested in characterizing the relationship between vaccinations and hospitalizations.

Two possibilities:

1. Conditional on being hospitalized, how many were vaccinated? ($Pr(Vaccinated|Hospitalization)$)
2. Conditional on being vaccinated, how many were hospitalized? ($Pr(Hospitalization|Vaccinated)$)

- How measure 1? How measure 2?
- What do you care about?
- Does the meaning change over time?

Conditioning Variable: Time

Start by defining a variable `all_dates` – all possible dates of interest (not just those with data!)

```
all_dates <- seq(min(Pres2020.PV$EndDate), election.day,
                 by = "days")
```

GOAL

- For each possible date, what is the average support for Biden and Trump among the polls taken during the `X` days prior?
- Requires define a moving “bandwidth” of dates and calculating average support among polls during that time.
- To do this we are going to “loop” over dates.
- ASIDE: often inefficient; vectorize your computations if possible!

Looping

GOAL: Calculate the average support for Trump in the last 3 days of the election.

- I could `filter`, `group_by`, and `summarize` but I could also:

```
mean(Pres2020.PV$Trump[Pres2020.PV$EndDate == "2020-10-31"])
mean(Pres2020.PV$Trump[Pres2020.PV$EndDate == "2020-11-01"])
mean(Pres2020.PV$Trump[Pres2020.PV$EndDate == "2020-11-02"])
```

- Could do this for every day, but: very inefficient & prone to error (copy and paste bad!)!

Looping

```r
dates <- c("2020-10-31","2020-11-01","2020-11-02")
```

Format of a loop is:

```r
for(i in dates){
  CODE TO REPEAT HERE
}
```

What does this do?

```r
for(i in dates){
  print(i)
}
```

What does this do?

```r
for(i in dates){
  print(i)
}
```

```r
for(i in dates){
  print(i)
}
```

```
## [1] "2020-10-31"
## [1] "2020-11-01"
## [1] "2020-11-02"
```

What does this do?

```r
for(i in dates){
  print(i)
  mean(Pres2020.PV$Trump[Pres2020.PV$EndDate == i])
}
```

- Think: what is the value of `i`? How is it changing?

What does this do?

```r
for(i in dates){
  print(i)
  mean(Pres2020.PV$Trump[Pres2020.PV$EndDate == i])
}
```

```
## [1] "2020-10-31"
## [1] "2020-11-01"
## [1] "2020-11-02"
```

What does this do?

```r
for(i in dates){
  print(i)
  mean(Pres2020.PV$Trump[Pres2020.PV$EndDate == i])
}
```

```
## [1] "2020-10-31"
## [1] "2020-11-01"
## [1] "2020-11-02"
```

- Useful for debugging!

What does this do?

```r
for(i in dates){
  print(i)
  mean(Pres2020.PV$Trump[Pres2020.PV$EndDate == i])
}
```

- Think: what is the value of `i`? How is it changing?
- Think: what are we asking R to do? What is it doing with what it is doing?

What does this do?

```r
PollAvg <- NULL

for(i in dates){
  print(i)
  PollAvg[i] <- mean(Pres2020.PV$Trump[Pres2020.PV$EndDate == i]
}
```

```
## [1] "2020-10-31"
## [1] "2020-11-01"
## [1] "2020-11-02"
```

What does this do?

```
PollAvg <- NULL

for(i in dates){
  print(i)
  PollAvg[i] <- mean(Pres2020.PV$Trump[Pres2020.PV$EndDate == i]
}
```

```
## [1] "2020-10-31"
## [1] "2020-11-01"
## [1] "2020-11-02"
```

```
PollAvg
```

```
## 2020-10-31 2020-11-01 2020-11-02
##   43.16667   42.44444   44.20000
```

Preliminaries for the Loop

```r
Bandwidth <- 2
PV_avg <- vector(length(all_dates), mode = "list")
```

Looping

```r
for (i in seq_along(all_dates)) {
  date <- all_dates[i]

    PV_avg[[i]] <- Pres2020.PV %>%
    filter(as.integer(EndDate - date) <= 0 &
           as.integer(EndDate - date) > - Bandwidth) %>%
    summarize(Biden = mean(Biden),
              Trump = mean(Trump)) %>%
      mutate(date = date)
}
```

