investigation-US-election-2016

```
library(conflicted)
    conflicts_prefer(dplyr::filter, dplyr::lag)
library(tidyverse)
library(R6)
library(formatR)
library(lubridate)
library(rmarkdown)
```

```
allstates = polls$state |>
    unique() |>
    sort()
polls = polls |>
    mutate(
        interval = startdate %--% enddate,
        middate = ymd(startdate + (enddate - startdate)/2),
        .keep = "unused",
        .after = "state"
    ) |>
    mutate(
        grade = grade |>
            factor(
                levels = c(
                    "A+",
                    "A",
                    "A-",
                    "B+",
                    "B",
                    "B-",
                    "C+",
                    "C",
                    "C-",
                    "D+",
                    "D",
                    "D-",
                    NA
                )
            ),
        .keep = "unused",
        .before = "samplesize"
    )
```

```
finalresults = tibble(
    state = allstates,
    clinton = c(34.36, 36.55, 45.13, 33.65, 61.73, 48.16, 54.57, 53.09, 90.86, 47.82, 45.64, 62.22, 27.4
    trump = c(62.08, 51.28, 48.67, 60.57, 31.62, 43.25, 40.93, 41.71, 4.09, 49.02, 50.77, 30.04, 59.26,
```

```
    johnson = c(2.09, 5.88, 4.13, 2.65, 3.37, 5.18, 2.96, 3.33, 1.58, 2.20, 3.05, 3.72, 4.10, 3.79, 4.9
    mcmullin = c(NA, NA, 0.68, 1.17, 0.28, 1.04, 0.13, 0.16, NA, NA, 0.32, NA, 6.73, 0.21, NA, 0.79, 0.5
)
```

```
# helps filter for candidate name in results
candidatenames = c("clinton", "trump", "johnson", "mcmullin")

candidate = R6Class(
    classname = "candidate",
    public = list(
        name = "character",
        opponents = "character",
        colour = "character",
        polls = "tbl_df",
        finalresults = "tbl_df",

        initialize = function(name, colour){
            self$name = name
            self$colour = colour
            self$opponents = candidatenames[candidatenames != self$name] # the list of other candidates
            self$polls = polls |>
                select(!contains(self$opponents)) |> # filter polls to only this candidate
                rename(
                    rawpolls = starts_with("rawpoll"),
                    adjpolls = starts_with("adjpoll")
                ) # rename columns for use in candidateplot functions
            self$finalresults = finalresults |>
                select(!contains(self$opponents))
        } # end of initialize
    ) # end of list
)
```

```
clinton = candidate$new(name = "clinton", colour = "blue")
trump = candidate$new(name = "trump", colour = "red")
johnson = candidate$new(name = "johnson", colour = "orange")
mcmullin = candidate$new(name = "mcmullin", colour = "purple")

candidates = list(clinton, trump, johnson, mcmullin)
rm(candidatenames) # only needed for candidate construction
```

```
densepollsbegin = function(thestate, thedate) { # for graphing purposes
    # calculate the earliest date after thedate (inclusive) where there are three
    # consecutive polls in short succession (within two months)
    statepolls = polls |>
        filter(state == thestate, middate >= thedate) |>
        arrange(middate)
    earlydate = statepolls$middate |>
        min()
    while (nrow(statepolls) > 2) {
        statepolls = statepolls[-1,]
        mos = interval(
            start = earlydate,
            end = earlydate %m+% months(2, abbreviate = FALSE)
```

```
        )
        if (
            mos |>
            int_overlaps(
                statepolls$interval[which.min(sapply(statepolls$interval, int_start))]
            )
        ) {
            temppolls = statepolls[-1,]
            if (
                mos |>
                int_overlaps(
                    temppolls$interval[which.min(sapply(temppolls$interval, int_start))]
                )
            ) {
                return(earlydate)
            }
        }
        earlydate = statepolls$middate |>
            min()
    }
    return(earliestdate) # failed to find three such polls; just graph all polls
}

earliestdate = ymd("2016-08-01")
finaldate = ymd("2016-11-08")
dayafter = finaldate + days(1) # used just for technical graphing purposes
```

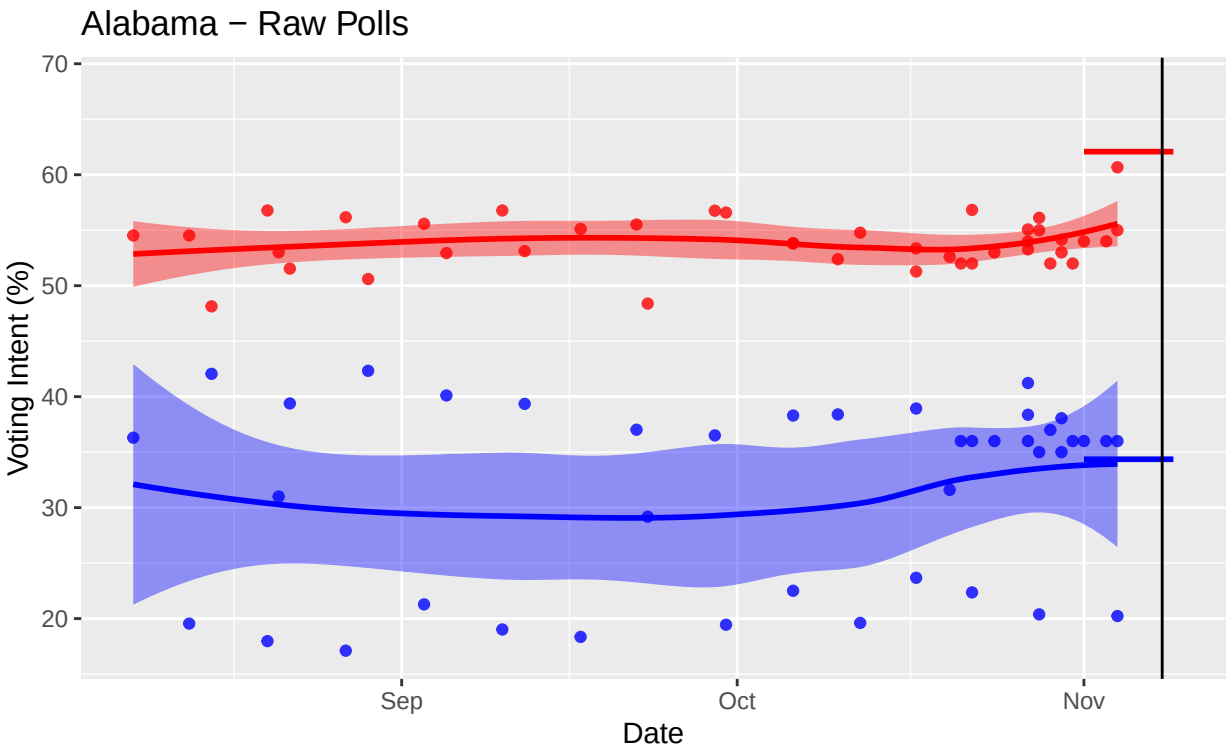
```
candidateplotraw = function(thestate, thecandidate, thefirstdate) {
    statepolls = thecandidate$polls |>
        filter(state == thestate)

    # methods to avoid plotting Johnson's numbers where his polls are insignificant
    # check whether Johnson obtained more than 13% raw in any poll conducted after Sep. 1st
    if (thecandidate$name == "johnson" && !(thestate %in% congdistricts)) {
        statepolls = statepolls |>
            filter(middate >= ymd("2016-09-01"))
        if (max(statepolls$rawpolls, na.rm = TRUE)<=13.0) {
            return(last_plot())
        }
    }

    if ( # in this state, not all polls for this candidate are NA
        any(!is.na(statepolls$rawpolls))
    ) {
        datevsraw = aes(
            x = statepolls$middate,
            y = statepolls$rawpolls
        )
        voteresult = filter(thecandidate$finalresults, state == thestate)[1,2] |>
            as.numeric()
        return(
            last_plot() + geom_point(
                mapping = datevsraw,
```

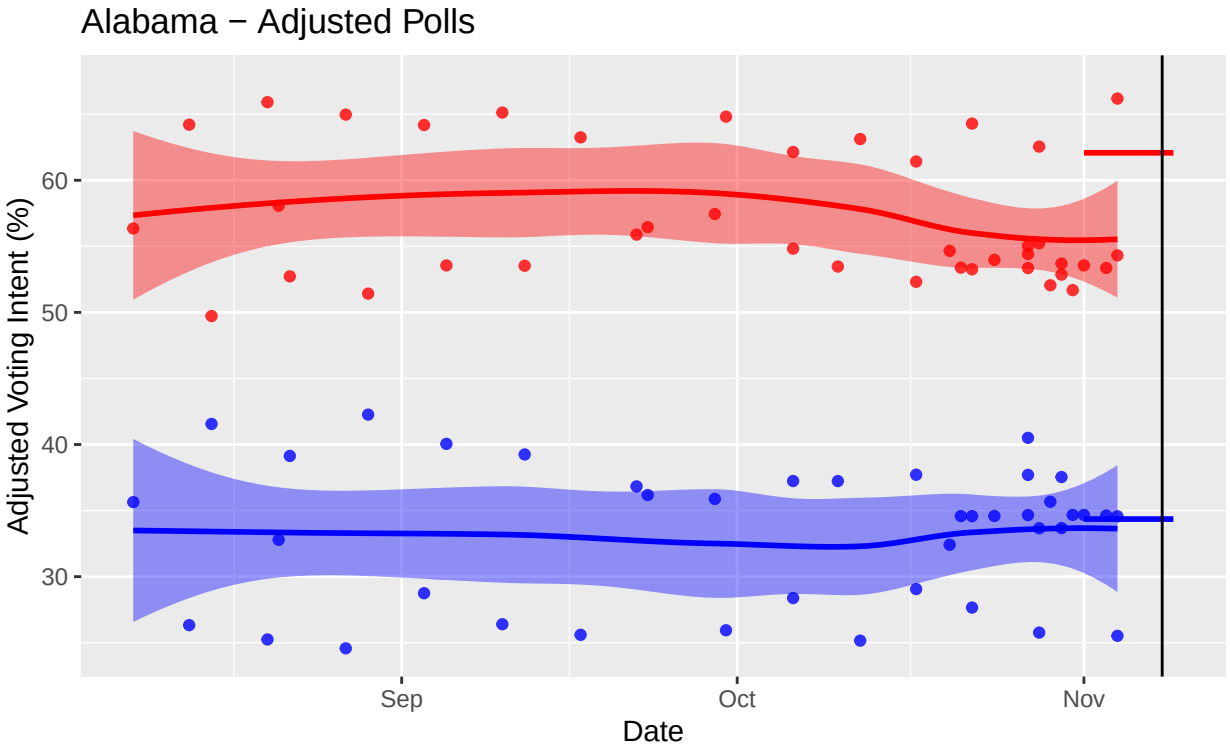
```
                colour = thecandidate$colour,
                alpha = 0.8,
                na.rm = TRUE
            ) + geom_smooth(
                mapping = datevsraw,
                colour = thecandidate$colour,
                fill = thecandidate$colour,
                alpha = 0.4,
                na.rm = TRUE
            ) + geom_segment(
                mapping = aes(
                x = ymd("2016-11-01"),
                y = voteresult,
                xend = dayafter,
                yend = voteresult
                ),
                colour = thecandidate$colour,
                linewidth = 1.0
            )
        ) # end of return
    } # end of if
    else return(last_plot()) # catch the case where the candidate was not polled in the given state
}

candidateplotadj = function(thestate, thecandidate, thefirstdate) {

    statepolls = thecandidate$polls |>
        filter(state == thestate) # candidate's polling numbers in the given state

    # methods to avoid plotting Johnson's numbers where his polls are insignificant
    # check whether Johnson obtained more than 13% (raw! for comparison to plots of raw polls) in any p
    if (thecandidate$name == "johnson") {
        statepolls = statepolls |>
            filter(middate >= ymd("2016-09-01"))
        if (max(statepolls$rawpolls, na.rm = TRUE)<=13.0) {
            return(last_plot())
        }
    }

    if ( # in this state, not all polls for this candidate are NA
        any(!is.na(statepolls$adjpolls))
    ) { # so we can display this candidate's polls here
        datevsadj = aes(
            x = statepolls$middate,
            y = statepolls$adjpolls
        )
        voteresult = filter(thecandidate$finalresults, state == thestate)[1,2] |>
            as.numeric()
        return(
            last_plot() + geom_point(
                mapping = datevsadj,
                colour = thecandidate$colour,
                alpha = 0.8,
```

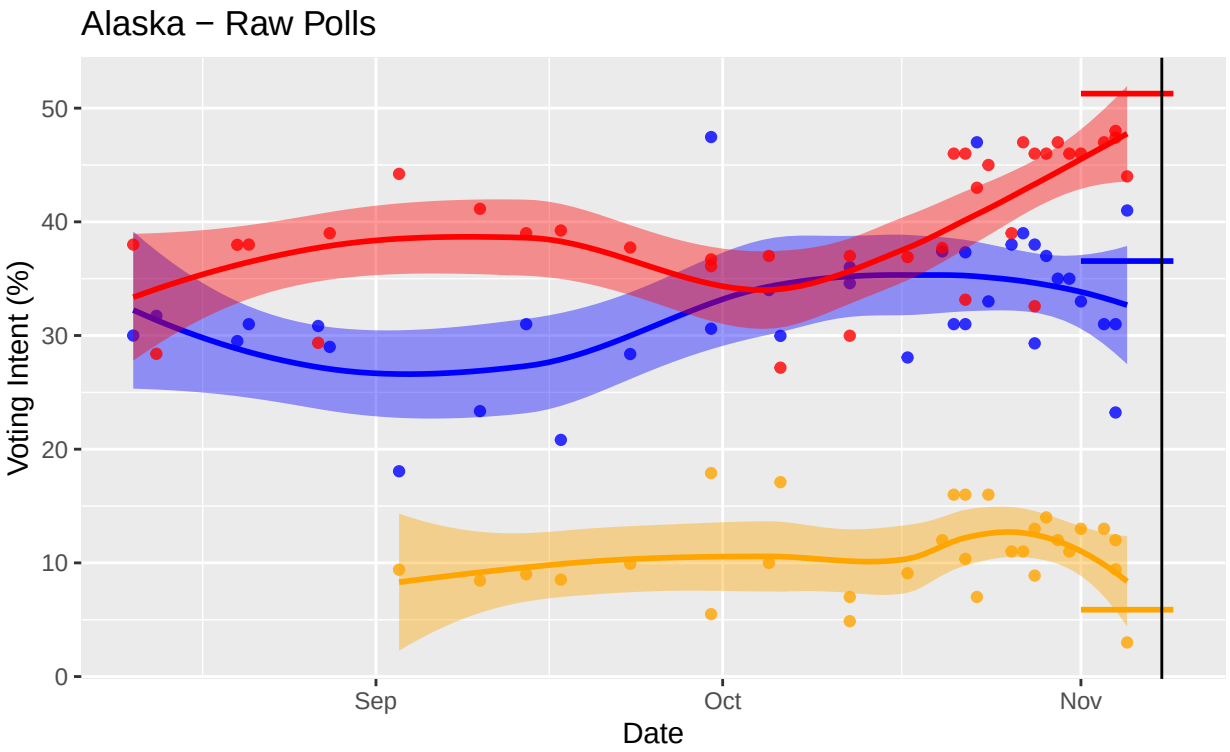
```
                na.rm = TRUE
            ) + geom_smooth(
                mapping = datevsadj,
                colour = thecandidate$colour,
                fill = thecandidate$colour,
                alpha = 0.4,
                na.rm = TRUE
            ) + geom_segment(
                mapping = aes(
                x = ymd("2016-11-01"),
                y = voteresult,
                xend = dayafter,
                yend = voteresult
                ),
                colour = thecandidate$colour,
                linewidth = 1.0
            )
        ) # end of return
    } # end of if
    else return(last_plot()) # catch the case where the candidate was not polled in the given state
}
```

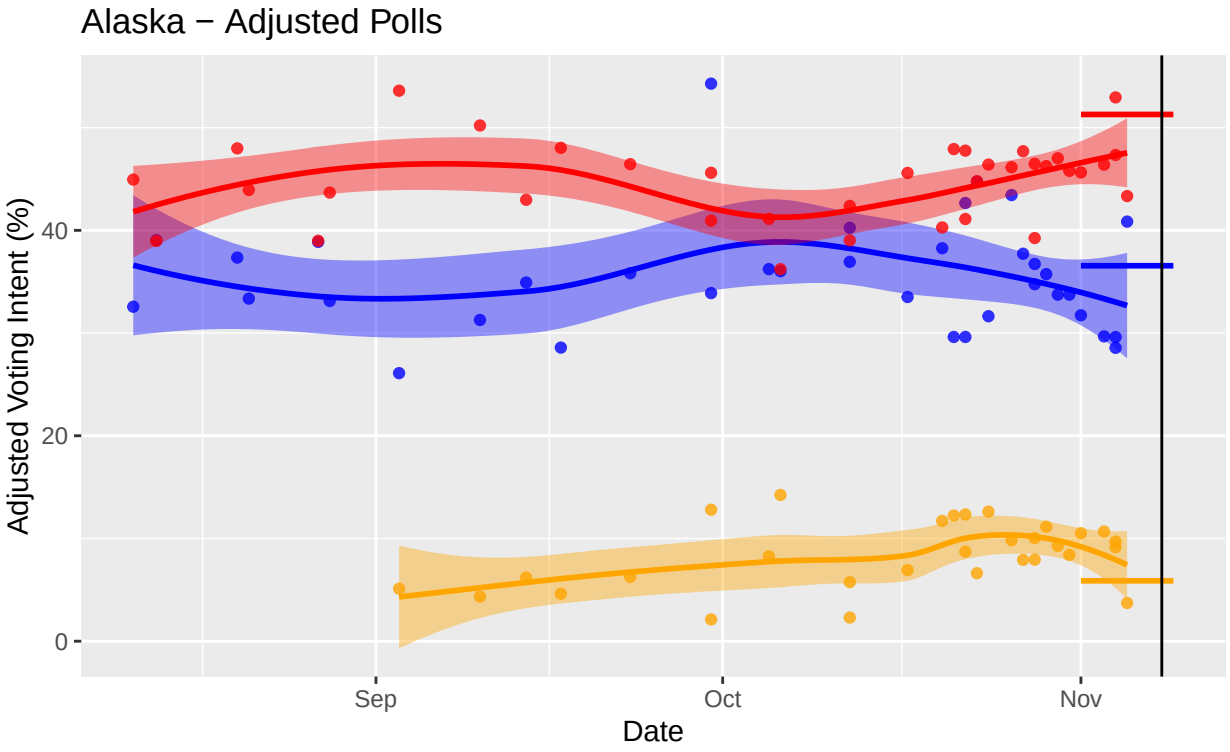
```
stateplotraw = function(thestate, firstdatetoplot) {
    plot = ggplot()
    for (cand in candidates){
        # Mcmullin was not a factor outside of Utah
        # don't attempt to plot his polls in other states
        if(cand$name != "mcmullin" || thestate == "Utah"){
            plot = candidateplotraw(thestate, cand, firstdatetoplot)
        }
    } # end of for
    stateraw = thestate |>
        paste("Raw Polls", sep = " - ")

    return(
        plot + geom_vline(xintercept = finaldate) + labs(
            title = stateraw,
            x = "Date",
            y = "Voting Intent (%)"
        ) + xlim(firstdatetoplot, dayafter)
    )
}

stateplotadj = function(thestate, firstdatetoplot) {
    plot = ggplot()
    for (cand in candidates){
        plot = candidateplotadj(thestate, cand, firstdatetoplot)
    }
    stateadj = thestate |>
        paste("Adjusted Polls", sep = " - ")

    return(
        plot + geom_vline(xintercept = finaldate) + labs(
```

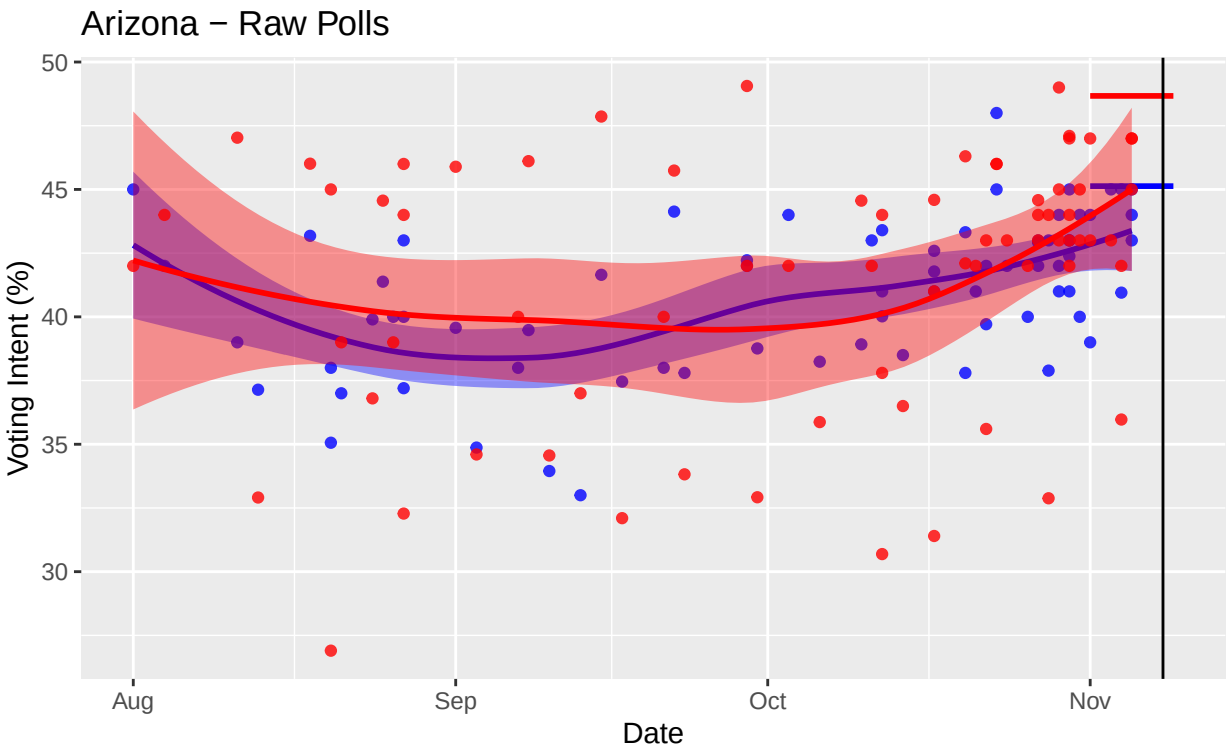
```
            title = stateadj,
            x = "Date",
            y = "Adjusted Voting Intent (%)"
        ) + xlim(firstdatetoplot, dayafter)
    )
}
```

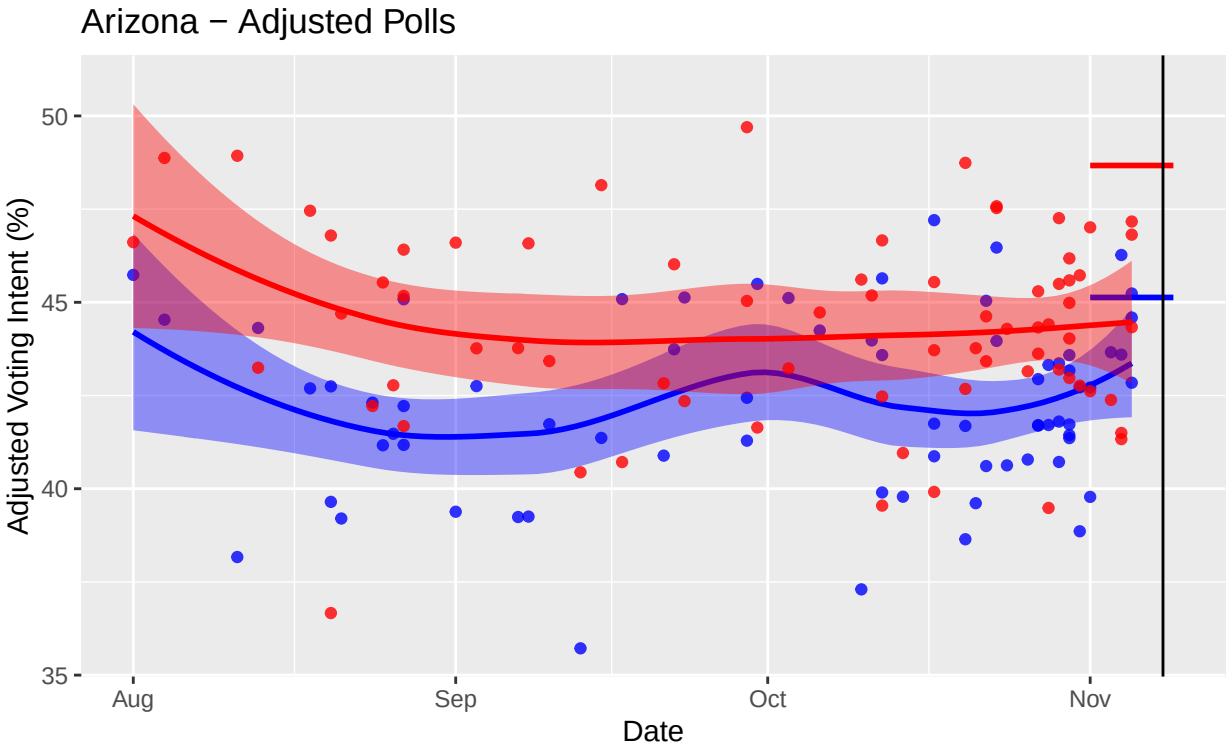

Alabama − Raw Polls
Voting Intent (%)
70
60
50
40
30
20
Sep
Oct
Nov
Date

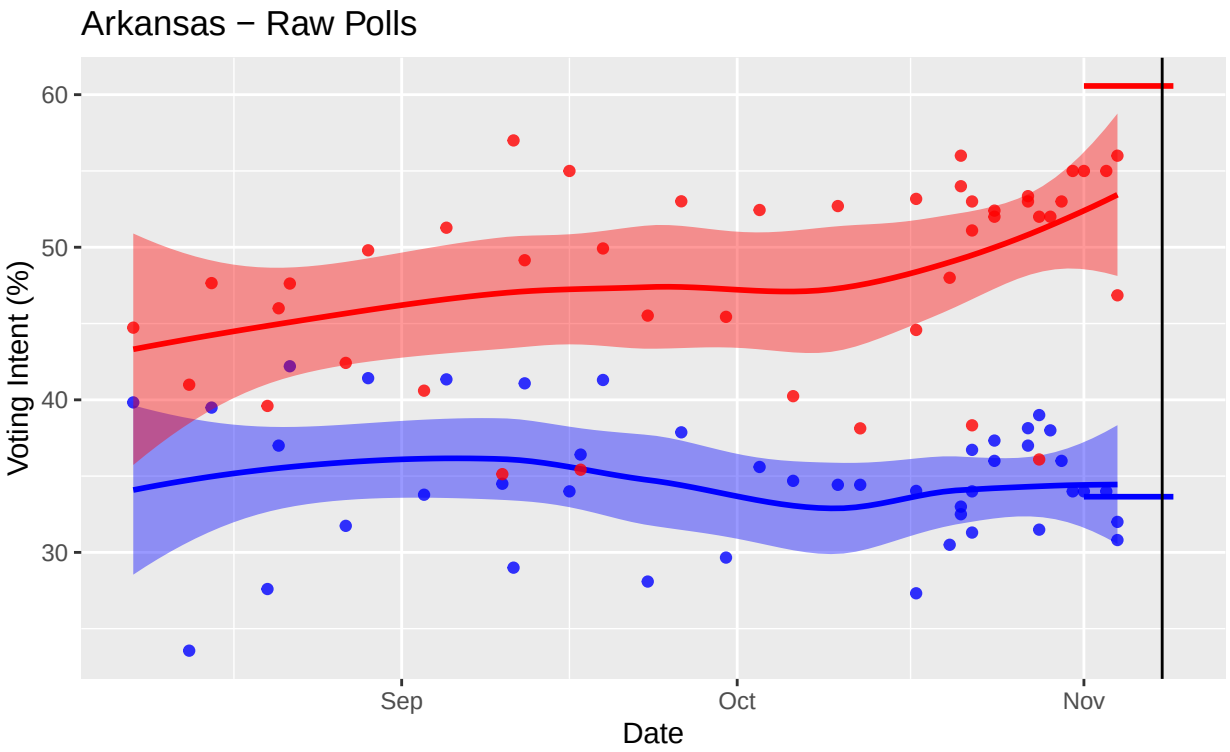

Alabama − Adjusted Polls
Adjusted Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