Building a dataframe from a list

```
class(PV_avg)
```

```
## [1] "list"
```

```
dim(PV_avg)
```

```
## NULL
```

```
PV_avg[1]
```

```
## [[1]]
## # A tibble: 1 x 3
##   Biden Trump date
##   <dbl> <dbl> <date>
## 1    49  38.5 2020-03-24
```

```
PV_avg[[1]]
```

```
## # A tibble: 1 x 3
##   Biden Trump date
##   <dbl> <dbl> <date>
## 1    49  38.5 2020-03-24
```

Building a dataframe from a list

```
pop_vote_avg <- bind_rows(PV_avg)
class(pop_vote_avg)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
dim(pop_vote_avg)
```

```
## [1] 225   3
```

```
head(pop_vote_avg)
```

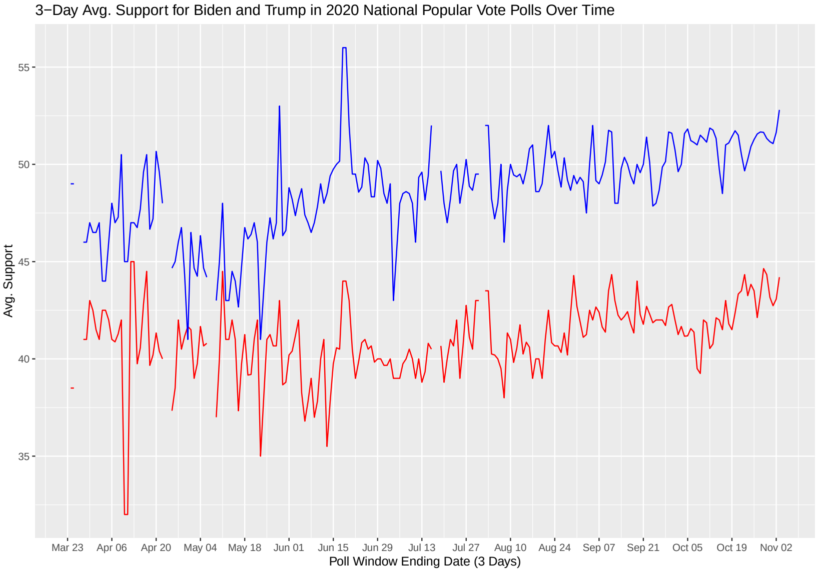
```
## # A tibble: 6 x 3
##   Biden Trump date
##   <dbl> <dbl> <date>
## 1    49  38.5 2020-03-24
## 2    49  38.5 2020-03-25
## 3   NaN NaN   2020-03-26
## 4   NaN NaN   2020-03-27
## 5    46  41   2020-03-28
## 6    46  41   2020-03-29
```

Ready to plot?

```
PlotTS <- ggplot(pop_vote_avg) +
  geom_line(aes(x = date, y = Trump), color = "red") +
  geom_line(aes(x = date, y = Biden), color = "blue")  +
  labs(title="3-Day Avg. Support for Biden and Trump in 2020 Nat
  labs(y = "Avg. Support") +
  labs(x = "Poll Window Ending Date (3 Days)") +
  scale_x_date(date_breaks = "2 week", date_labels = "%b %d") +
  scale_y_continuous(breaks=seq(30,70,by=5))
```

Ready to plot?

```
PlotTS
```



Weeklong bandwidth?

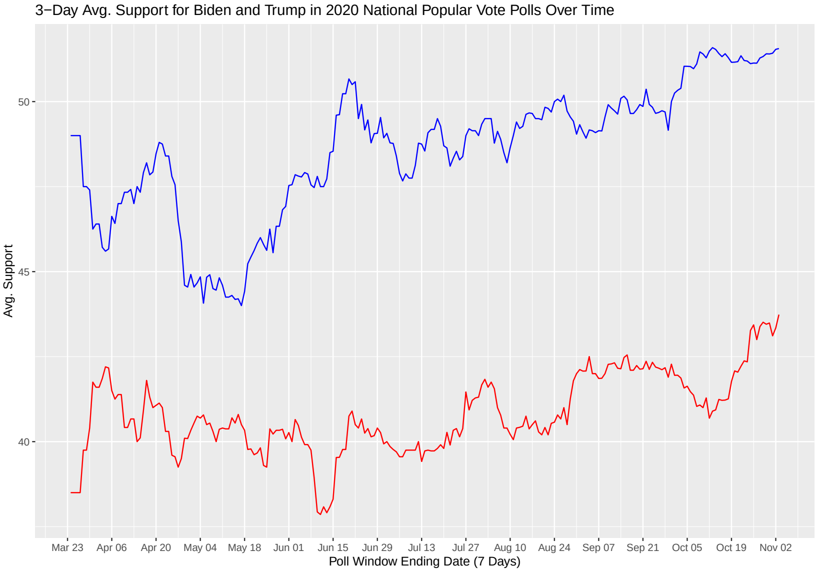
```r
Bandwidth <- 7
PV_avg <- vector(length(all_dates), mode = "list") # holding var

for (i in seq_along(all_dates)) {
  date <- all_dates[i]

  PV_avg[[i]] <- Pres2020.PV %>%
    filter(as.integer(EndDate - date) <= 0,
           as.integer(EndDate - date) > - Bandwidth) %>%
    summarize(Biden = mean(Biden),
              Trump = mean(Trump)) %>%
    mutate(date = date)
}

pop_vote_avg7 <- bind_rows(PV_avg)
```