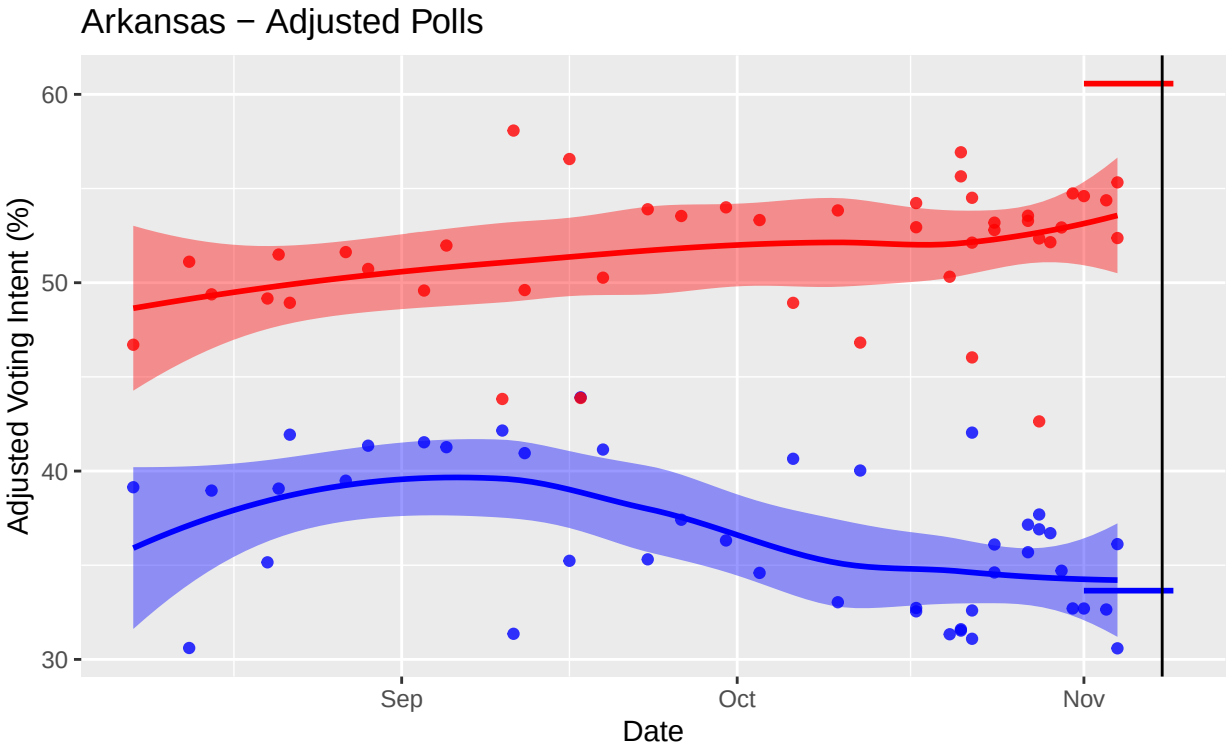

Alaska − Raw Polls
Voting Intent (%)
50
40
30
20
10
0
Sep
Oct
Nov
Date

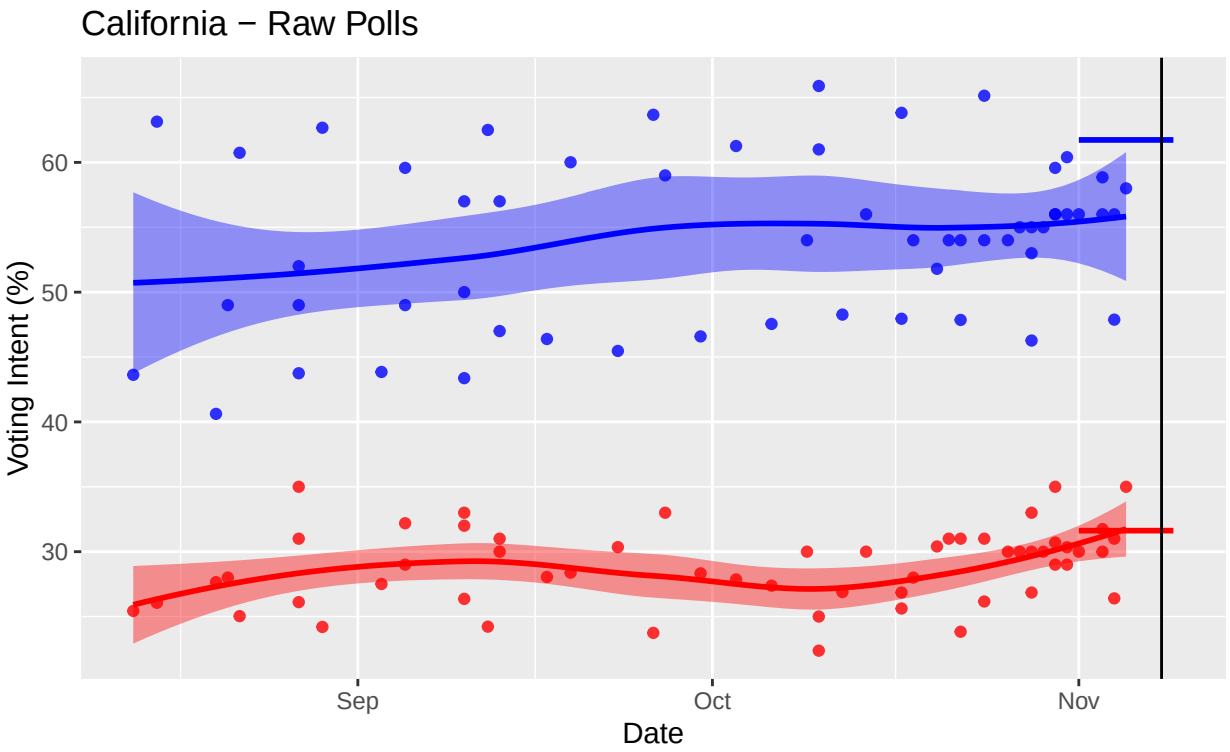

Alaska − Adjusted Polls
Adjusted Voting Intent (%)
40
20
0
Sep
Oct
Nov
Date

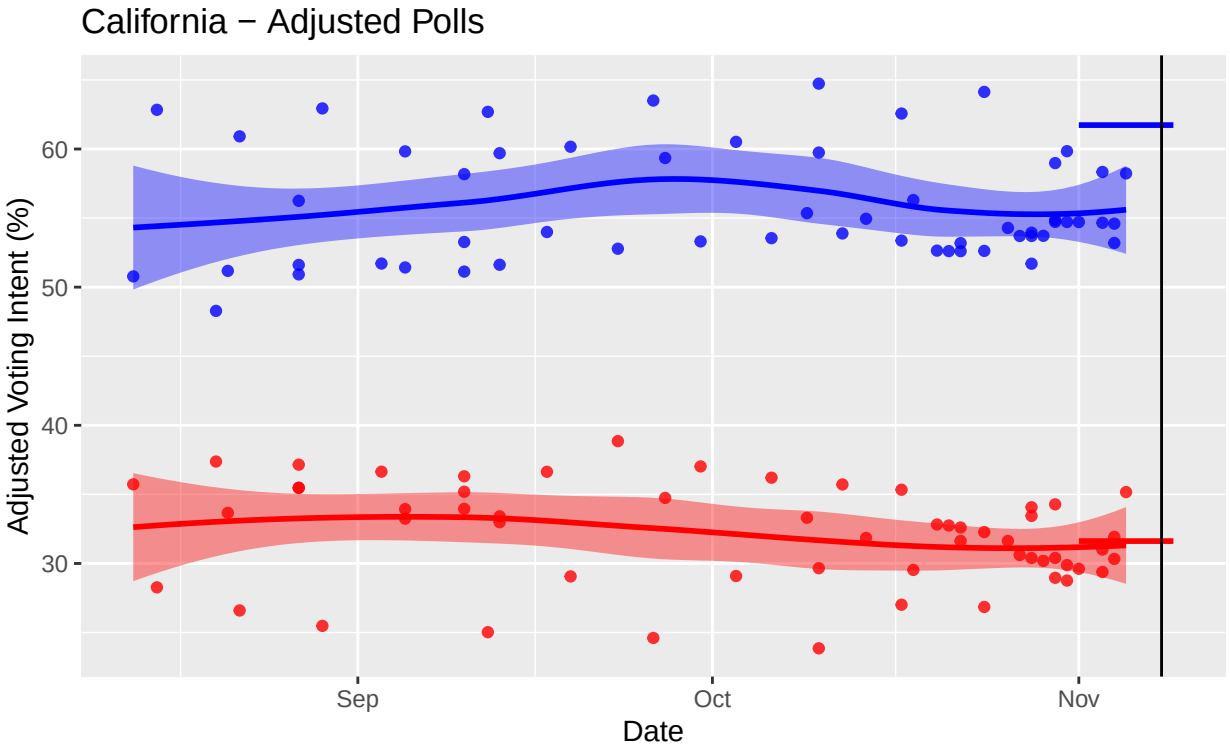

Arizona − Raw Polls
Voting Intent (%)
50
45
40
35
30
Aug
Sep
Oct
Nov
Date

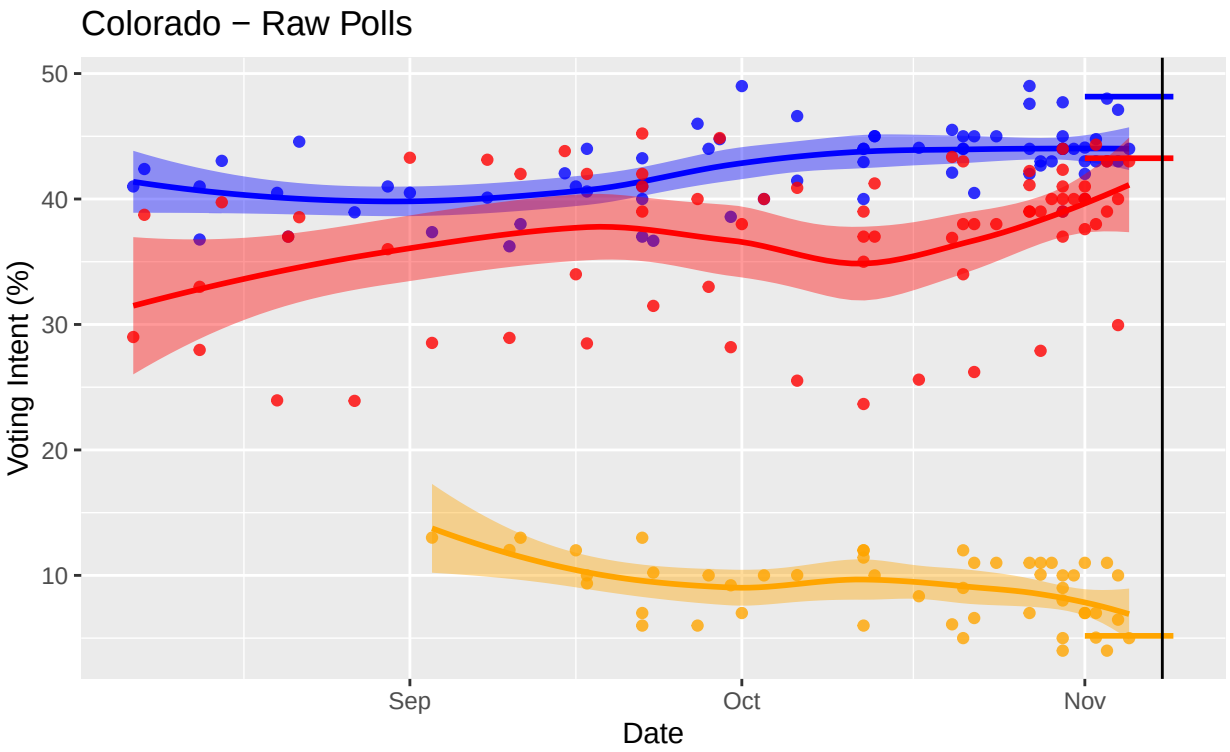

Arizona − Adjusted Polls
Adjusted Voting Intent (%)
50
45
40
35
Aug
Sep
Oct
Nov
Date

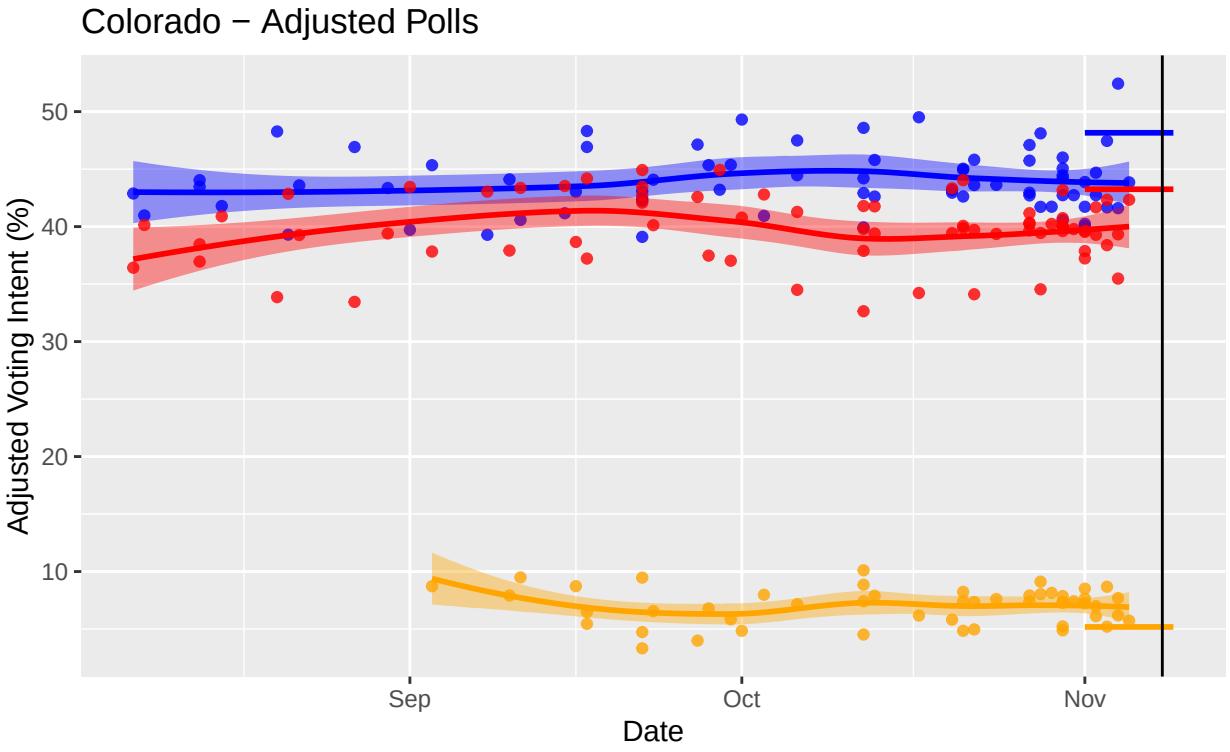

Arkansas − Raw Polls
Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

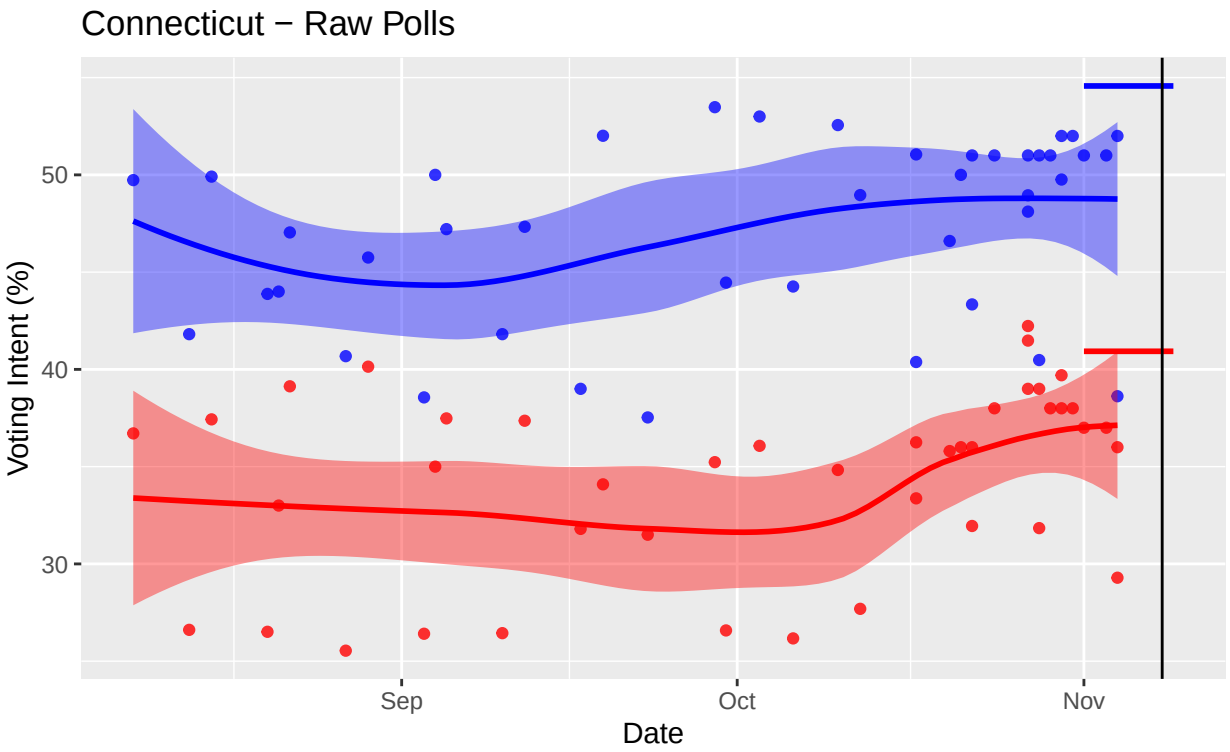

Arkansas − Adjusted Polls
Adjusted Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

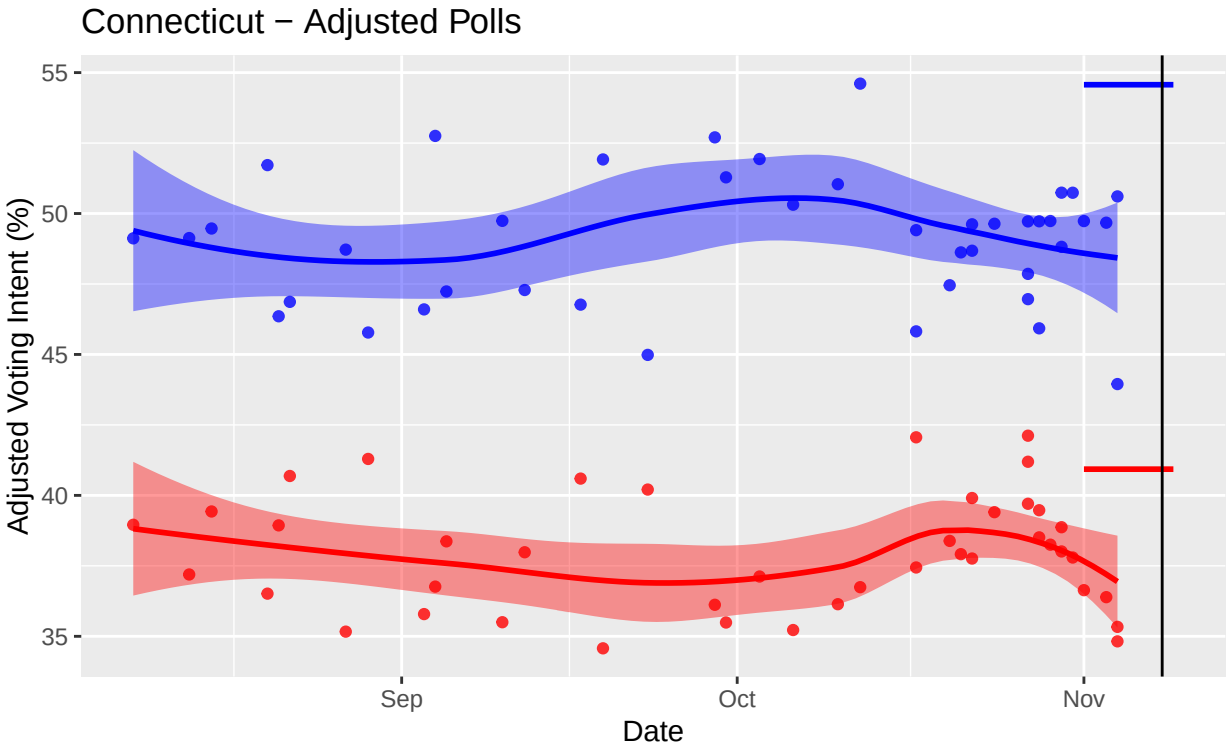

California − Raw Polls
Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

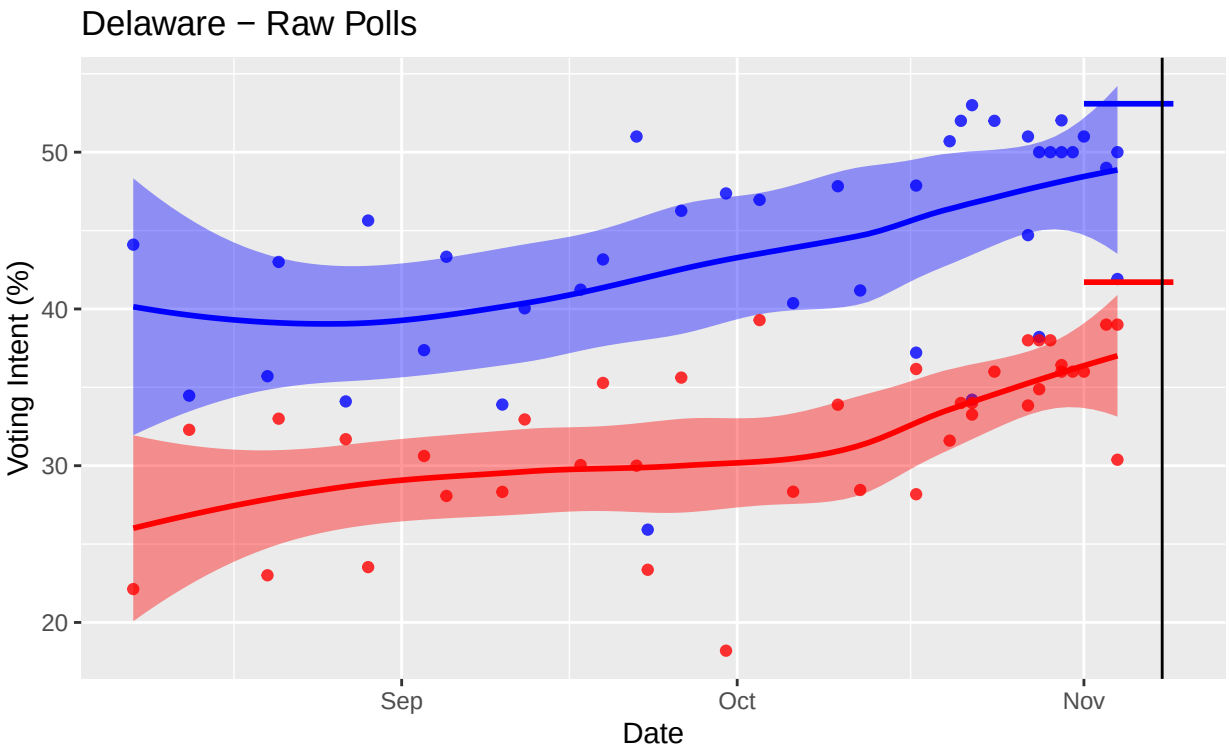

California − Adjusted Polls
Adjusted Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

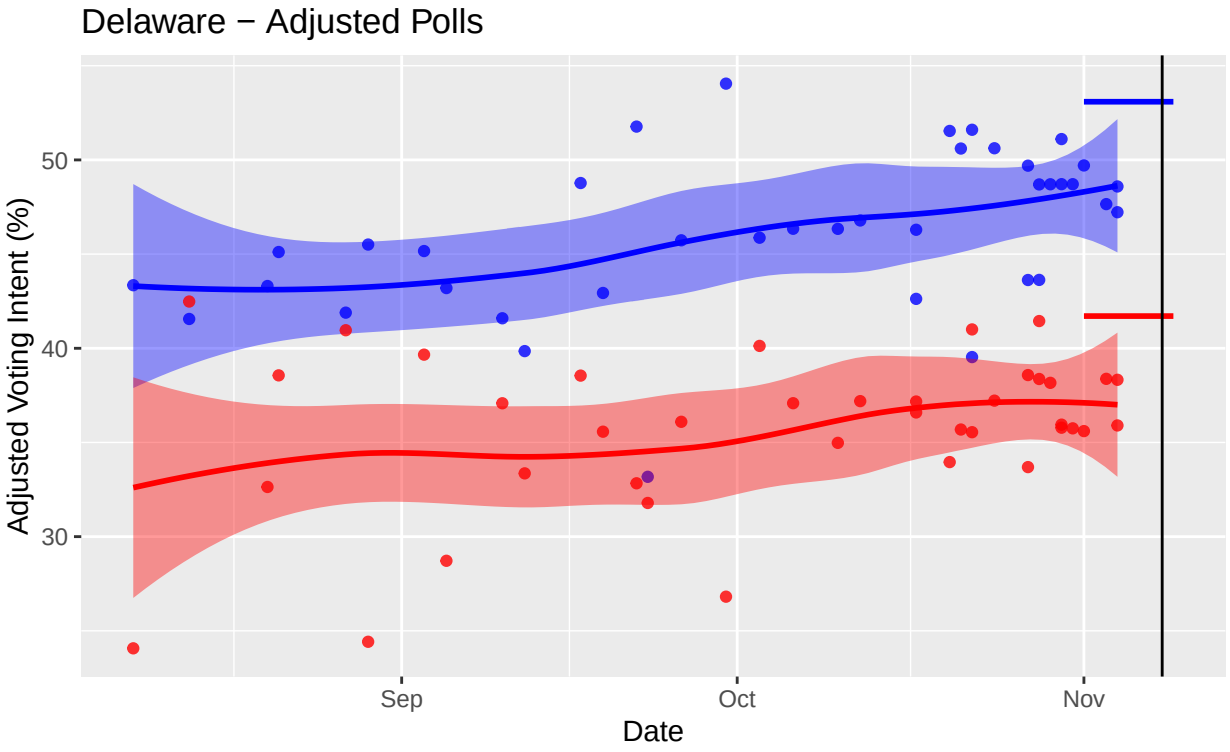

Colorado − Raw Polls
Voting Intent (%)
50
40
30
20
10
Sep
Oct
Nov
Date

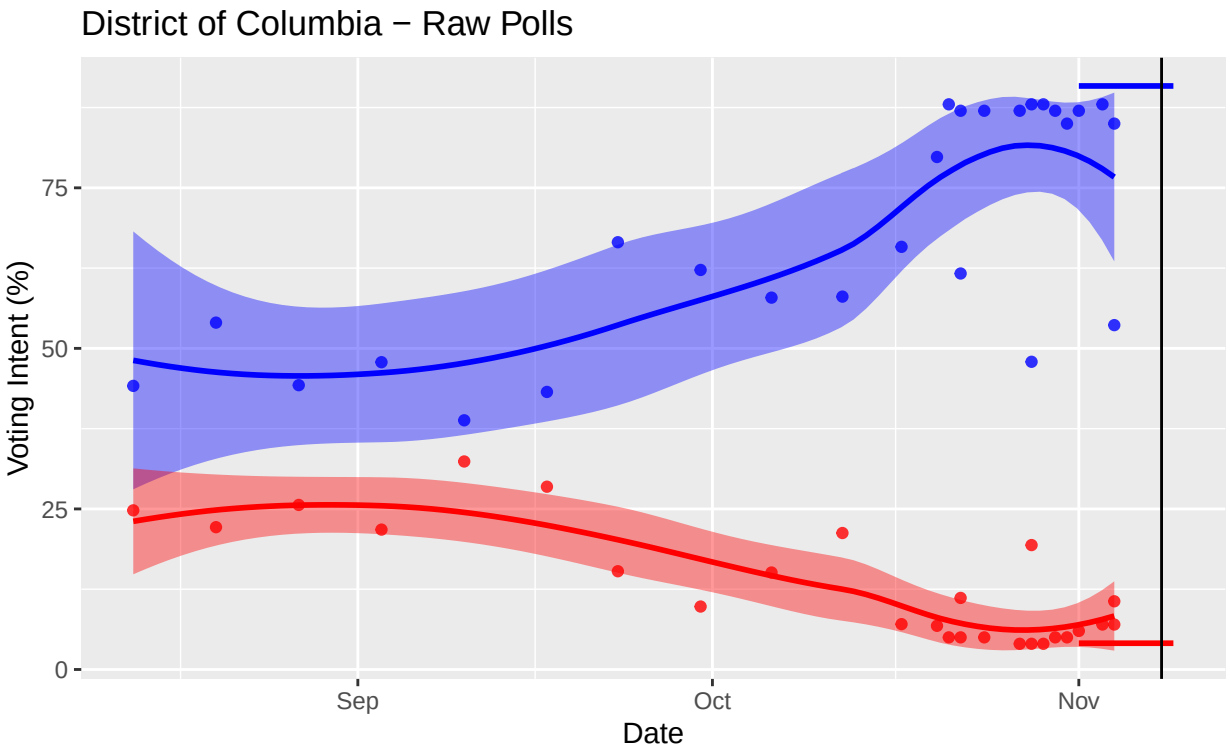

Colorado − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
20
10
Sep
Oct
Nov
Date

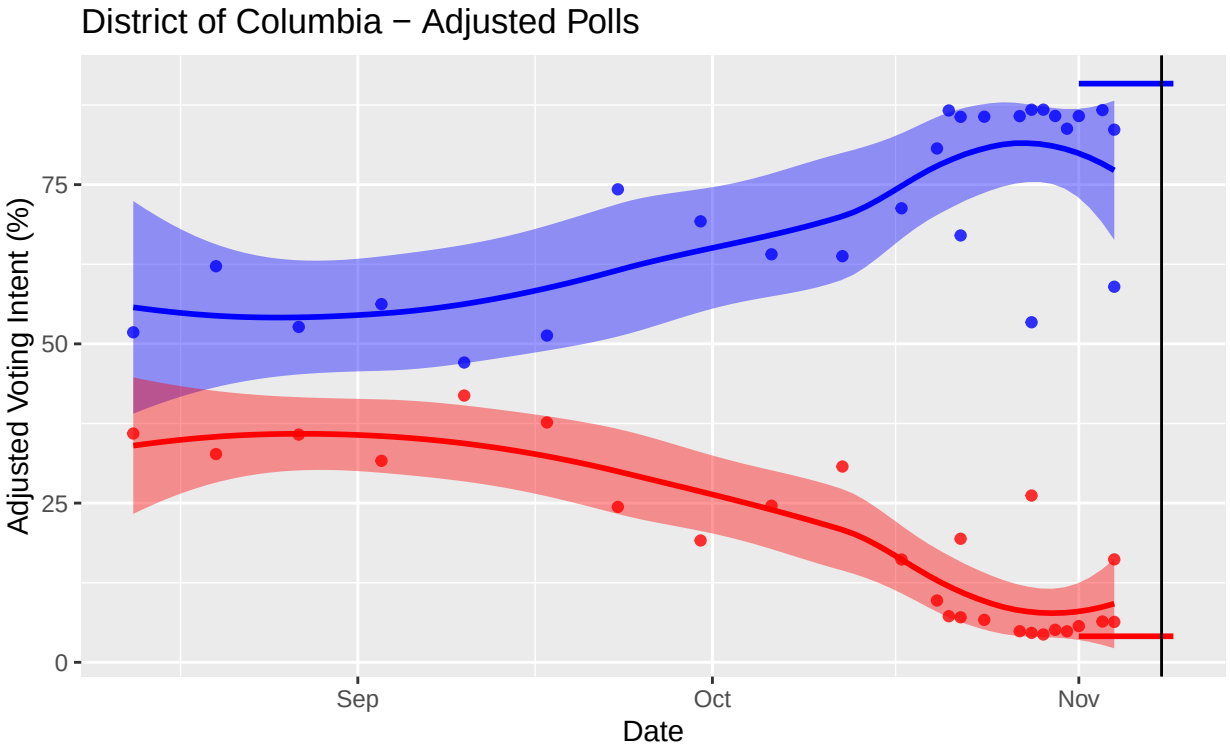

Connecticut − Raw Polls
Voting Intent (%)
50
40
30
Sep
Oct
Nov
Date

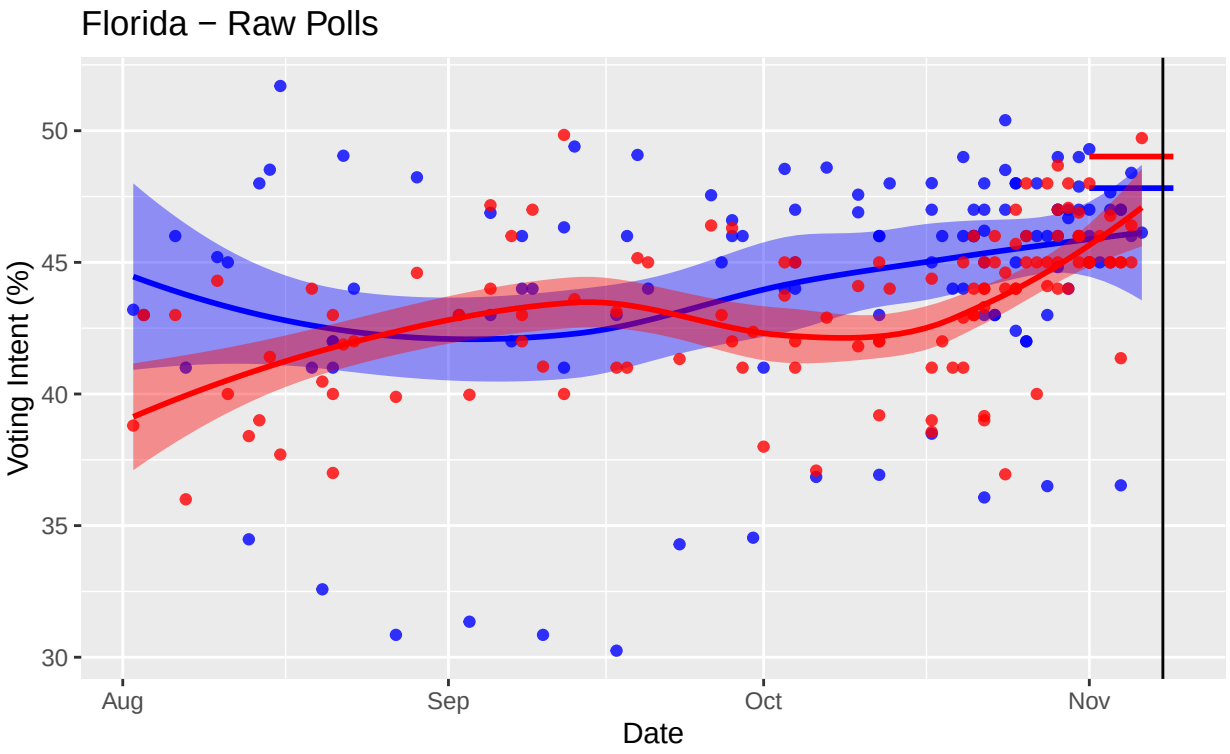

Connecticut − Adjusted Polls
Adjusted Voting Intent (%)
55
50
45
40
35
Sep
Oct
Nov
Date

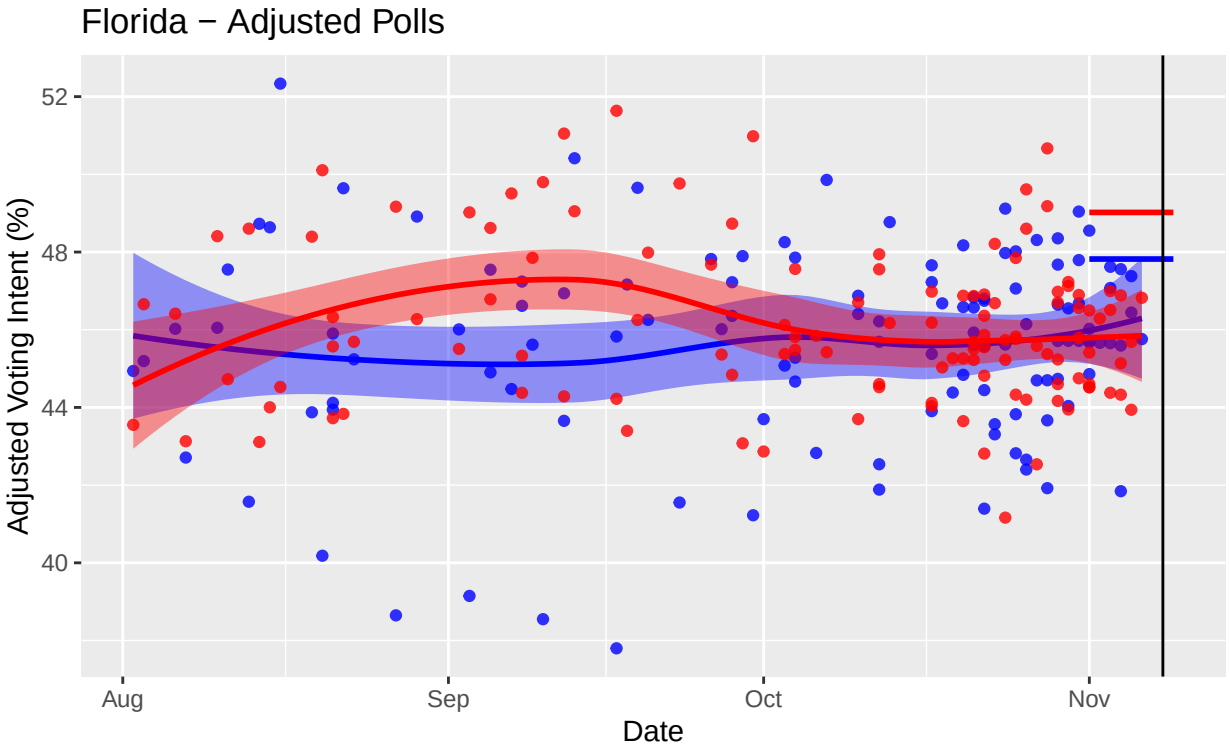

Delaware − Raw Polls
Voting Intent (%)
50
40
30
20
Sep
Oct
Nov
Date

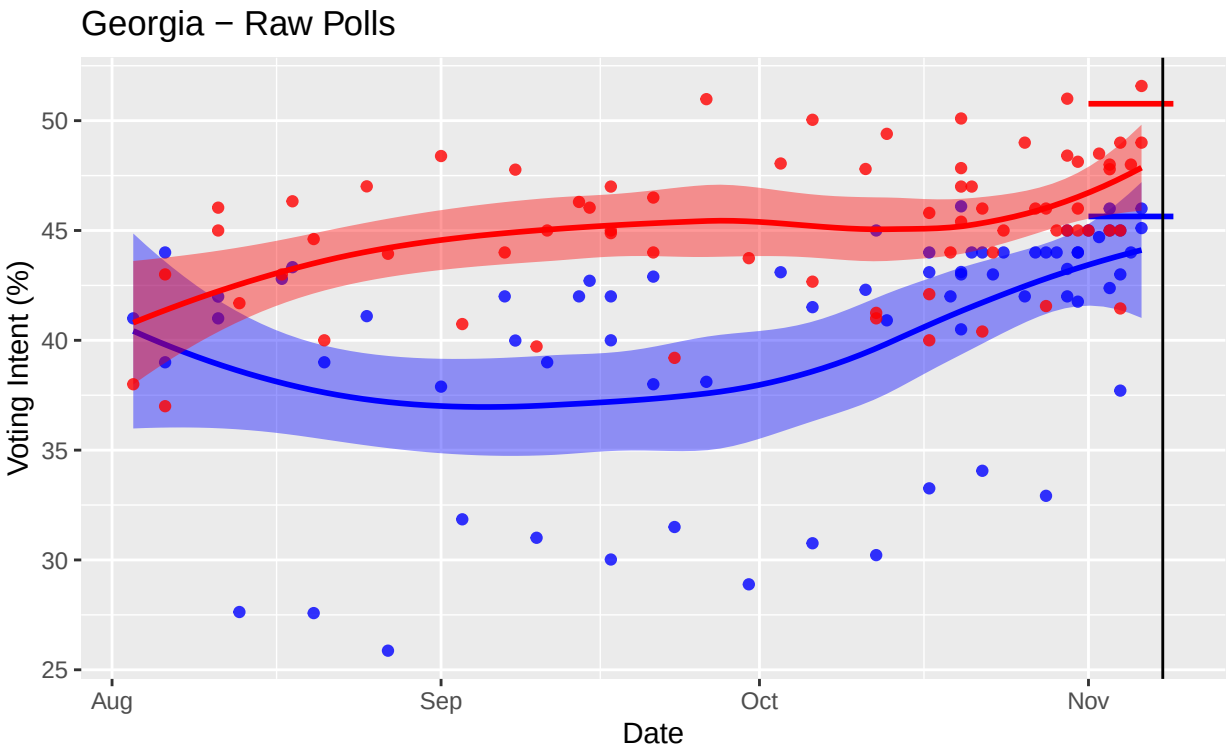

Delaware − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
Sep
Oct
Nov
Date

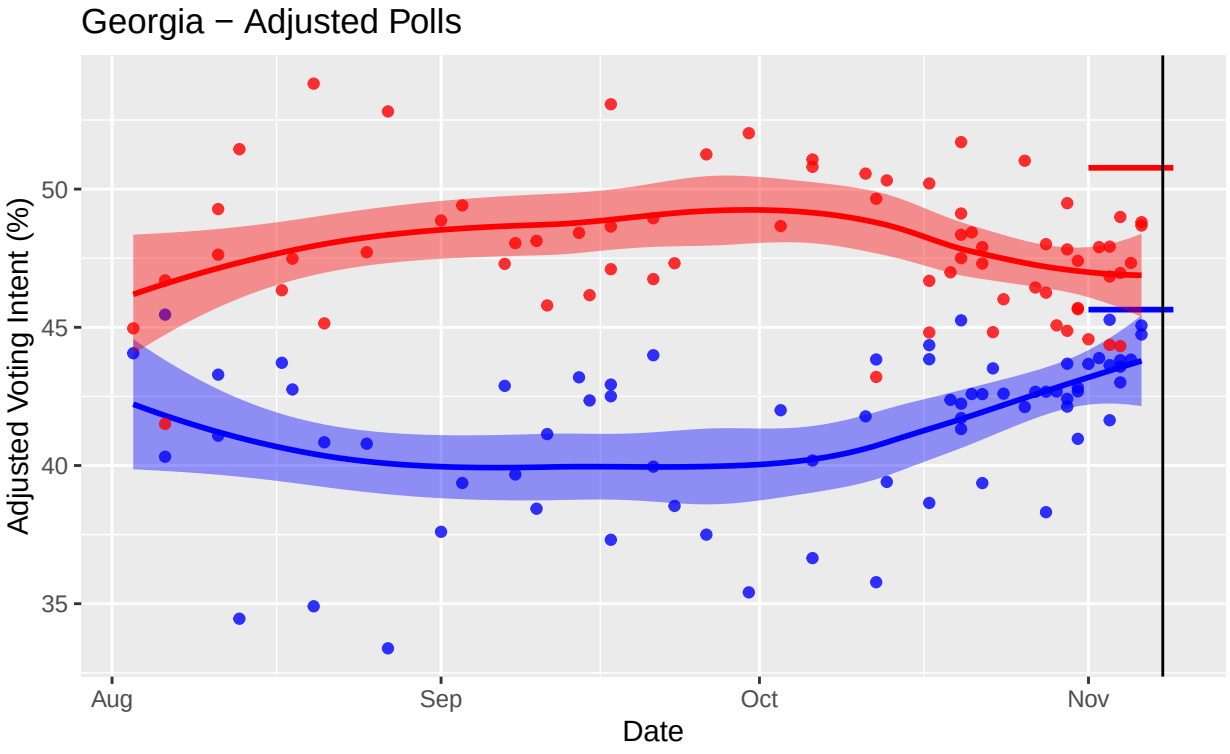

District of Columbia − Raw Polls
Voting Intent (%)
75
50
25
0
Sep
Oct
Nov
Date

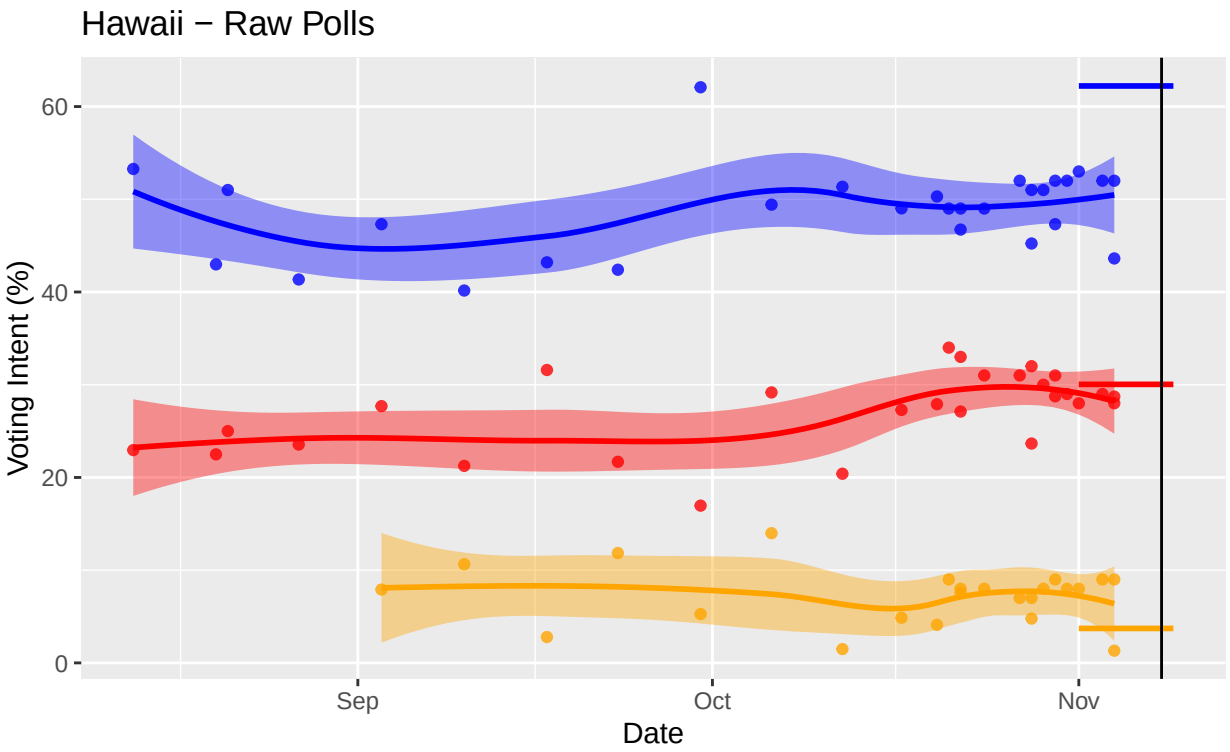

District of Columbia − Adjusted Polls
Adjusted Voting Intent (%)
75
50
25
0
Sep
Oct
Nov
Date

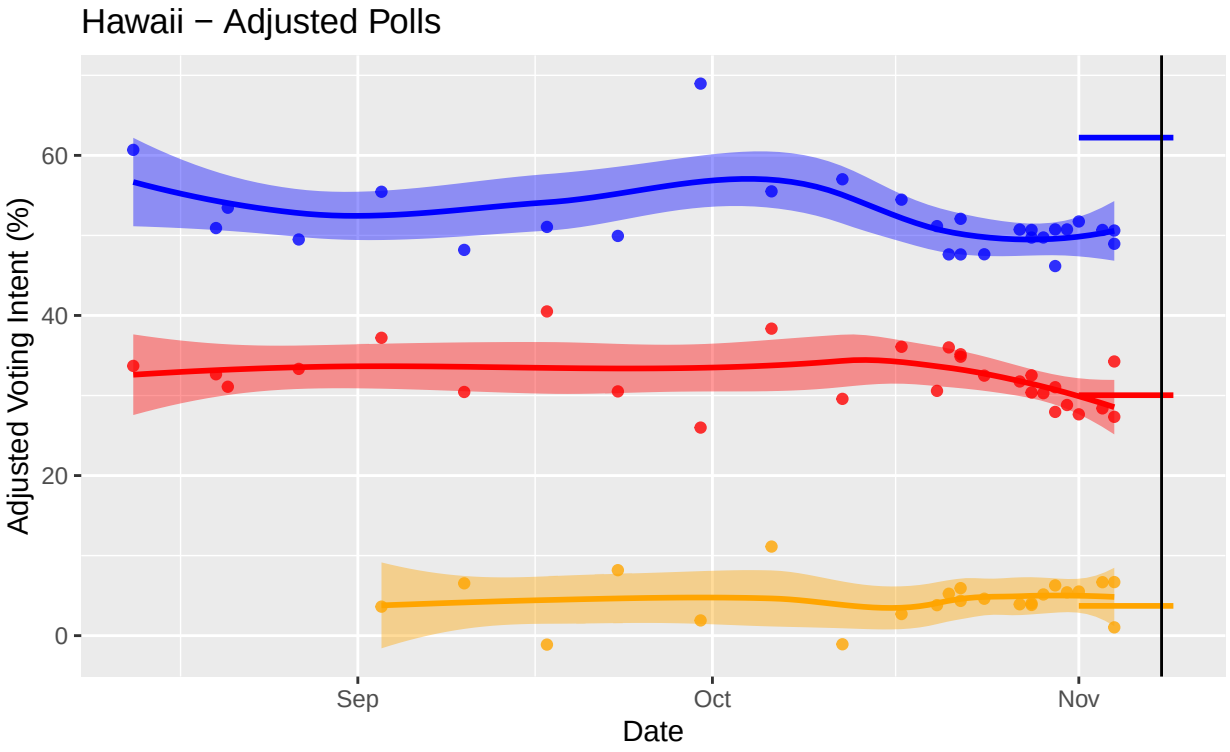

Florida − Raw Polls
Voting Intent (%)
50
45
40
35
30
Aug
Sep
Oct
Nov
Date

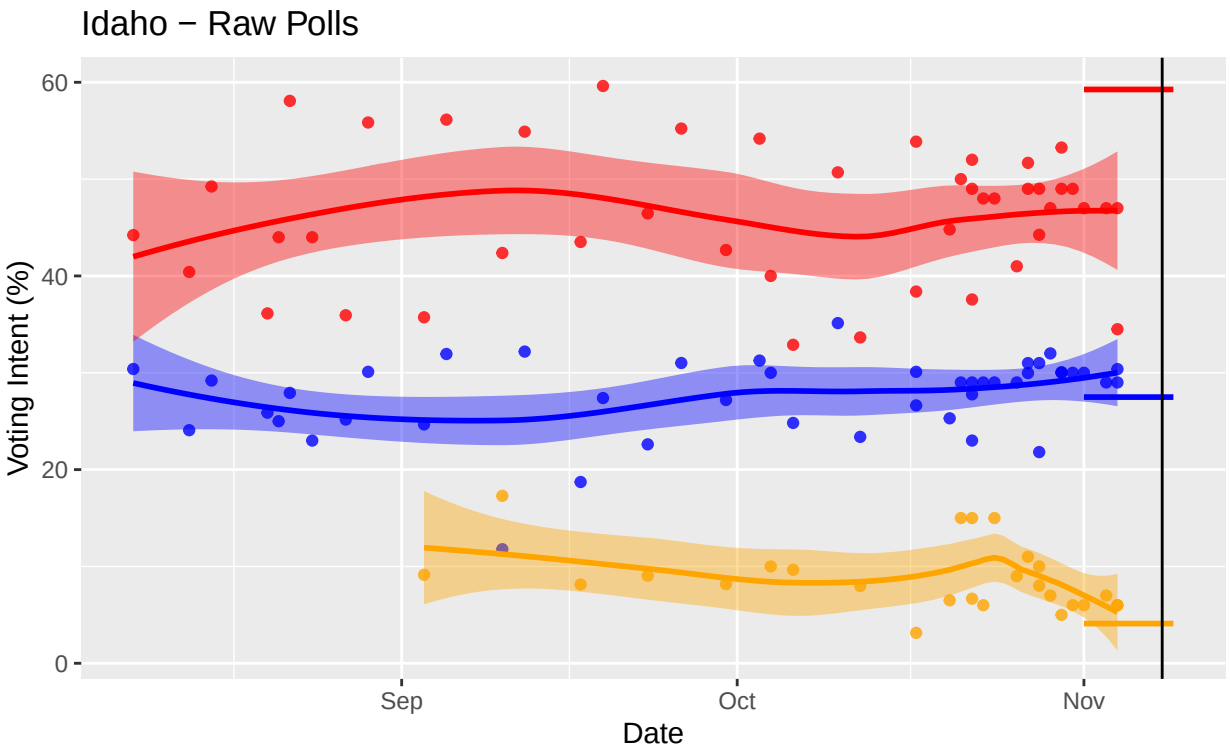

Florida − Adjusted Polls
Adjusted Voting Intent (%)
52
48
44
40
Aug
Sep
Oct
Nov
Date