Weeklong bandwidth?

```
PlotTS7 <- ggplot(pop_vote_avg7) +
  geom_line(aes(x = date, y = Trump), color = "red") +
  geom_line(aes(x = date, y = Biden), color = "blue")  +
  labs(title="3-Day Avg. Support for Biden and Trump in 2020 Nat
  labs(y = "Avg. Support") +
  labs(x = "Poll Window Ending Date (7 Days)") +
  scale_x_date(date_breaks = "2 week", date_labels = "%b %d") +
  scale_y_continuous(breaks=seq(30,70,by=5))
```

Weeklong bandwidth?

```
PlotTS7
```

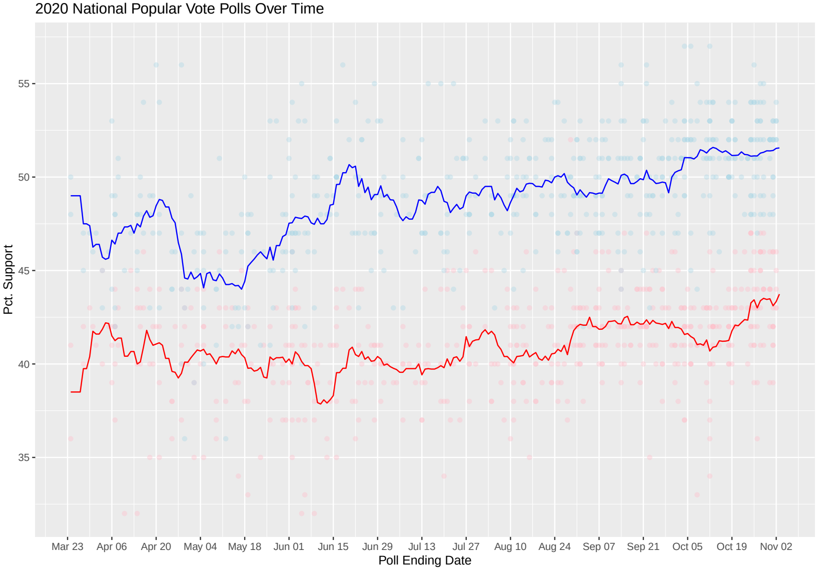


Add Points!

```
# Now overlay on points!
PopVotePlot <- ggplot() +
  geom_point(data=Pres2020.PV,aes(x = EndDate, y = Trump),
             color = "pink", alpha=.4) +
  geom_point(data=Pres2020.PV,aes(x = EndDate, y = Biden),
             color = "light blue", alpha=.4) +
  geom_line(data=pop_vote_avg7, aes(x = date, y = Trump),
            color = "red") +
  geom_line(data=pop_vote_avg7, aes(x = date, y = Biden),
            color = "blue") +
  labs(title="2020 National Popular Vote Polls Over Time") +
  labs(y = "Pct. Support") +
  labs(x = "Poll Ending Date") +
  scale_x_date(date_breaks = "2 week", date_labels = "%b %d") +
  scale_y_continuous(breaks=seq(30,70,by=5))
```

The final product?

```
PopVotePlot
```



Going Forward: On your own?

- What is the right bandwidth? How much change is “real”?
- Every poll counted equally (`SampleSize`)?
- Every type of poll counted equally? (`filter` by different types of polls?)
- Is this really what we care about?
- Other data? (Past Elections? Current Elections? Non-Elections?)