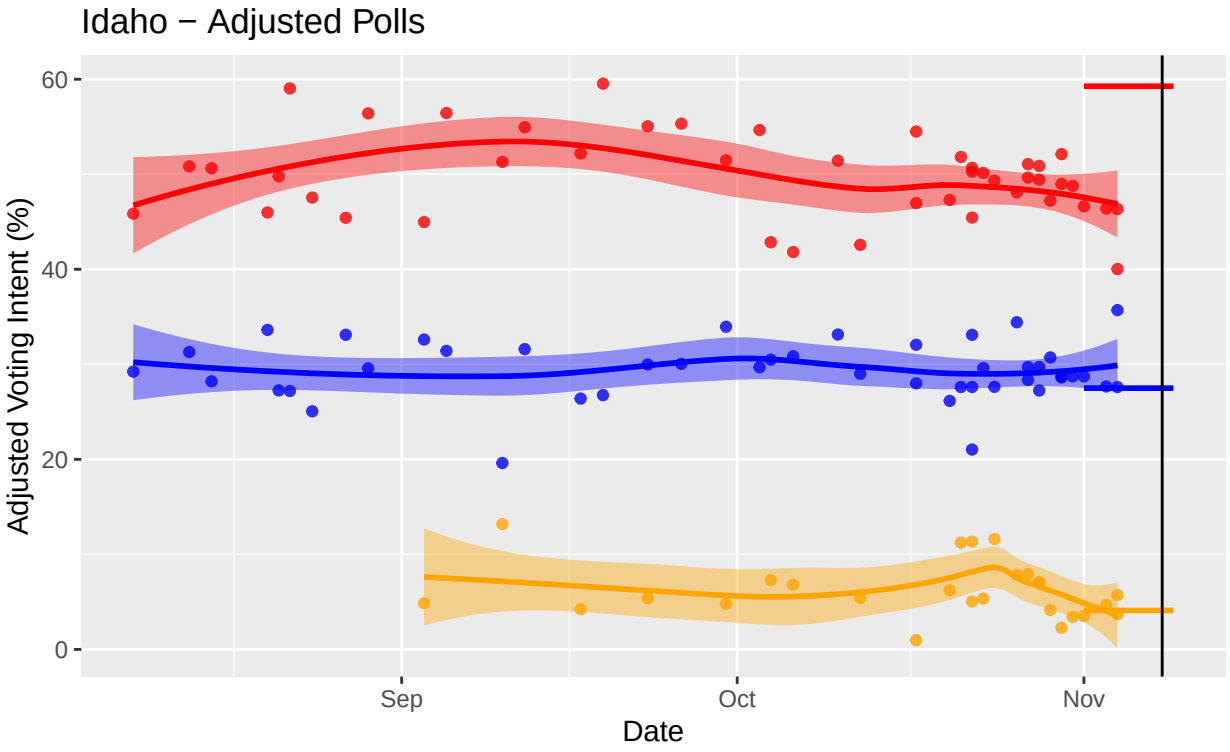

Georgia − Raw Polls
Voting Intent (%)
50
45
40
35
30
25
Aug
Sep
Oct
Nov
Date

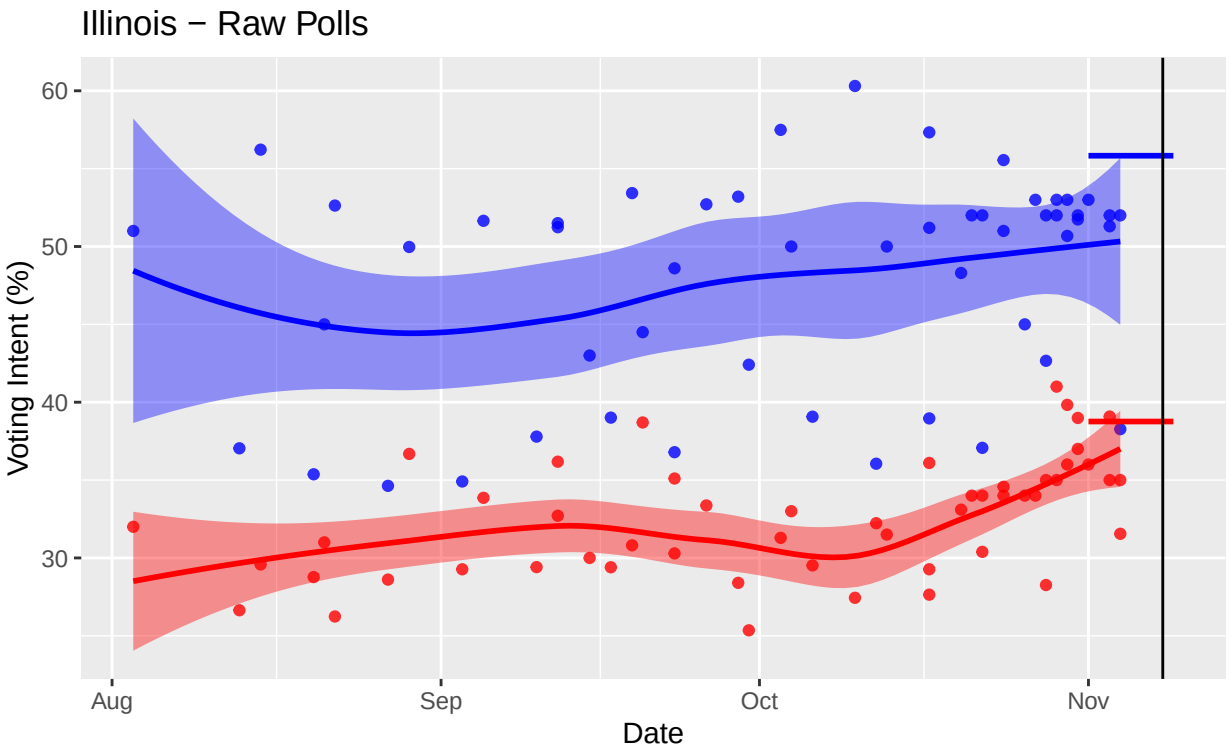

Georgia − Adjusted Polls
Adjusted Voting Intent (%)
50
45
40
35
Aug
Sep
Oct
Nov
Date

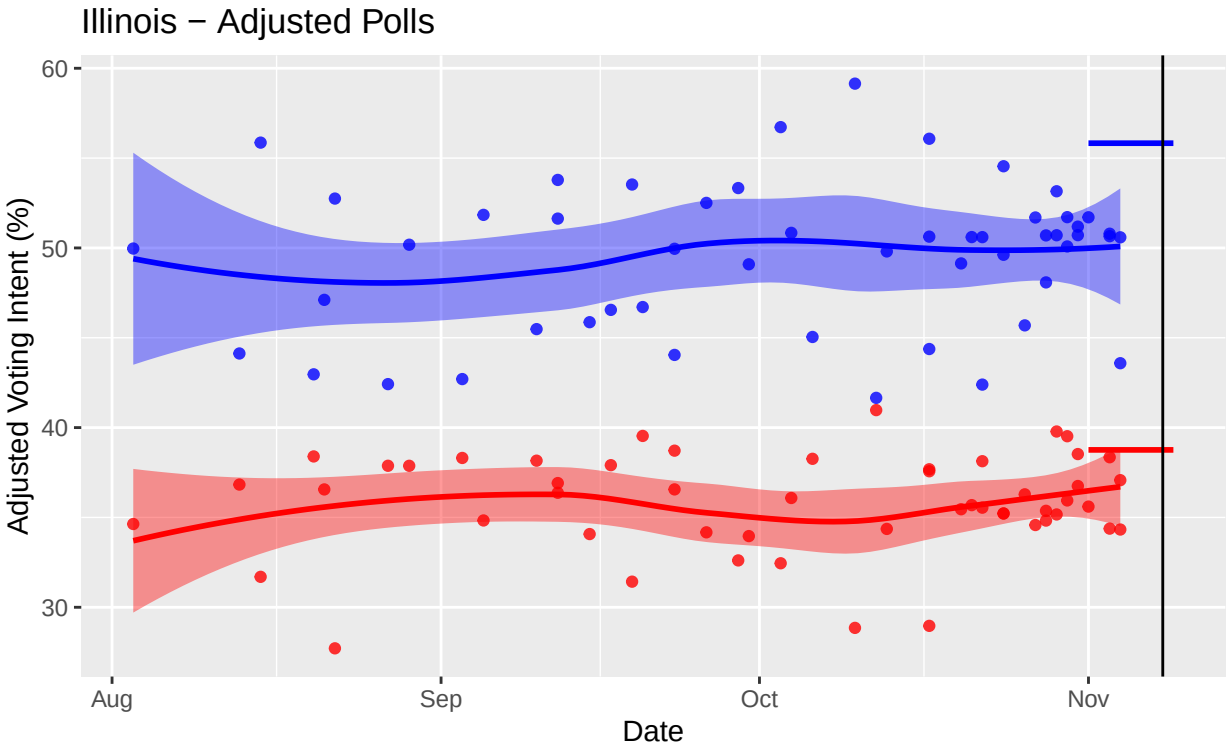

Hawaii − Raw Polls
Voting Intent (%)
60
40
20
0
Sep
Oct
Nov
Date

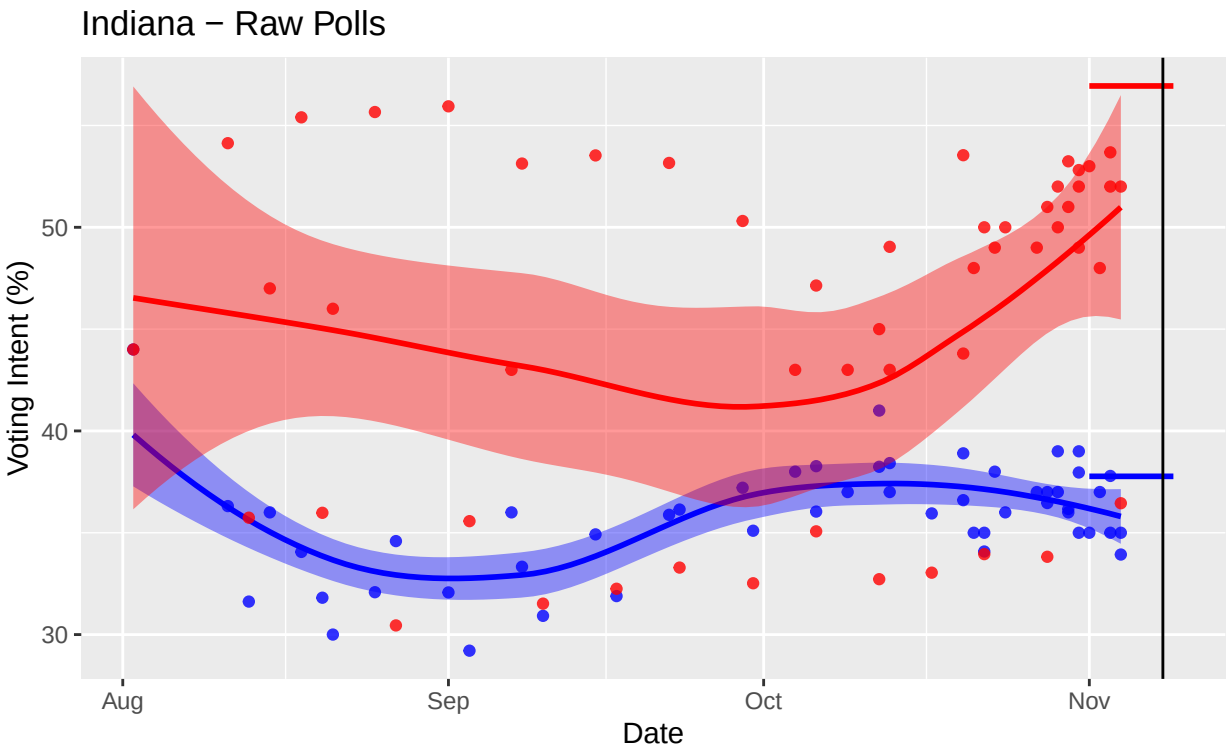

Hawaii − Adjusted Polls
Adjusted Voting Intent (%)
60
40
20
0
Sep
Oct
Nov
Date

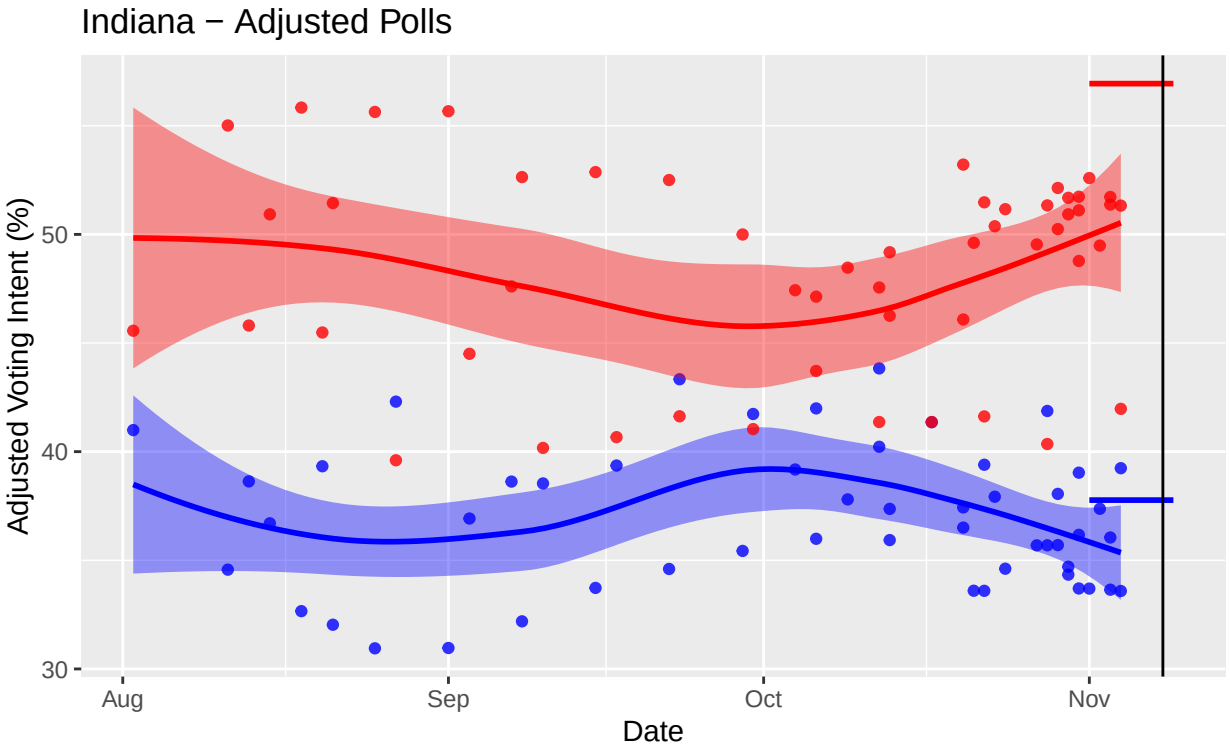

Idaho − Raw Polls
Voting Intent (%)
60
40
20
0
Sep
Oct
Nov
Date

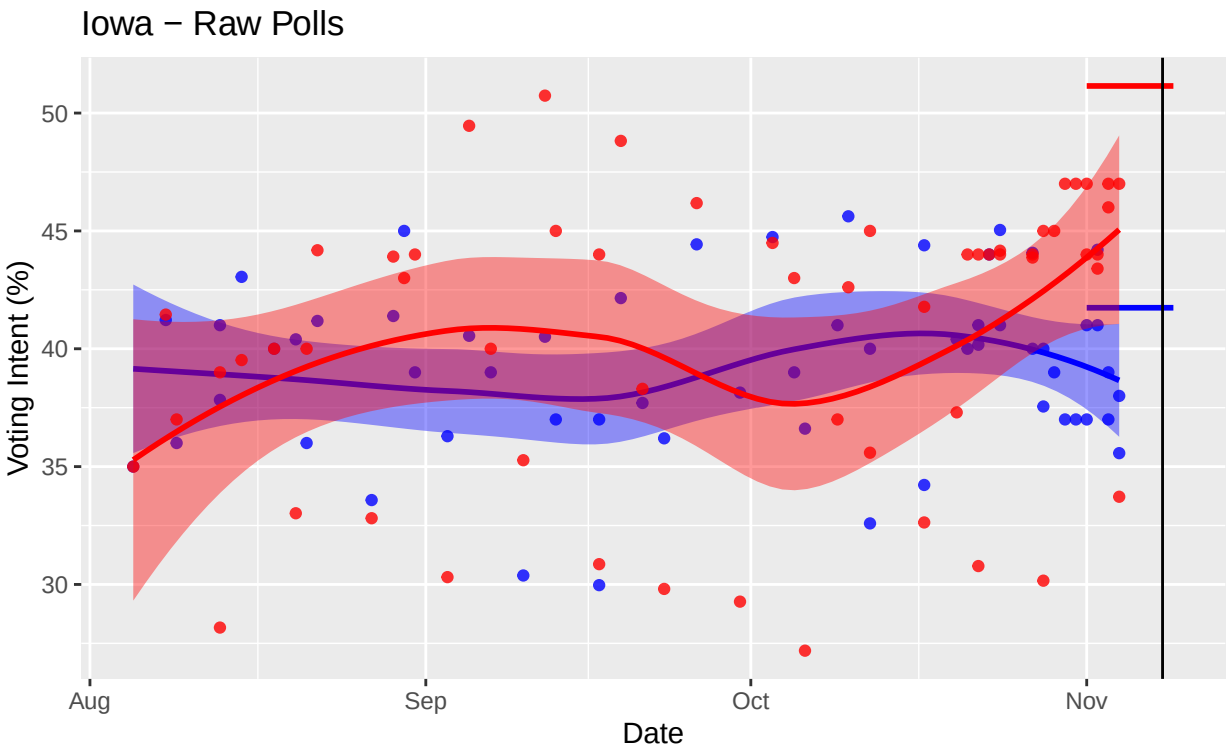

Idaho − Adjusted Polls
Adjusted Voting Intent (%)
60
40
20
0
Sep
Oct
Nov
Date

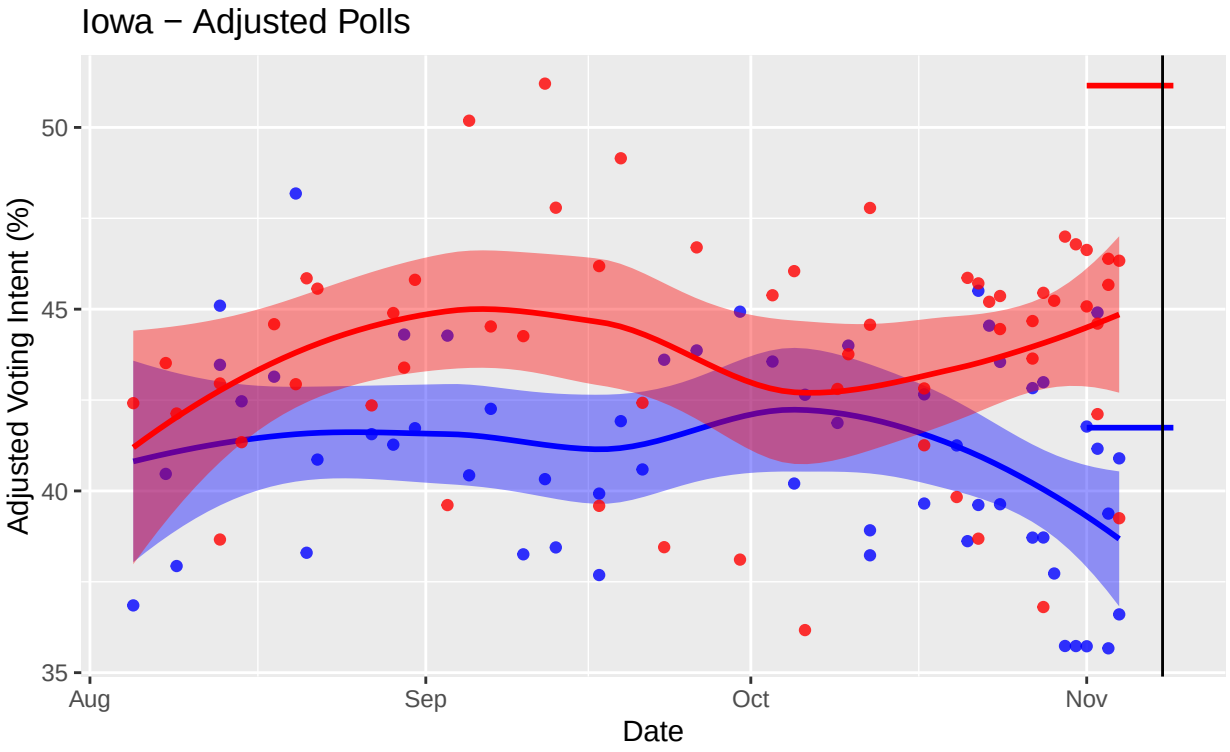

Illinois − Raw Polls
Voting Intent (%)
60
50
40
30
Aug
Sep
Oct
Nov
Date

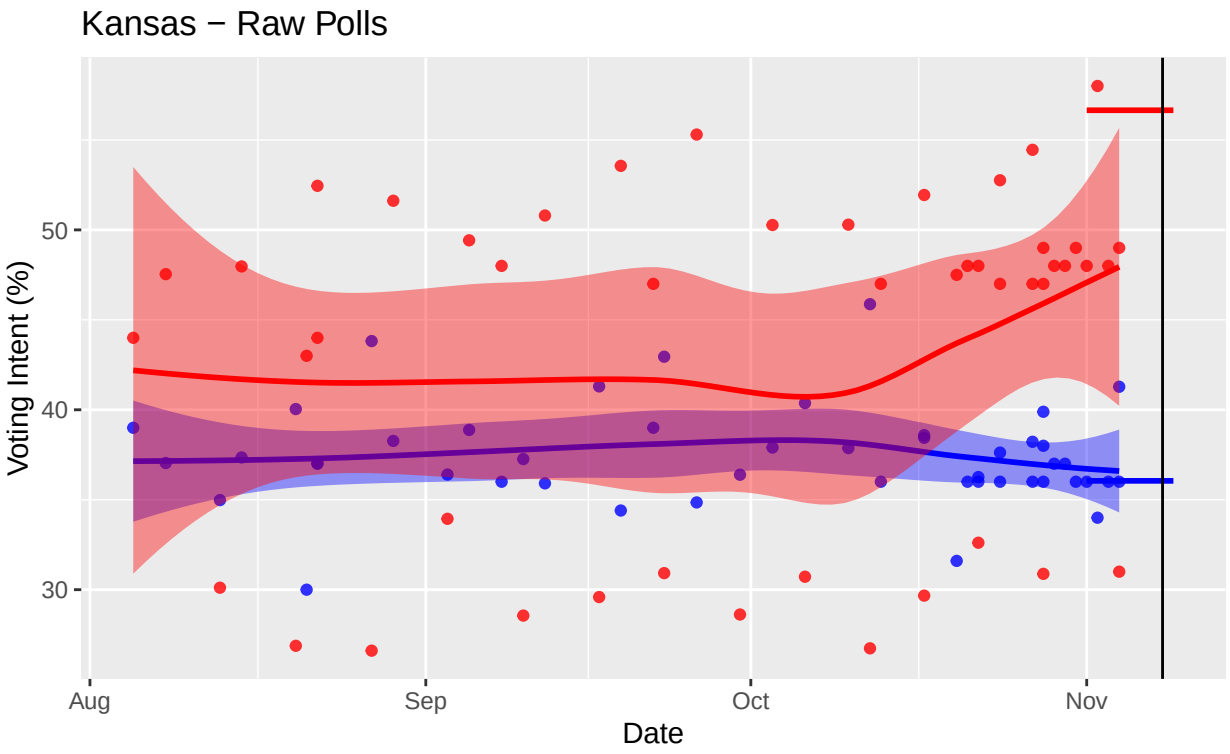

Illinois − Adjusted Polls
Adjusted Voting Intent (%)
60
50
40
30
Aug
Sep
Oct
Nov
Date

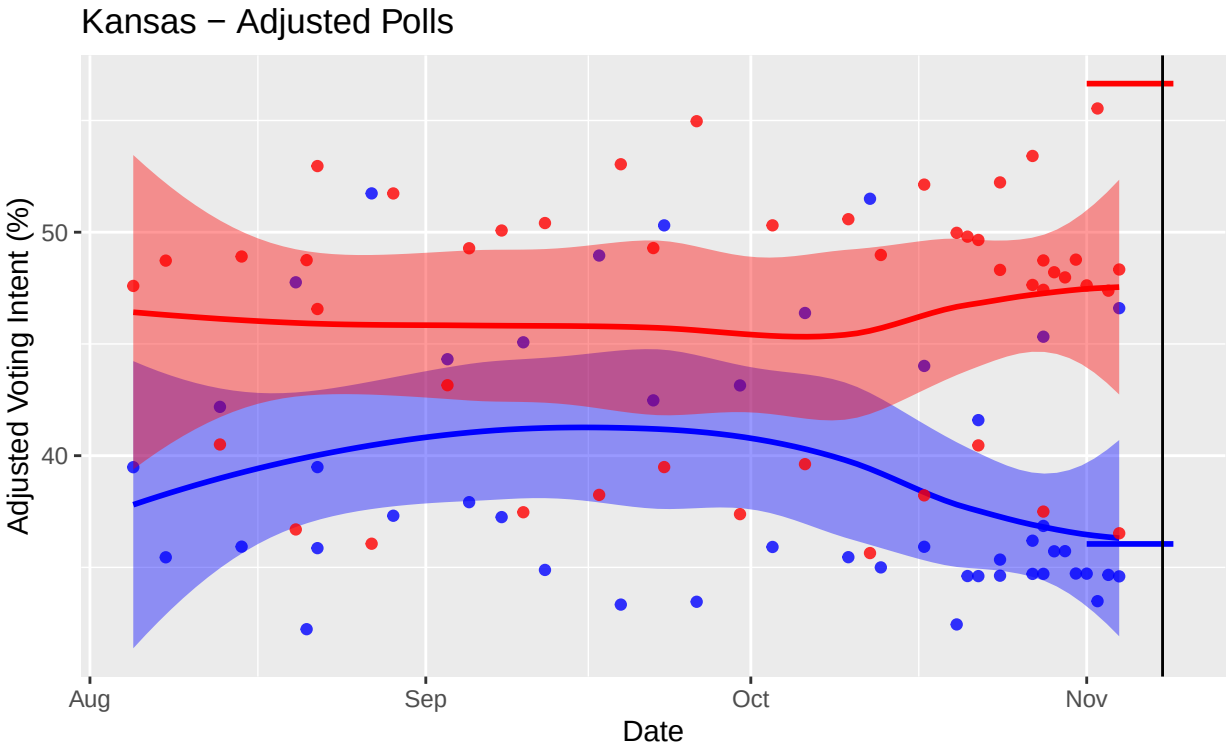

Indiana − Raw Polls
Voting Intent (%)
50
40
30
Aug
Sep
Oct
Nov
Date

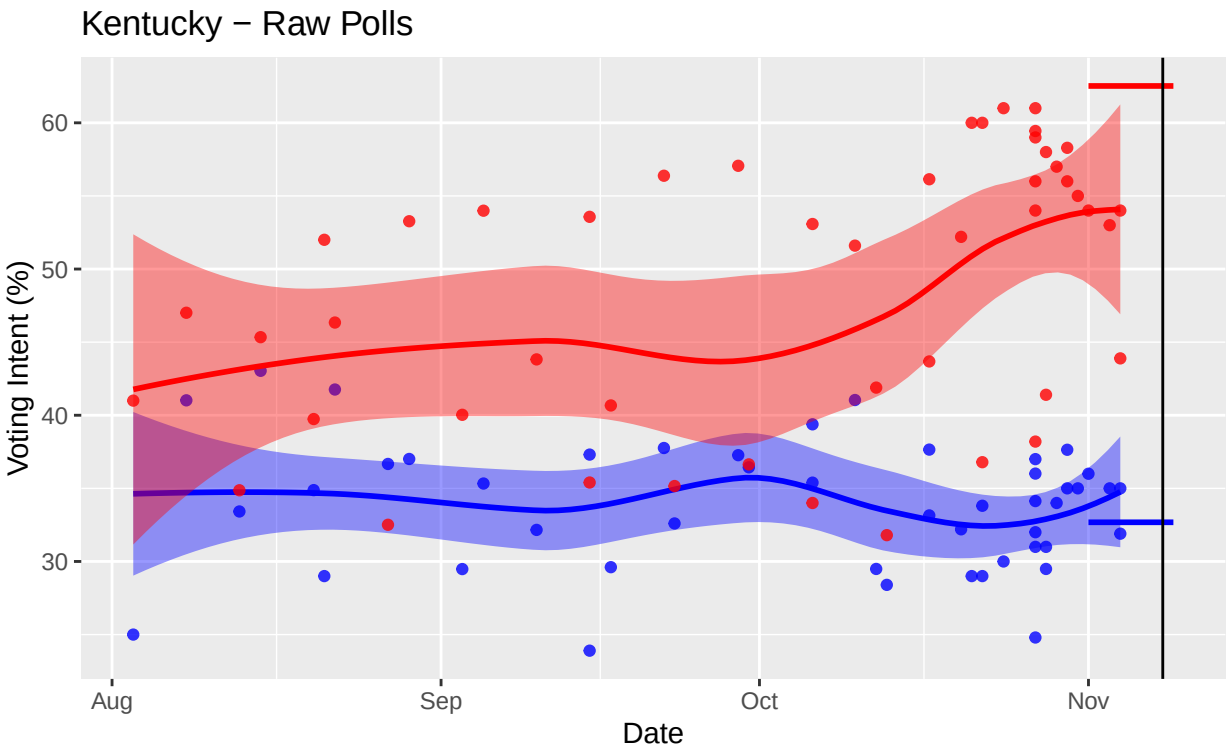

Indiana − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
Aug
Sep
Oct
Nov
Date

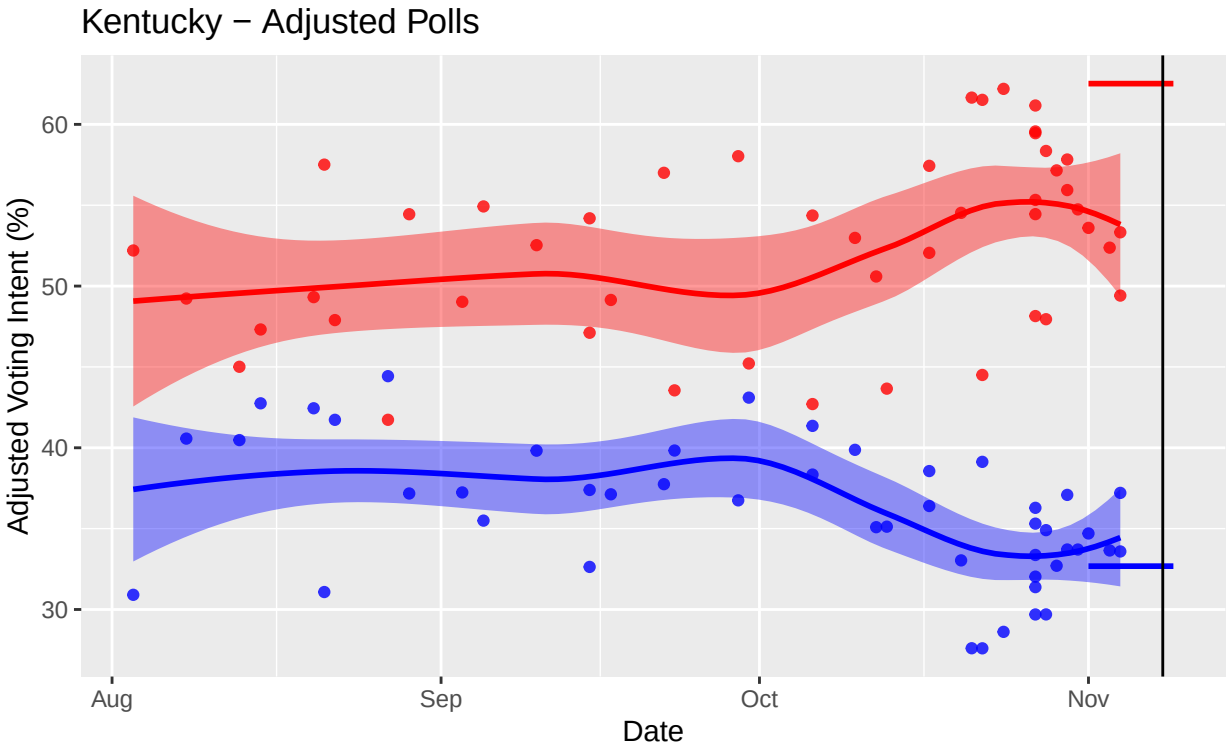

Iowa − Raw Polls
Voting Intent (%)
50
45
40
35
30
Aug
Sep
Oct
Nov
Date

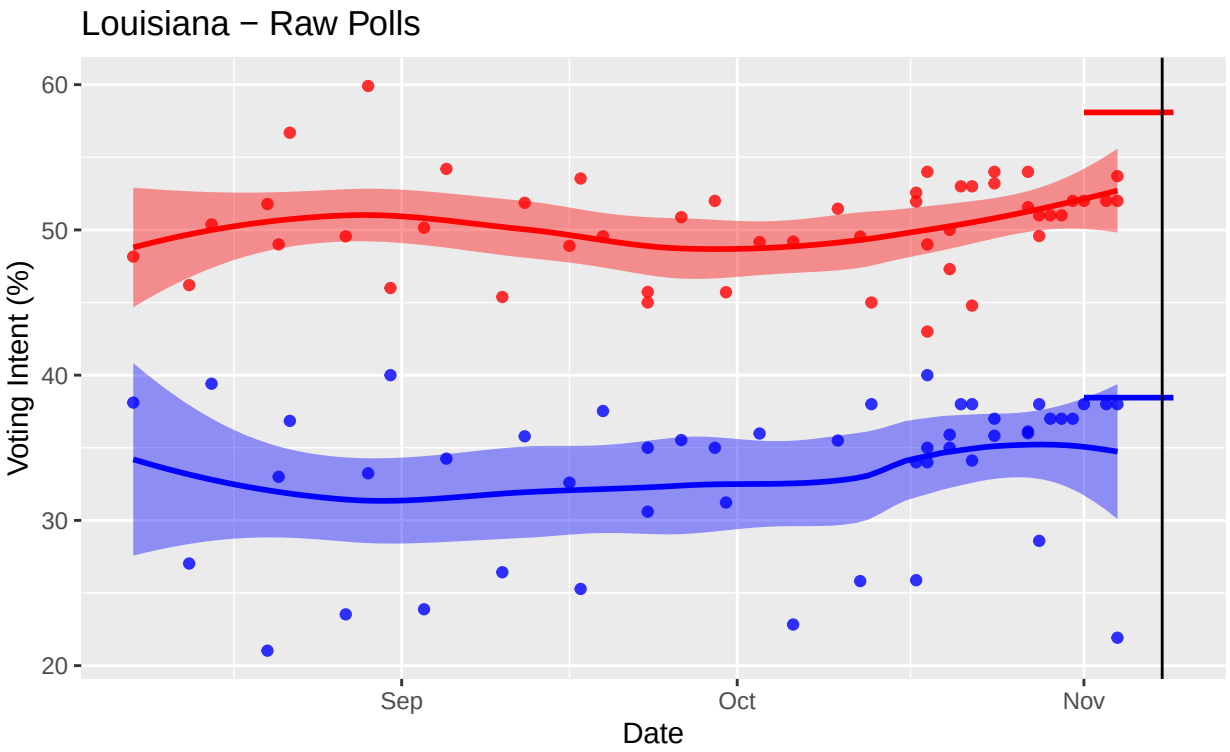

Iowa − Adjusted Polls
Adjusted Voting Intent (%)
50
45
40
35
Aug
Sep
Oct
Nov
Date

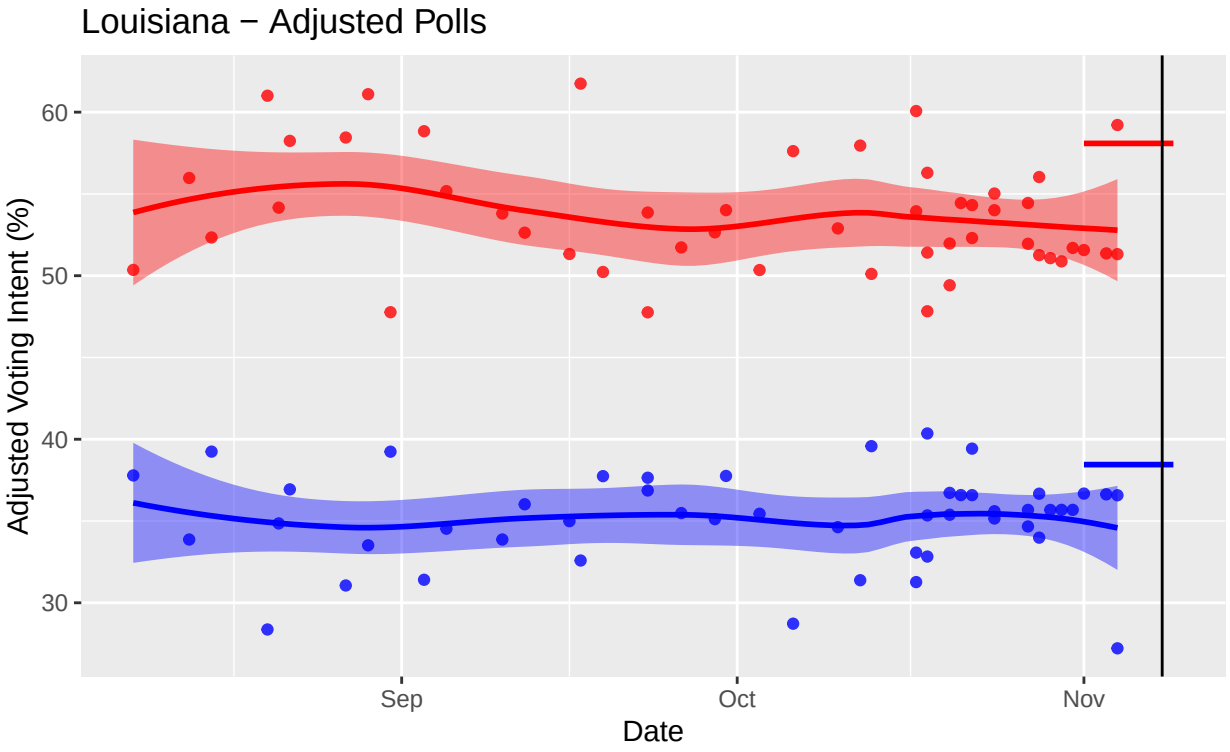

Kansas − Raw Polls
Voting Intent (%)
50
40
30
Aug
Sep
Oct
Nov
Date

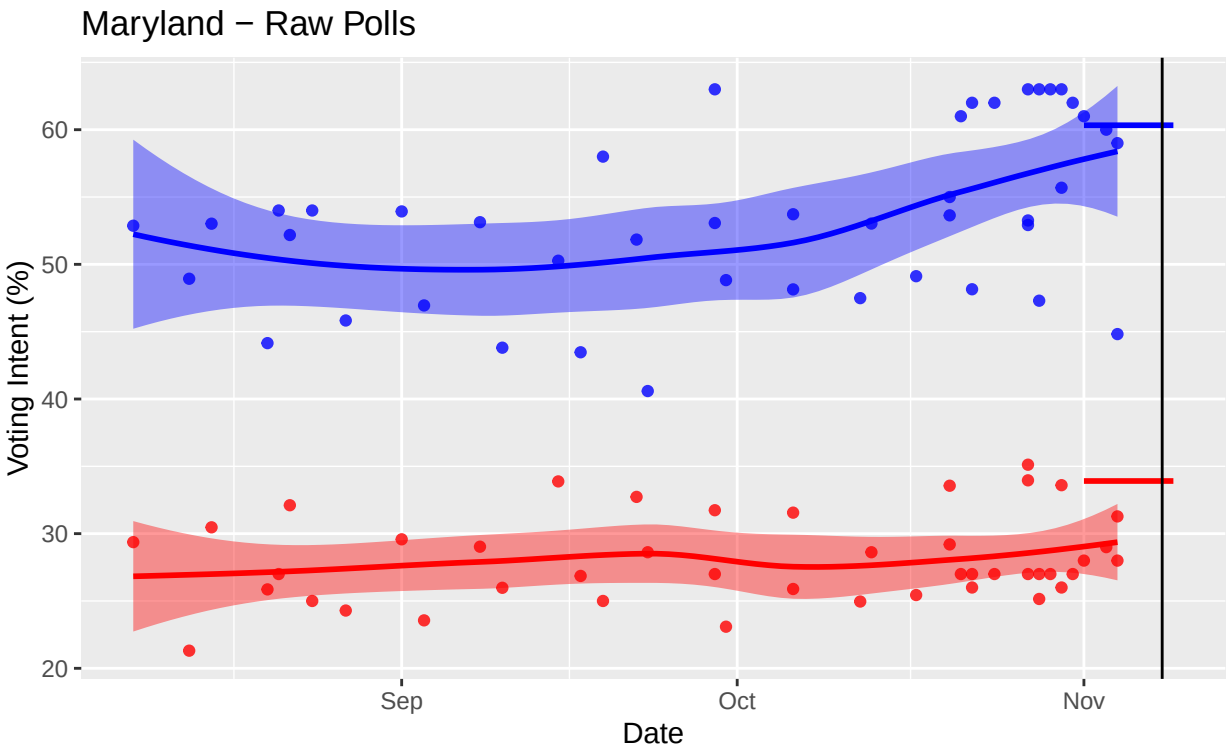

Kansas − Adjusted Polls
Adjusted Voting Intent (%)
50
40
Aug
Sep
Oct
Nov
Date

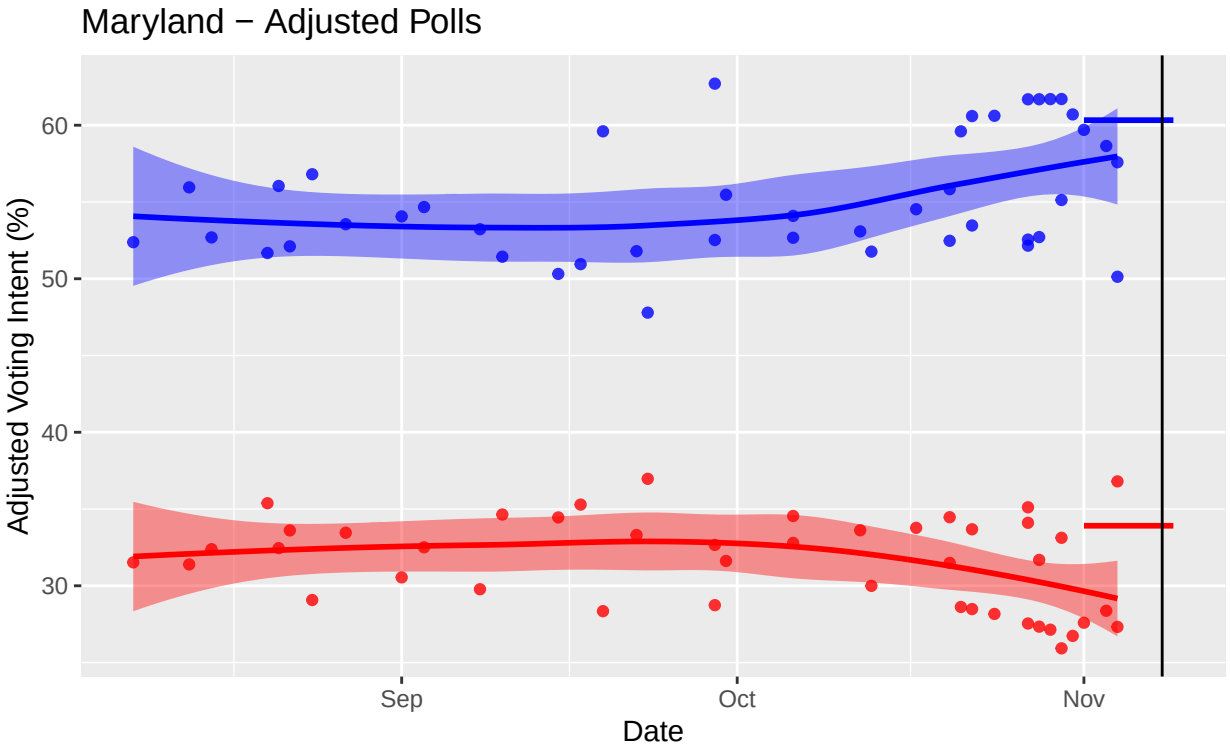

Kentucky − Raw Polls
Voting Intent (%)
60
50
40
30
Aug
Sep
Oct
Nov
Date

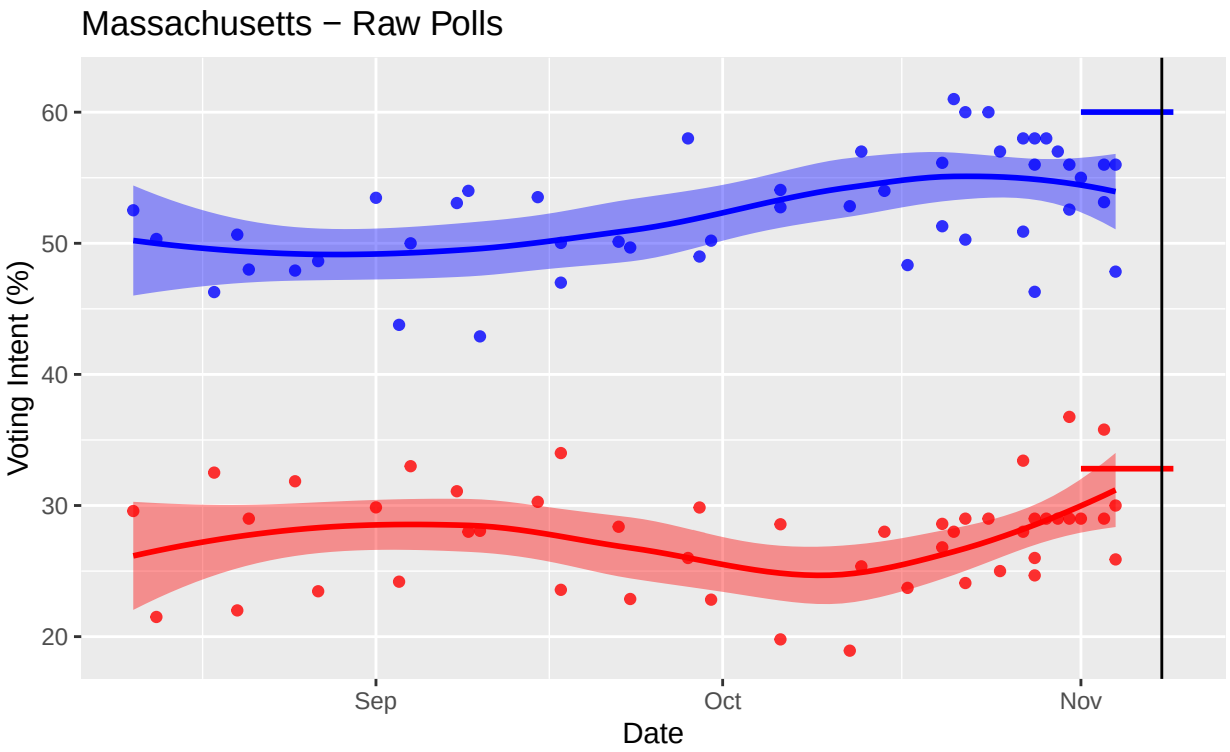

Kentucky − Adjusted Polls
Adjusted Voting Intent (%)
60
50
40
30
Aug
Sep
Oct
Nov
Date

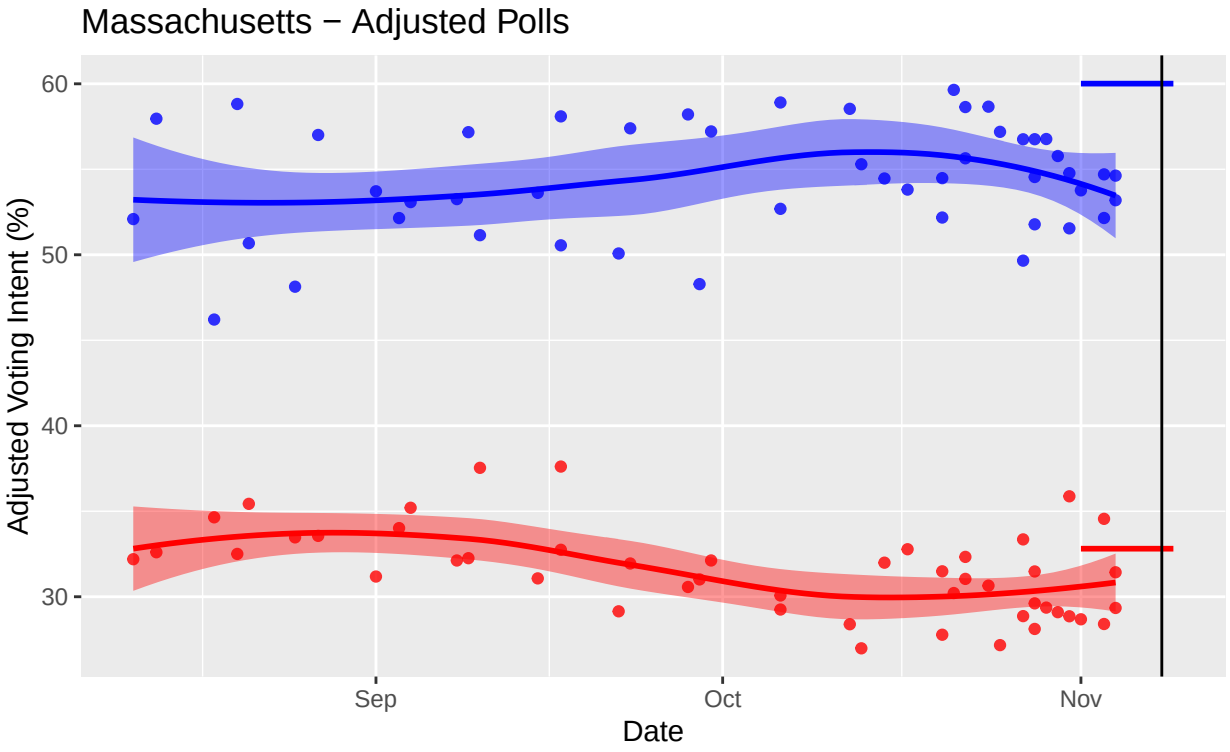

Louisiana − Raw Polls
Voting Intent (%)
60
50
40
30
20
Sep
Oct
Nov
Date

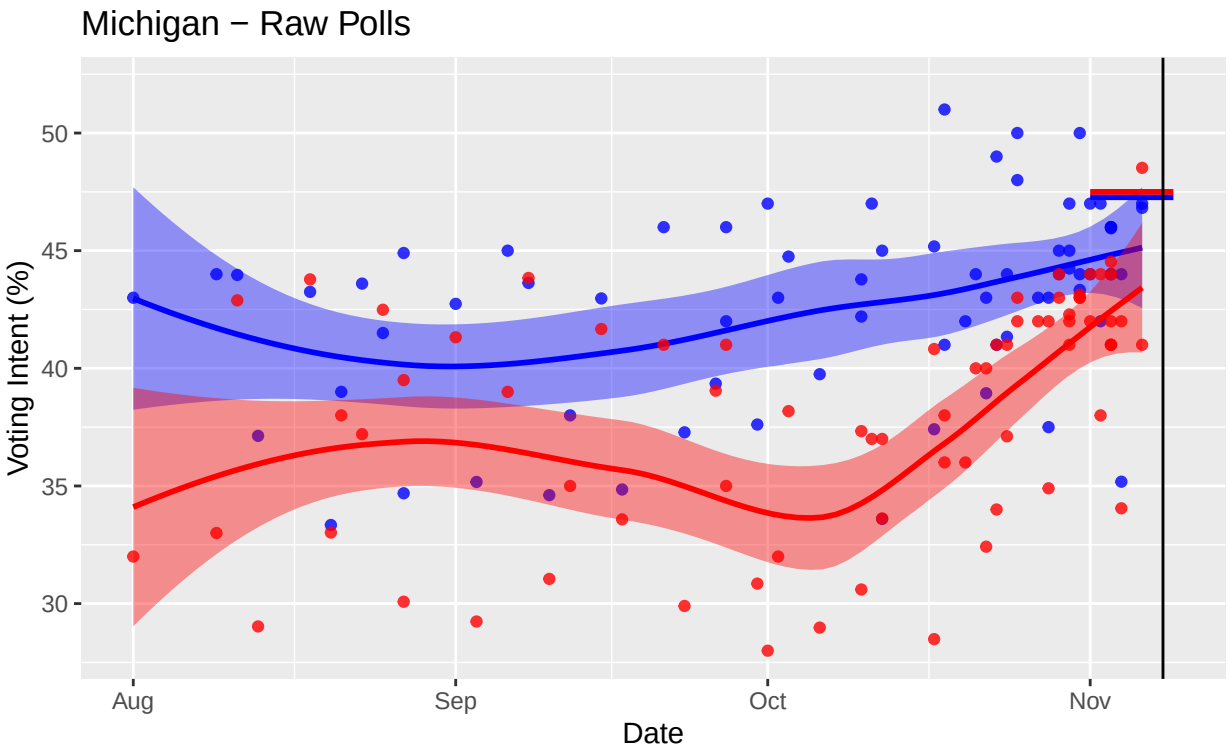

Louisiana − Adjusted Polls
Adjusted Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

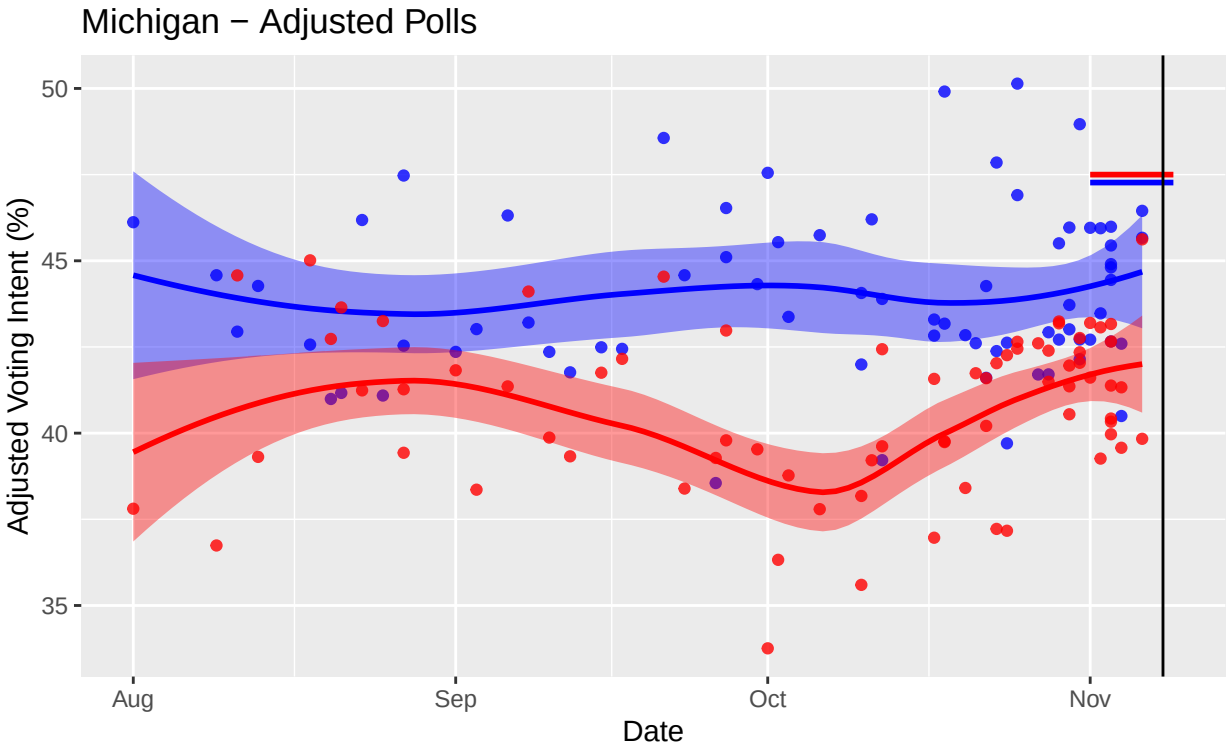

Maryland − Raw Polls
Voting Intent (%)
60
50
40
30
20
Sep
Oct
Nov
Date

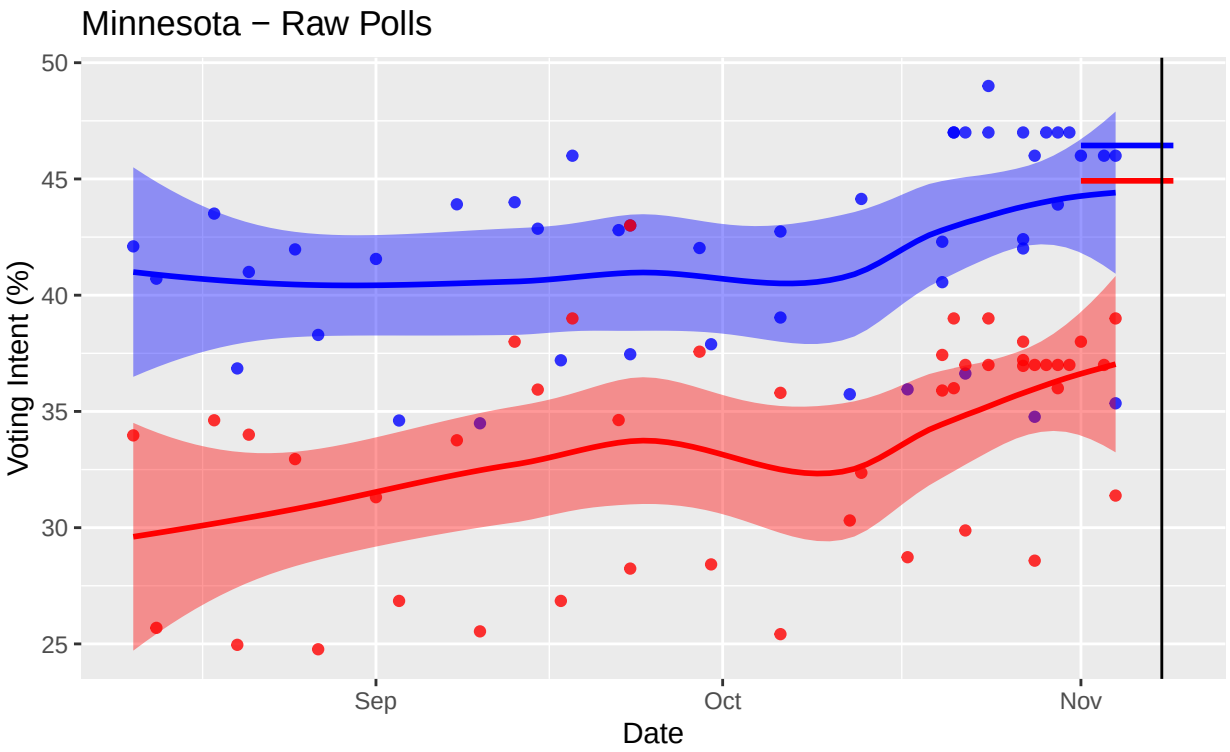

Maryland − Adjusted Polls
Adjusted Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

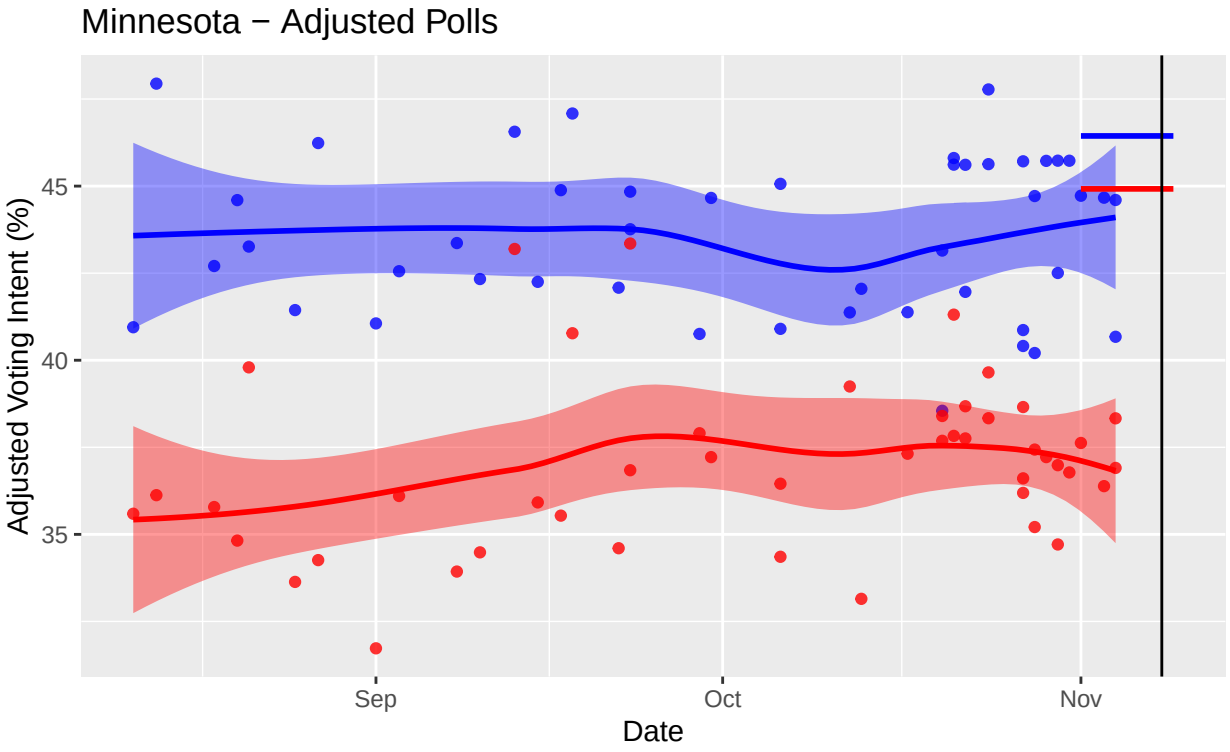

Massachusetts − Raw Polls
Voting Intent (%)
60
50
40
30
20
Sep
Oct
Nov
Date

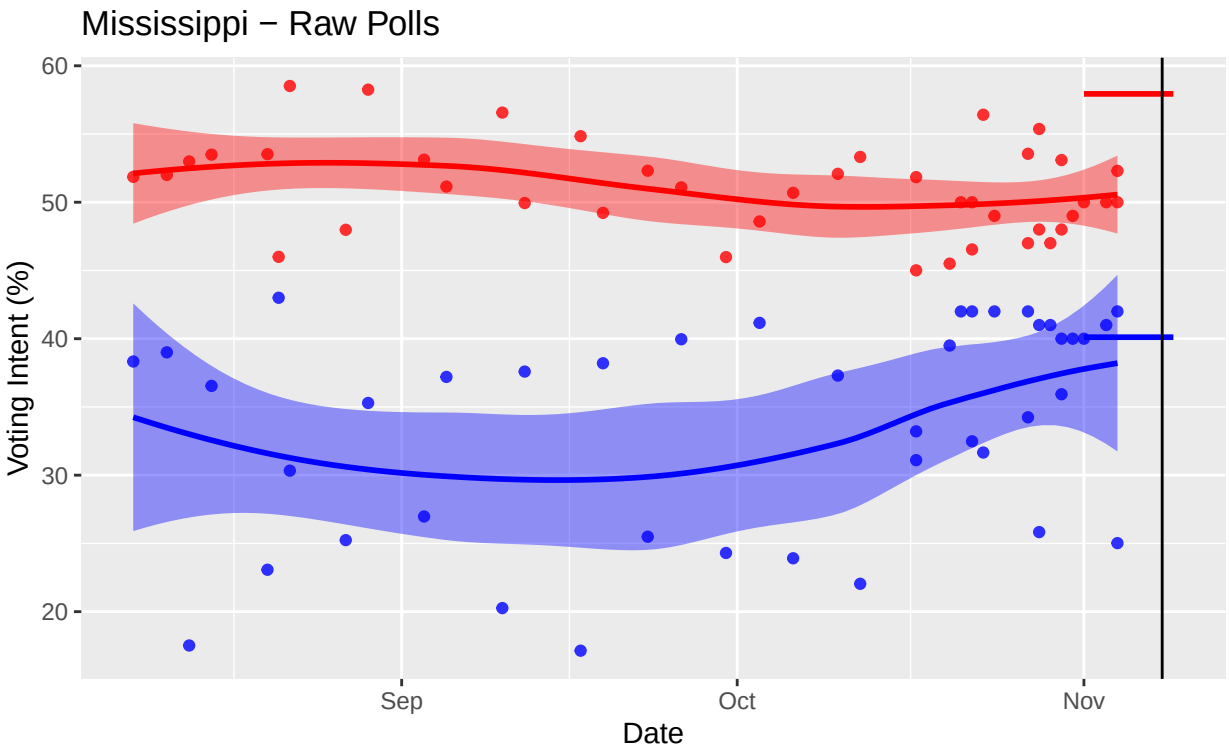

Massachusetts − Adjusted Polls
Adjusted Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

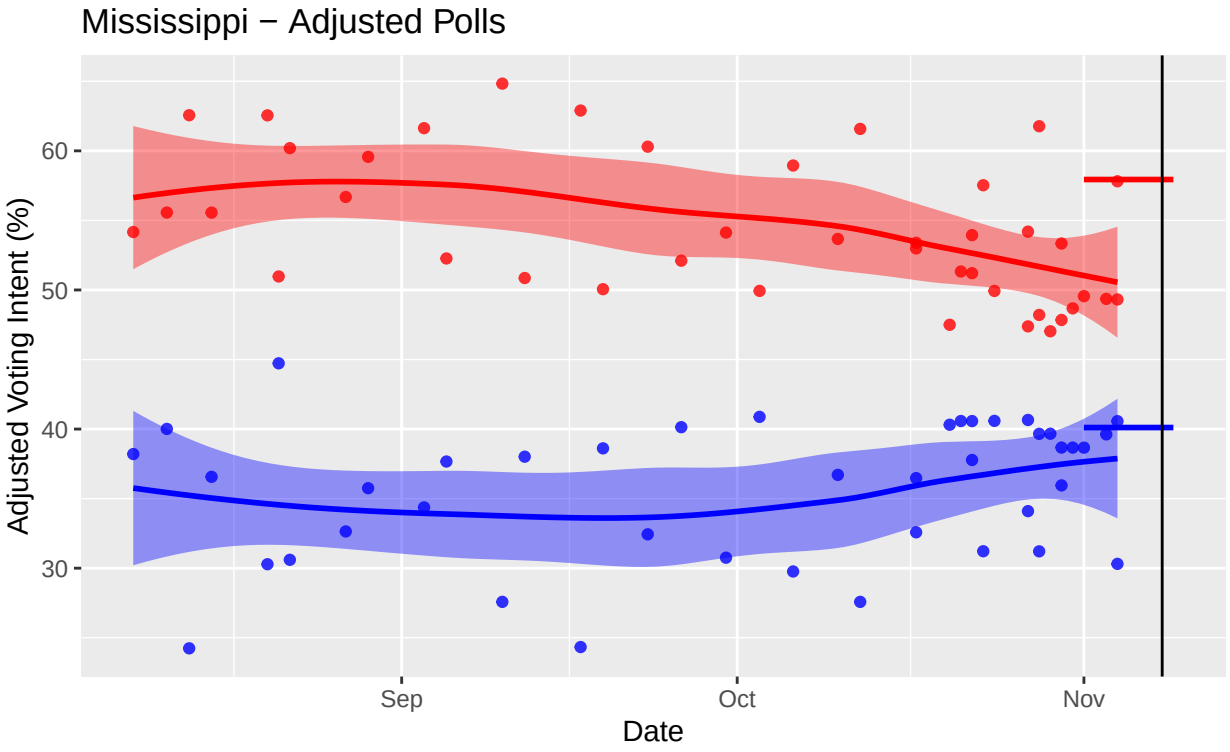

Michigan − Raw Polls
Voting Intent (%)
50
45
40
35
30
Aug
Sep
Oct
Nov
Date

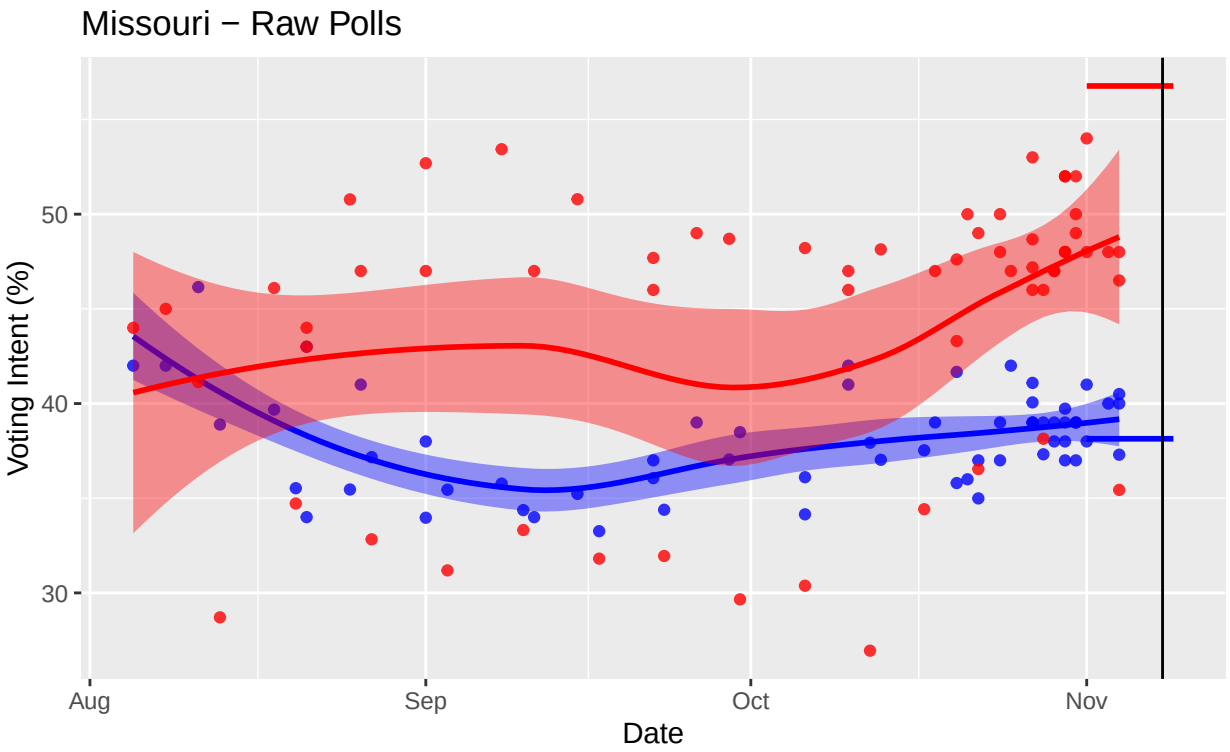

Michigan − Adjusted Polls
Adjusted Voting Intent (%)
50
45
40
35
Aug
Sep
Oct
Nov
Date

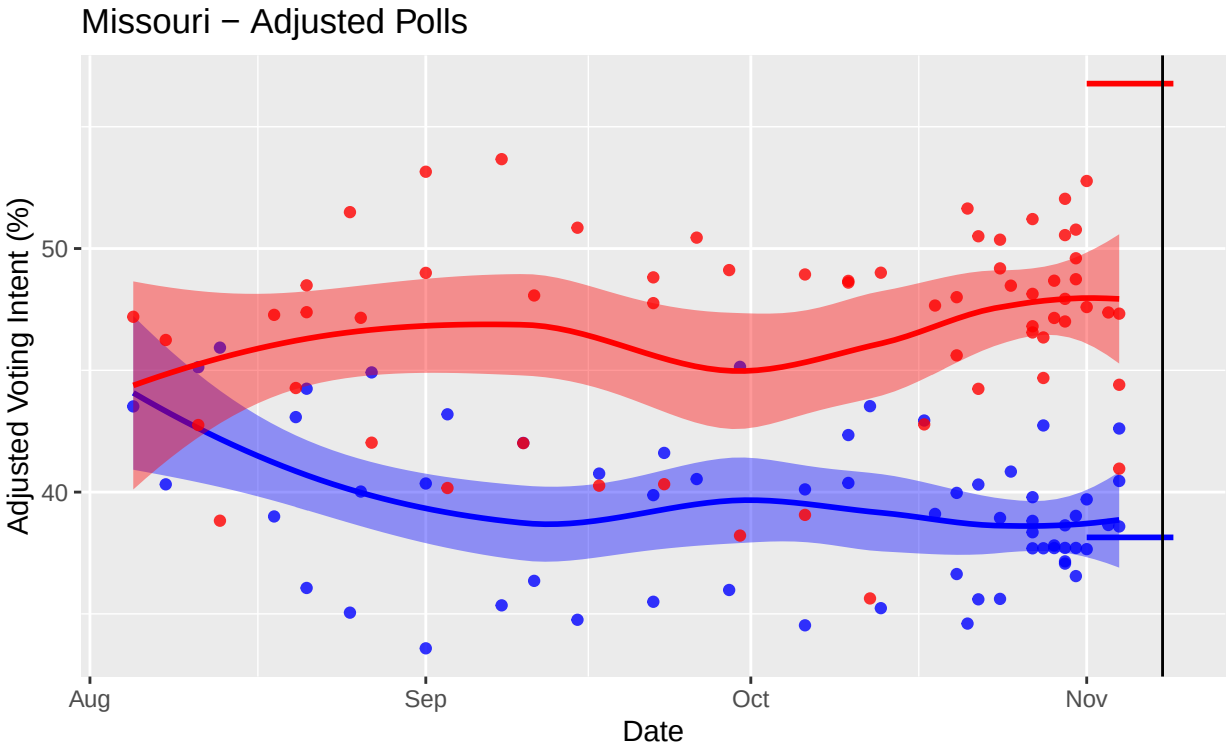

Minnesota − Raw Polls
Voting Intent (%)
50
45
40
35
30
25
Sep
Oct
Nov
Date

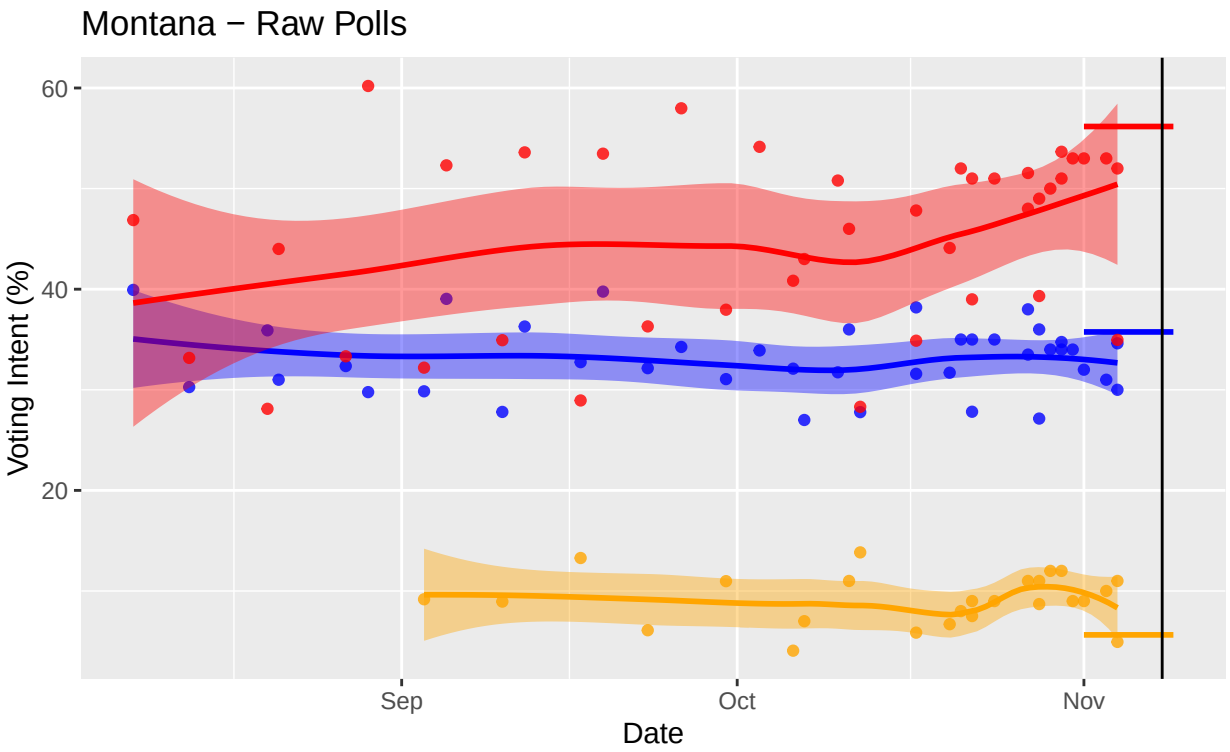

Minnesota − Adjusted Polls
Adjusted Voting Intent (%)
45
40
35
Sep
Oct
Nov
Date

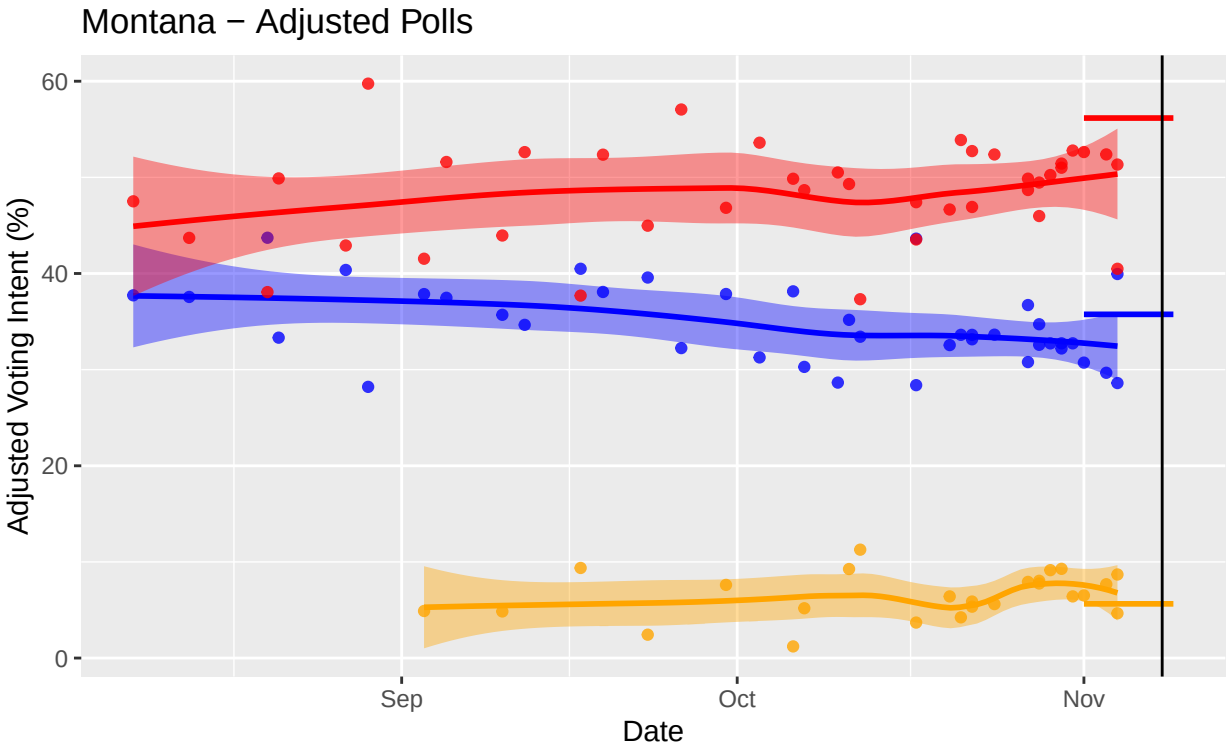

Mississippi − Raw Polls
Voting Intent (%)
60
50
40
30
20
Sep
Oct
Nov
Date

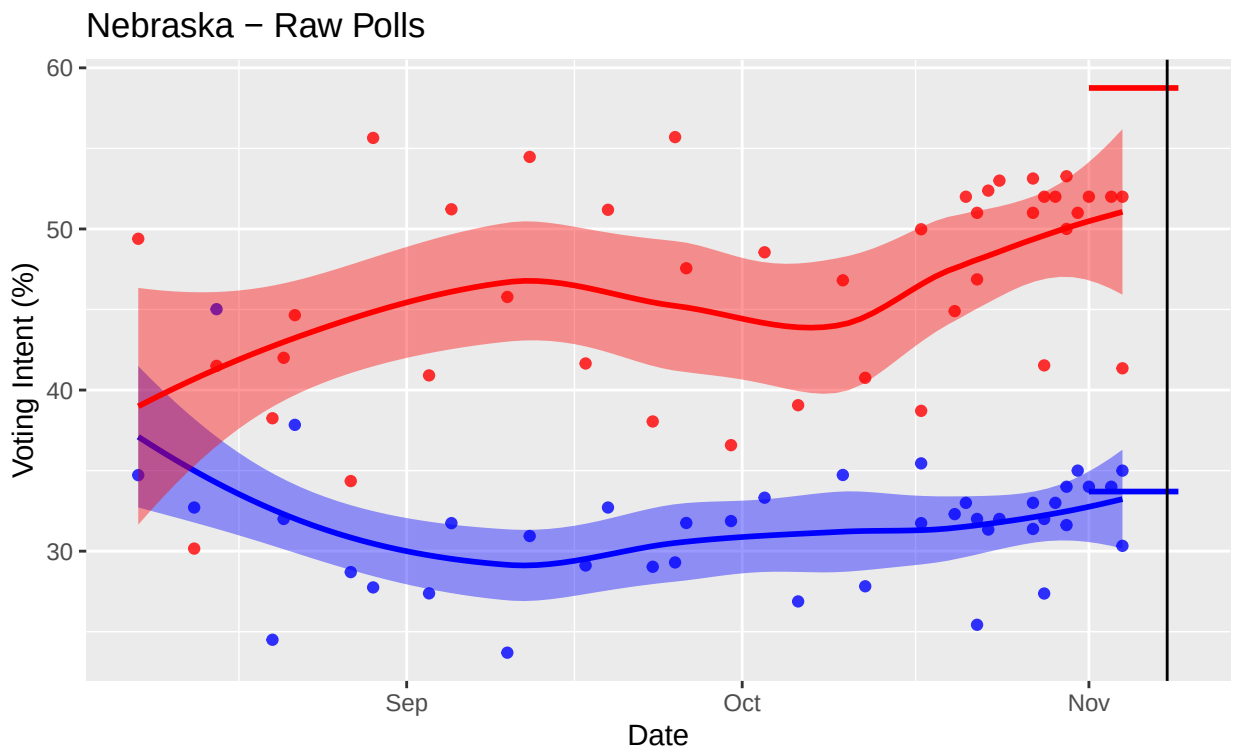

Mississippi − Adjusted Polls
Adjusted Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date


Missouri − Raw Polls
Voting Intent (%)
50
40
30
Aug
Sep
Oct
Nov
Date

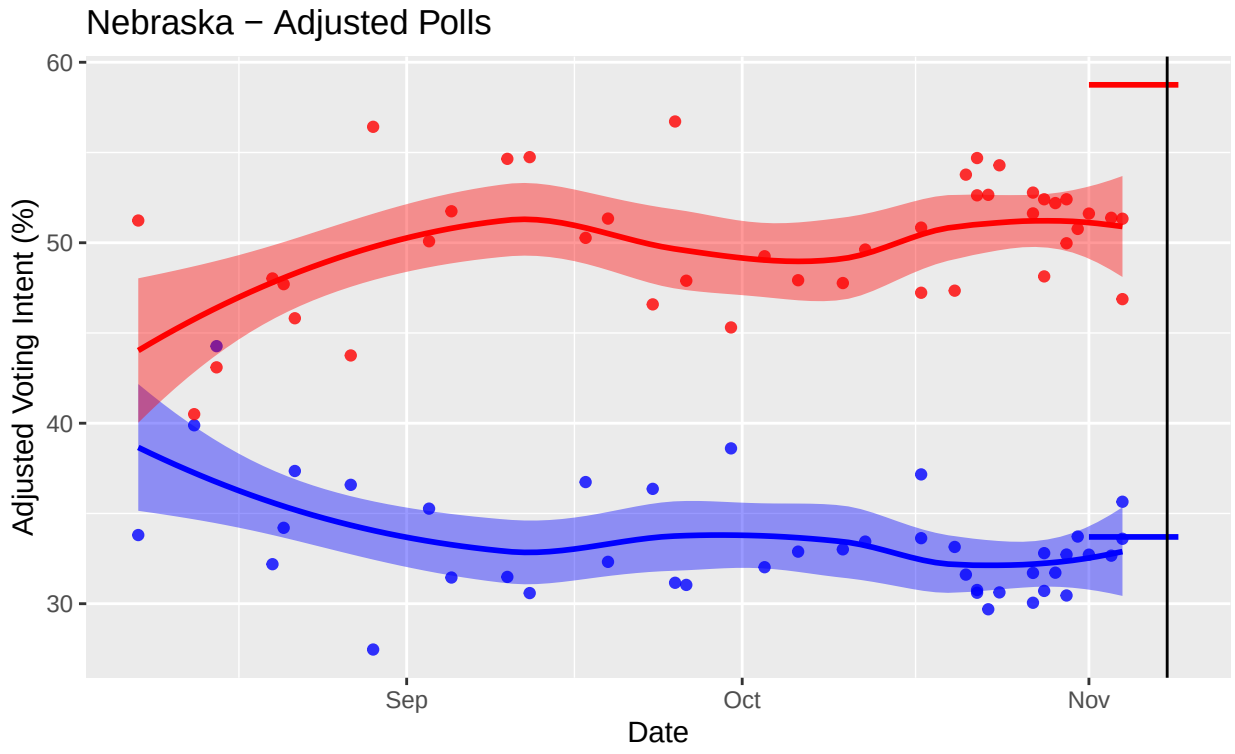

Missouri − Adjusted Polls
Adjusted Voting Intent (%)
50
40
Aug
Sep
Oct
Nov
Date


Montana − Raw Polls
60
40
20
Voting Intent (%)
Sep
Oct
Nov
Date


Montana − Adjusted Polls
60
40
20
0
Adjusted Voting Intent (%)
Sep
Oct
Nov
Date

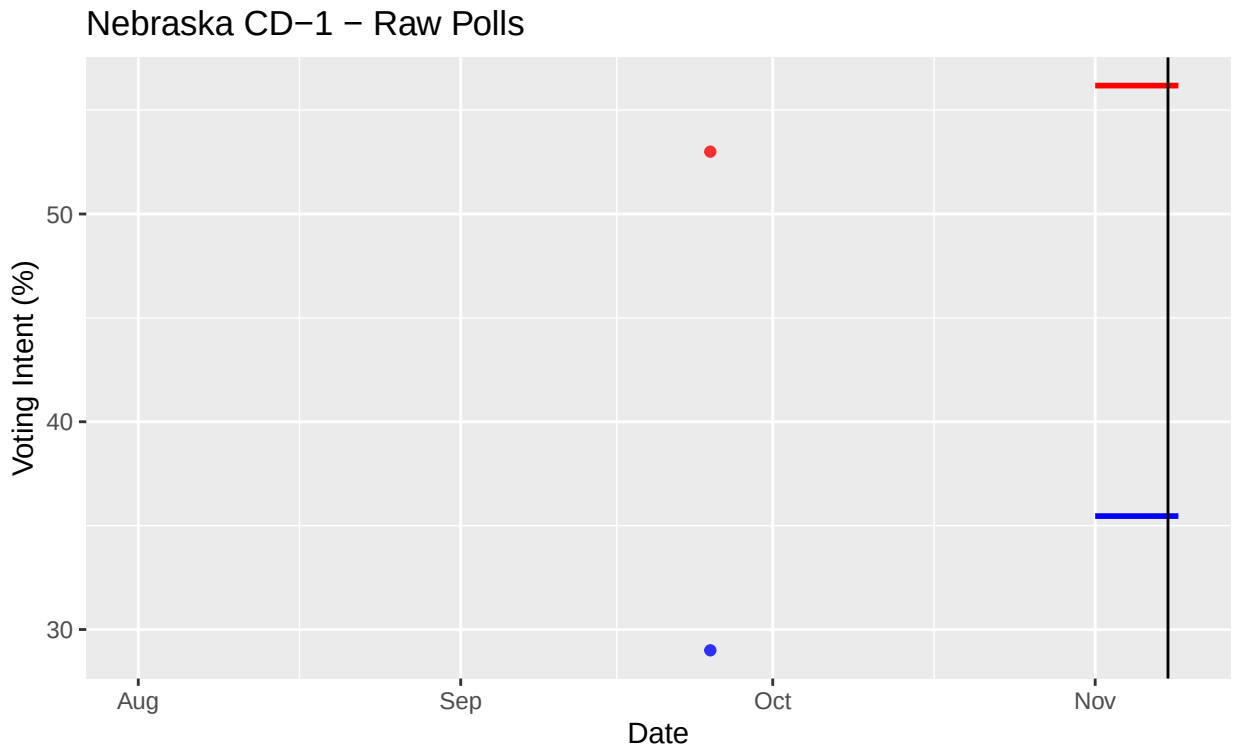


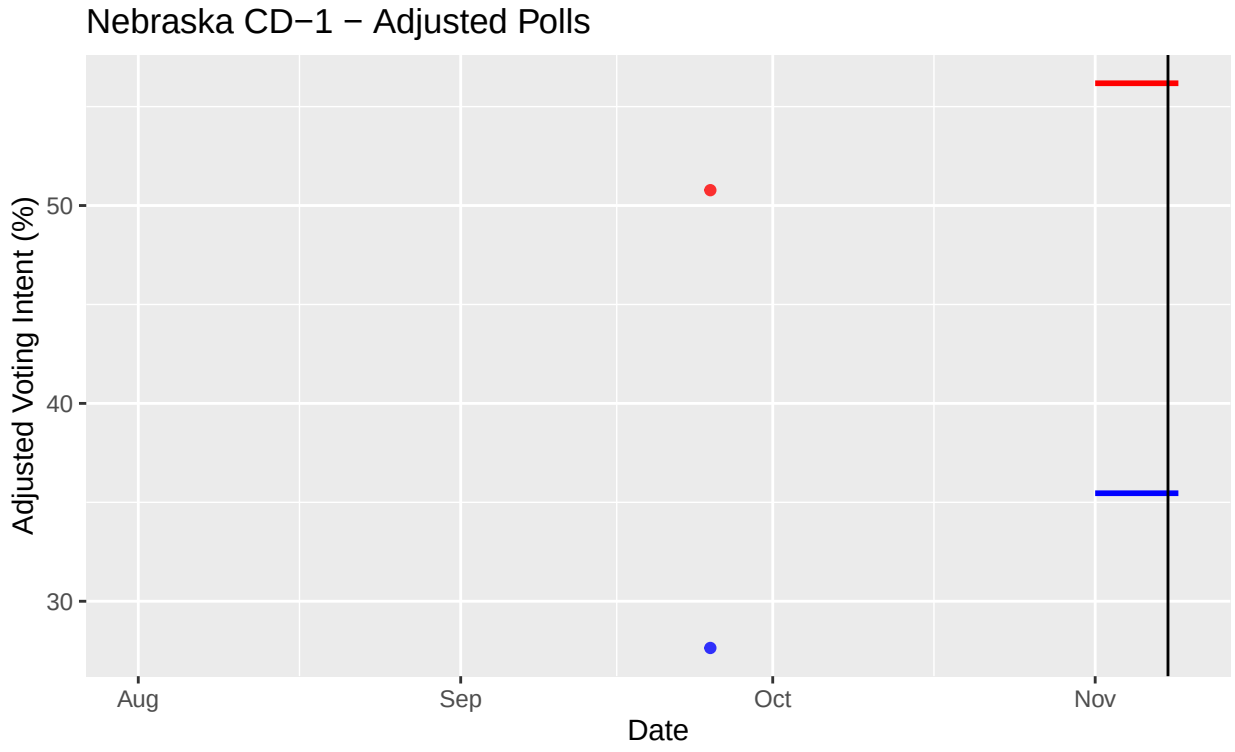

```
## Warning in max(statepolls$rawpolls, na.rm = TRUE): no non-missing arguments to
## max; returning -Inf
```




```
## Warning in max(statepolls$rawpolls, na.rm = TRUE): no non-missing arguments to
```

```
## max; returning -Inf
```

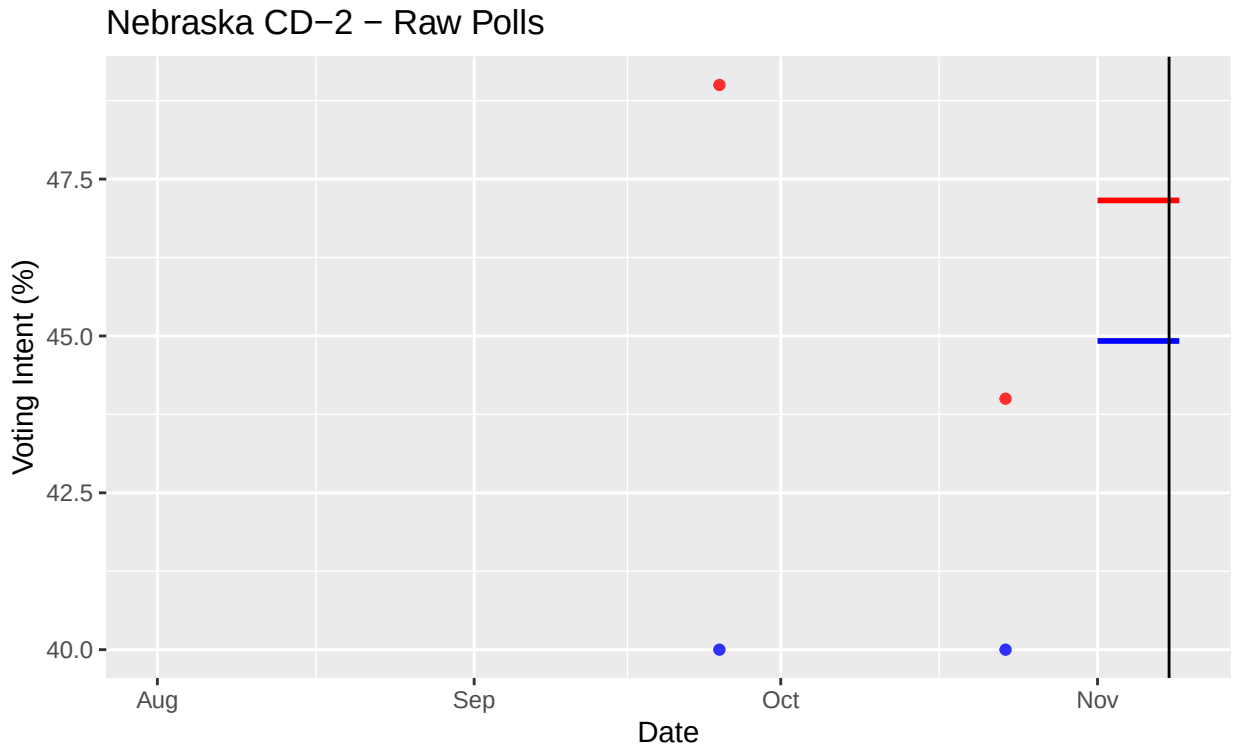





```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : span too small.  fewer data values than degrees of freedom.
```

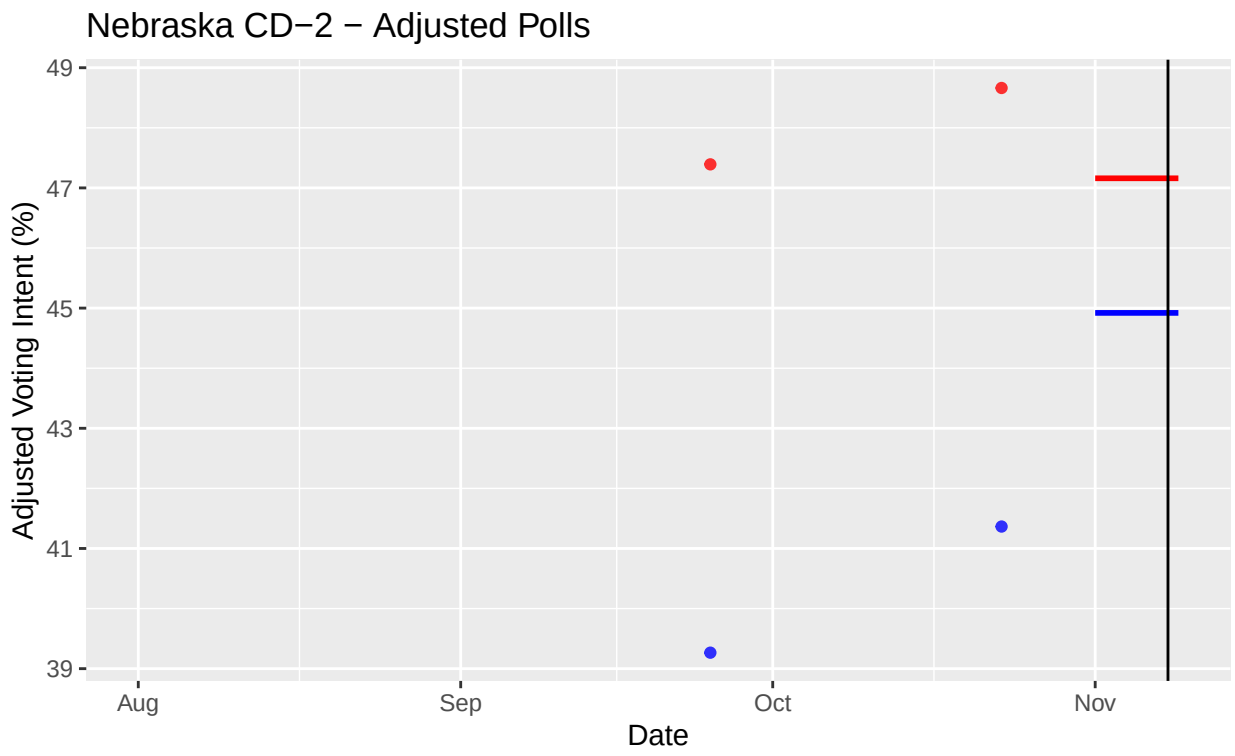
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : at 17069

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : radius 0.0196

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at 17069

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 0.14

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : at 17097

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : radius 0.0196

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : There are other near singularities as well. 0.0196

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : zero-width neighborhood. make span bigger

## Warning: Computation failed in `stat_smooth()`
## Caused by error in `predLoess()`:
## ! NA/NaN/Inf in foreign function call (arg 5)

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : span too small.  fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : at 17069

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : radius 0.0196

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : all data on boundary of neighborhood. make span bigger
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at 17069

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 0.14

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : at 17097

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : radius 0.0196

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : There are other near singularities as well. 0.0196

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : zero-width neighborhood. make span bigger

## Warning: Computation failed in `stat_smooth()`
## Caused by error in `predLoess()`:
## ! NA/NaN/Inf in foreign function call (arg 5)
```

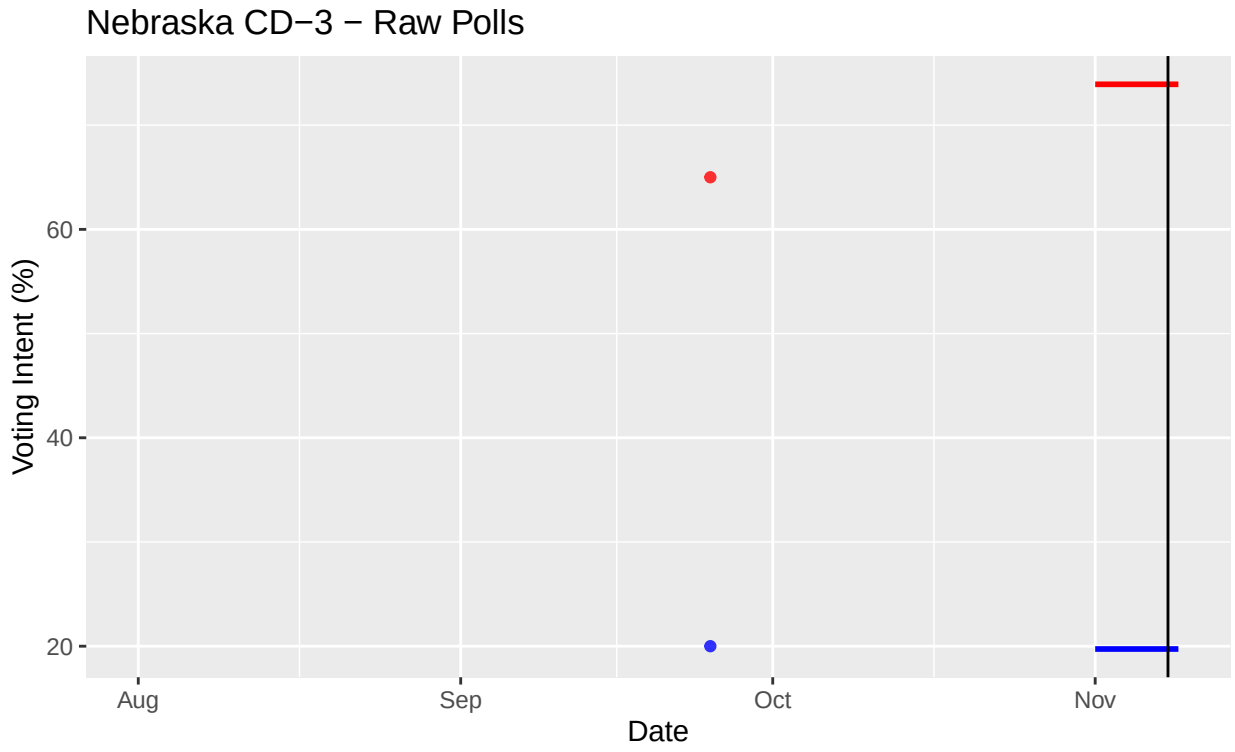


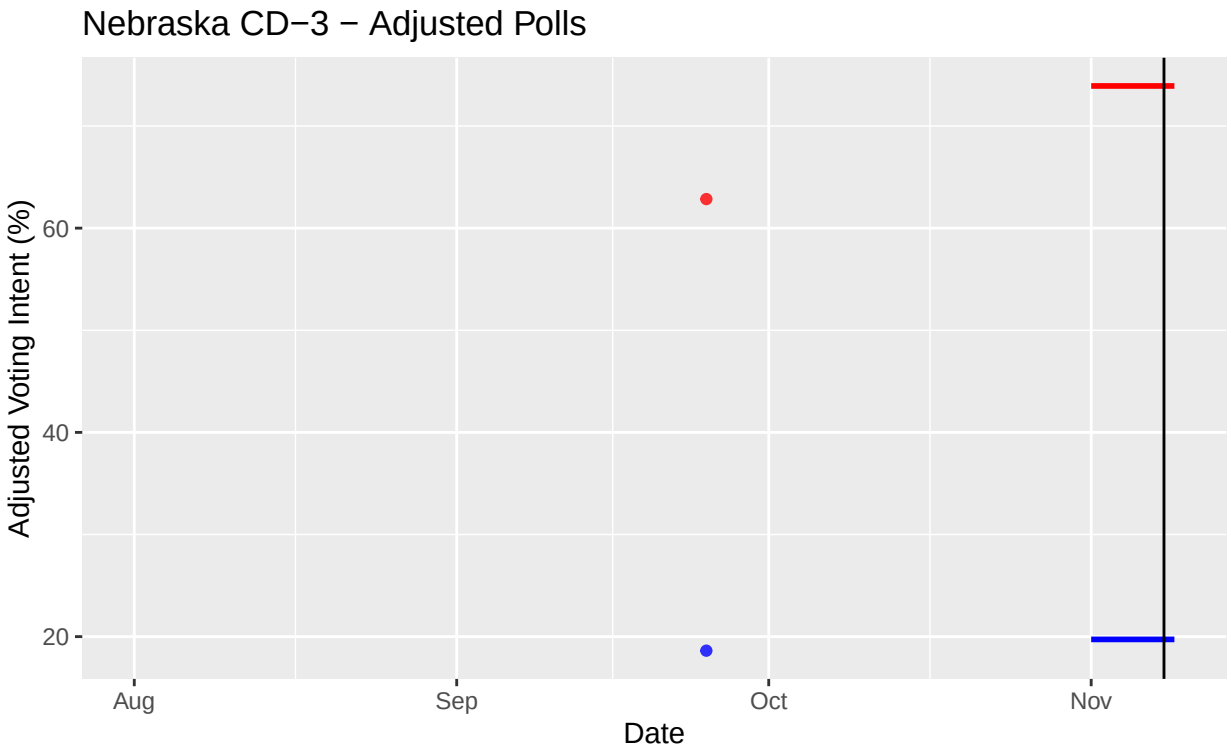
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : span too small.  fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : at 17069

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : radius 0.0196

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at 17069

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 0.14

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : at 17097

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : radius 0.0196

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : There are other near singularities as well. 0.0196

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : zero-width neighborhood. make span bigger

## Warning: Computation failed in `stat_smooth()`
## Caused by error in `predLoess()`:
## ! NA/NaN/Inf in foreign function call (arg 5)

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : span too small.  fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : at 17069

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : radius 0.0196
```

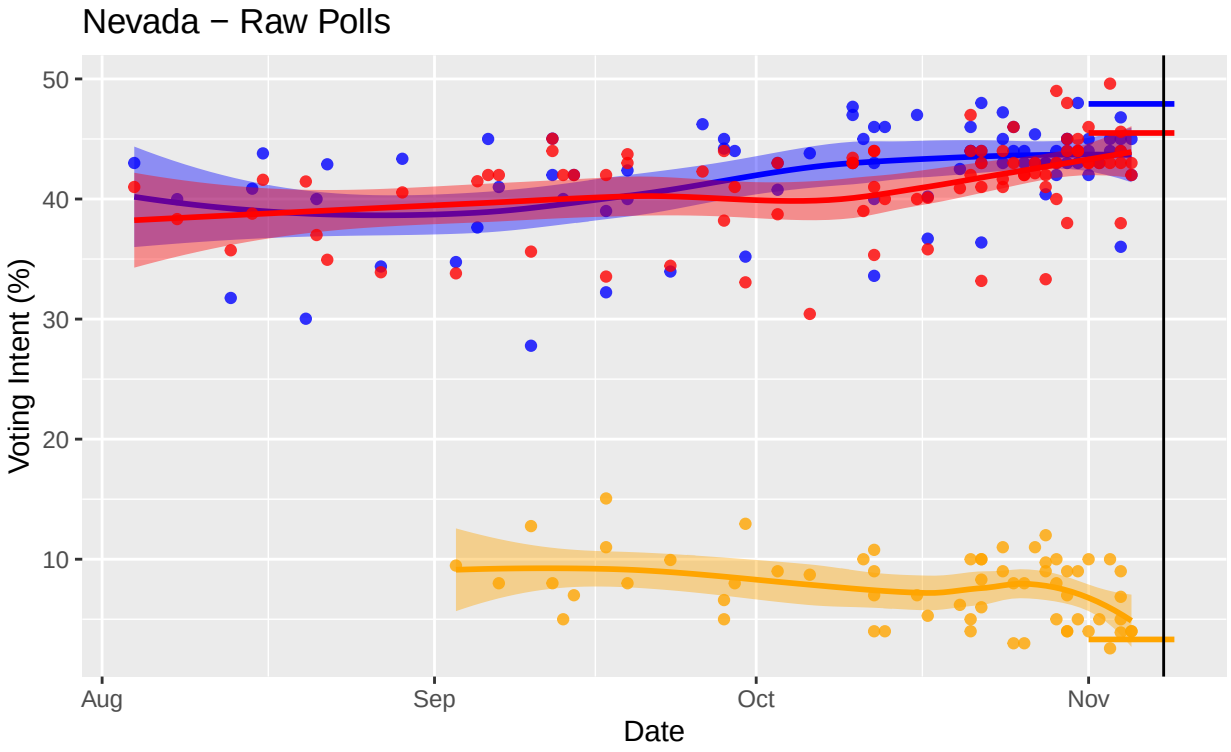
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at 17069

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 0.14

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : at 17097

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : radius 0.0196

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : all data on boundary of neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : There are other near singularities as well. 0.0196

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : zero-width neighborhood. make span bigger

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : zero-width neighborhood. make span bigger

## Warning: Computation failed in `stat_smooth()`
## Caused by error in `predLoess()`:
## ! NA/NaN/Inf in foreign function call (arg 5)

## Warning in max(statepolls$rawpolls, na.rm = TRUE): no non-missing arguments to
## max; returning -Inf
```

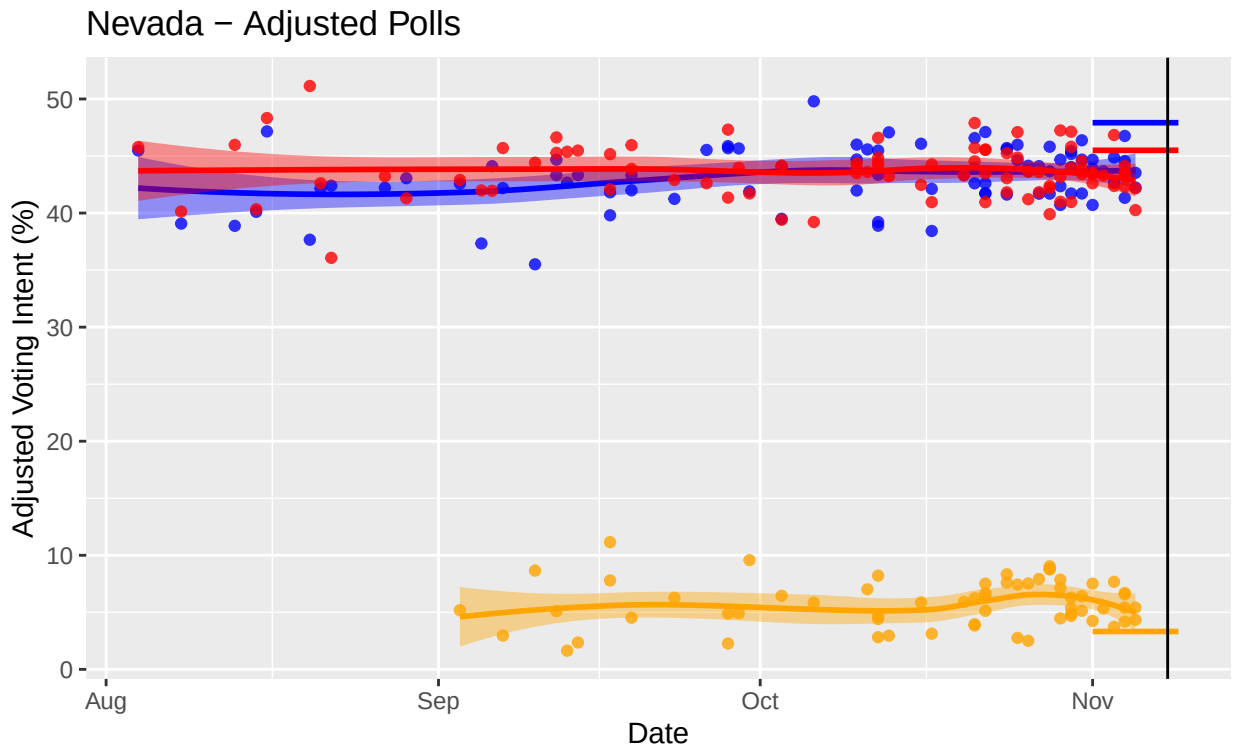


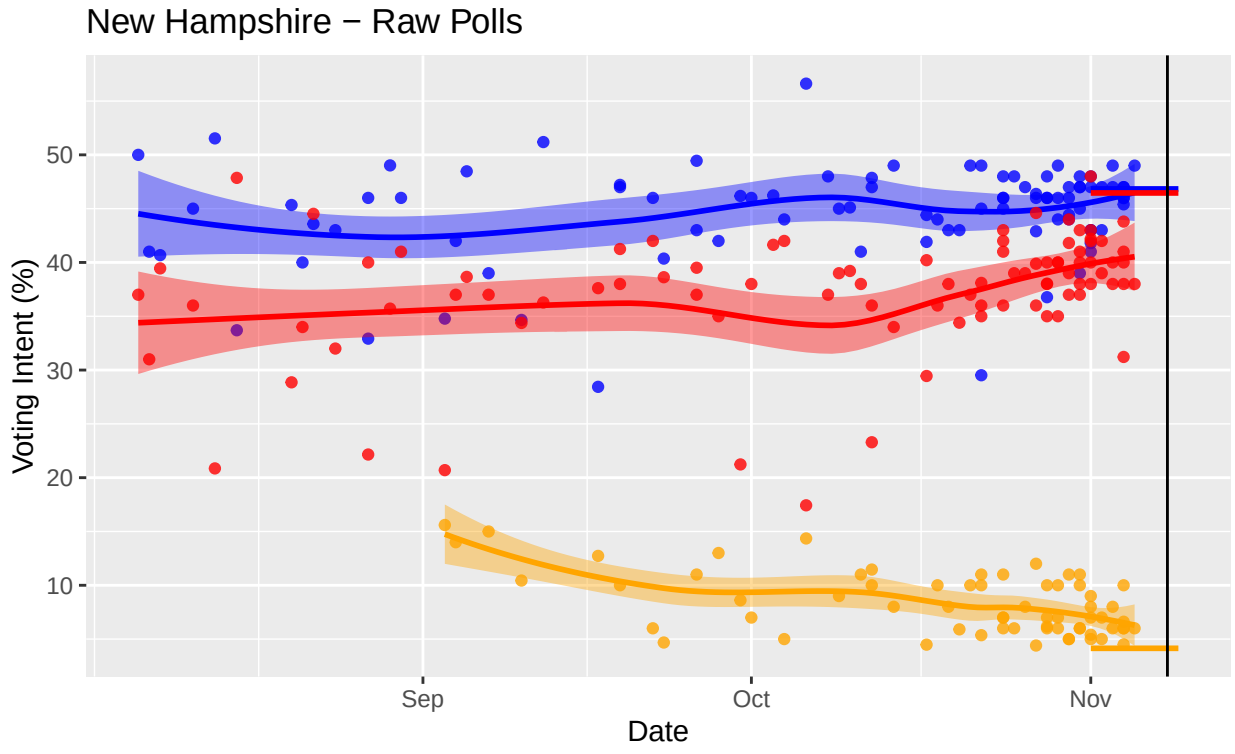

```
## Warning in max(statepolls$rawpolls, na.rm = TRUE): no non-missing arguments to
## max; returning -Inf
```

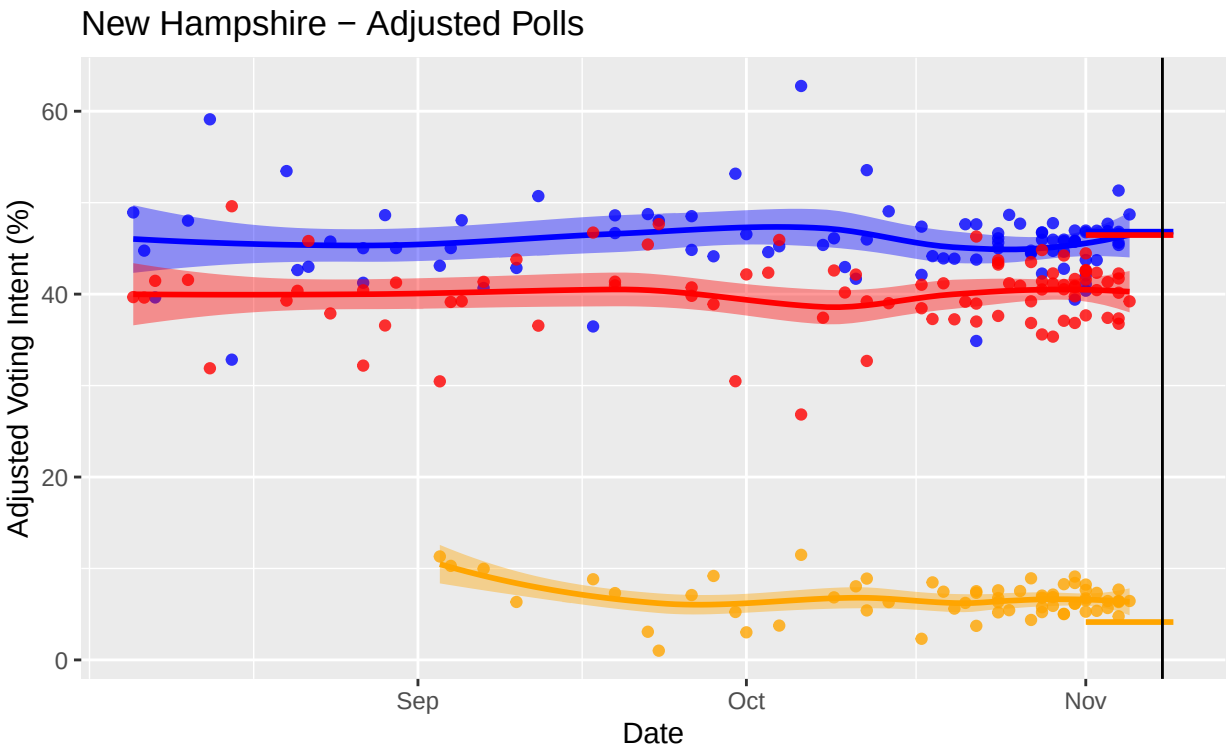


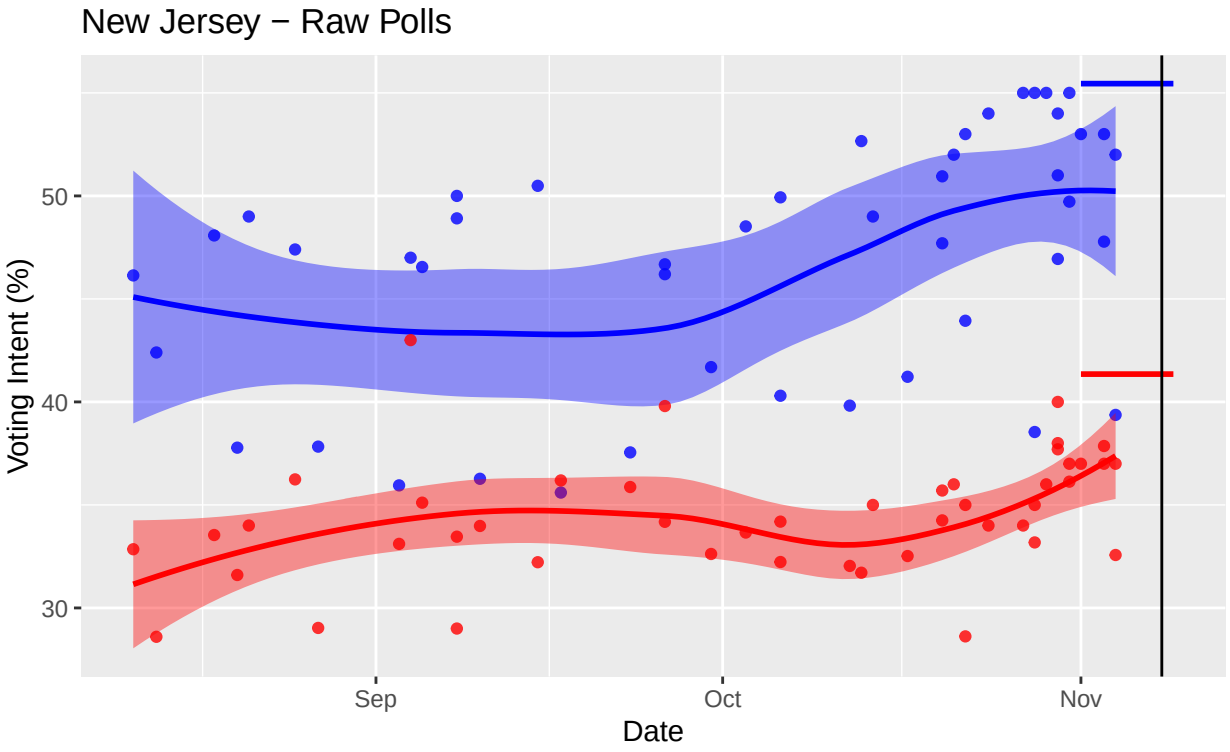

Nebraska CD−3 − Adjusted Polls
Adjusted Voting Intent (%)
60
40
20
Aug
Sep
Oct
Nov
Date

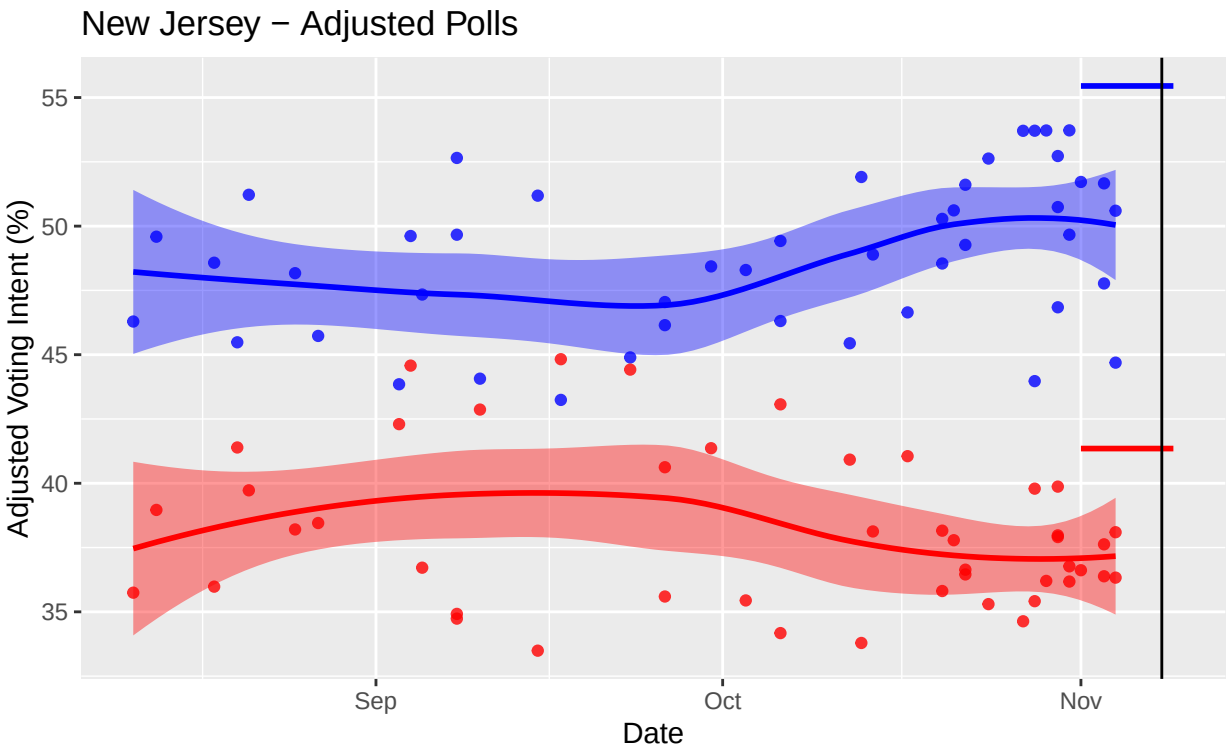

Nevada − Raw Polls
Voting Intent (%)
50
40
30
20
10
Aug
Sep
Oct
Nov
Date

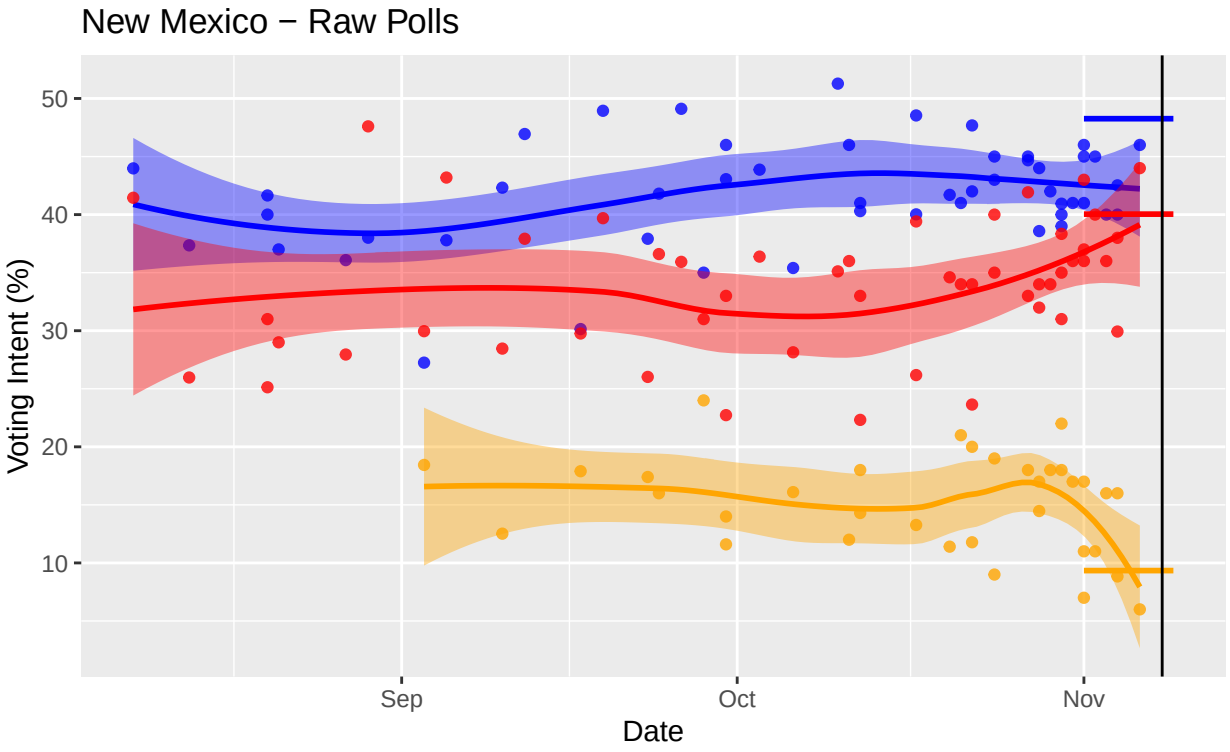

Nevada − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
20
10
0
Aug
Sep
Oct
Nov
Date

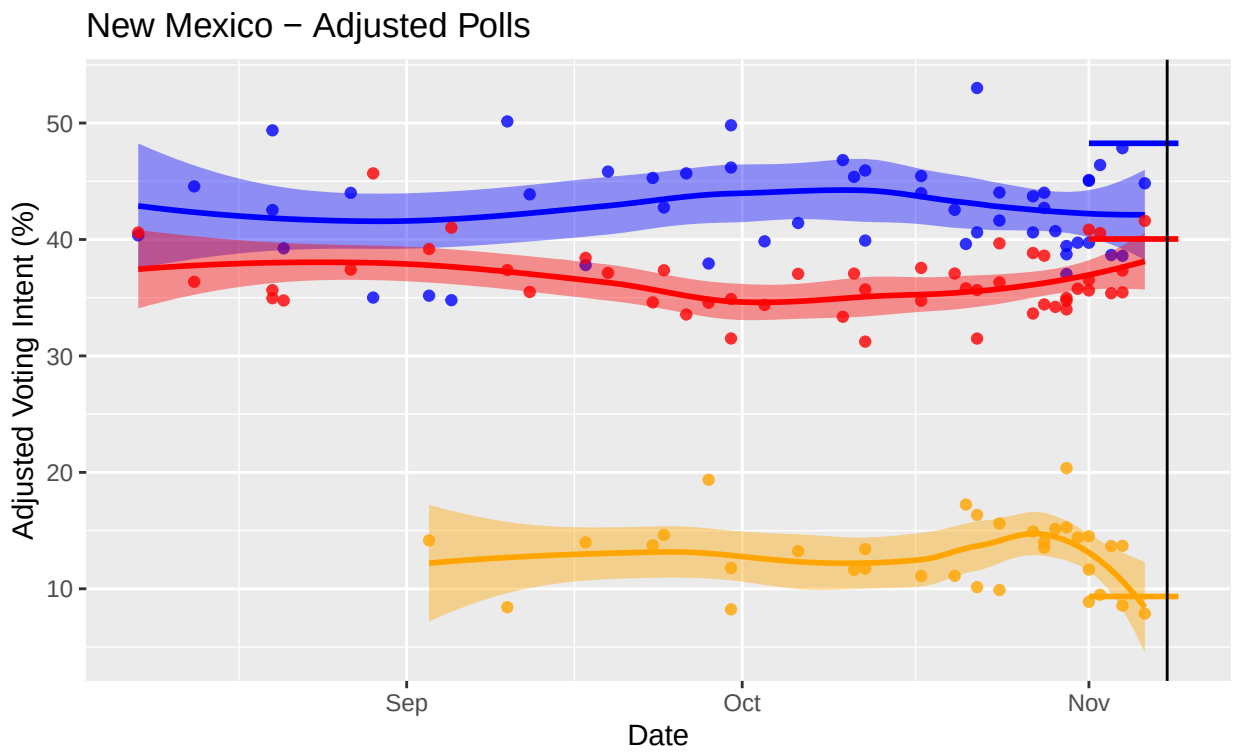

New Hampshire − Raw Polls
Voting Intent (%)
50
40
30
20
10
Sep
Oct
Nov
Date

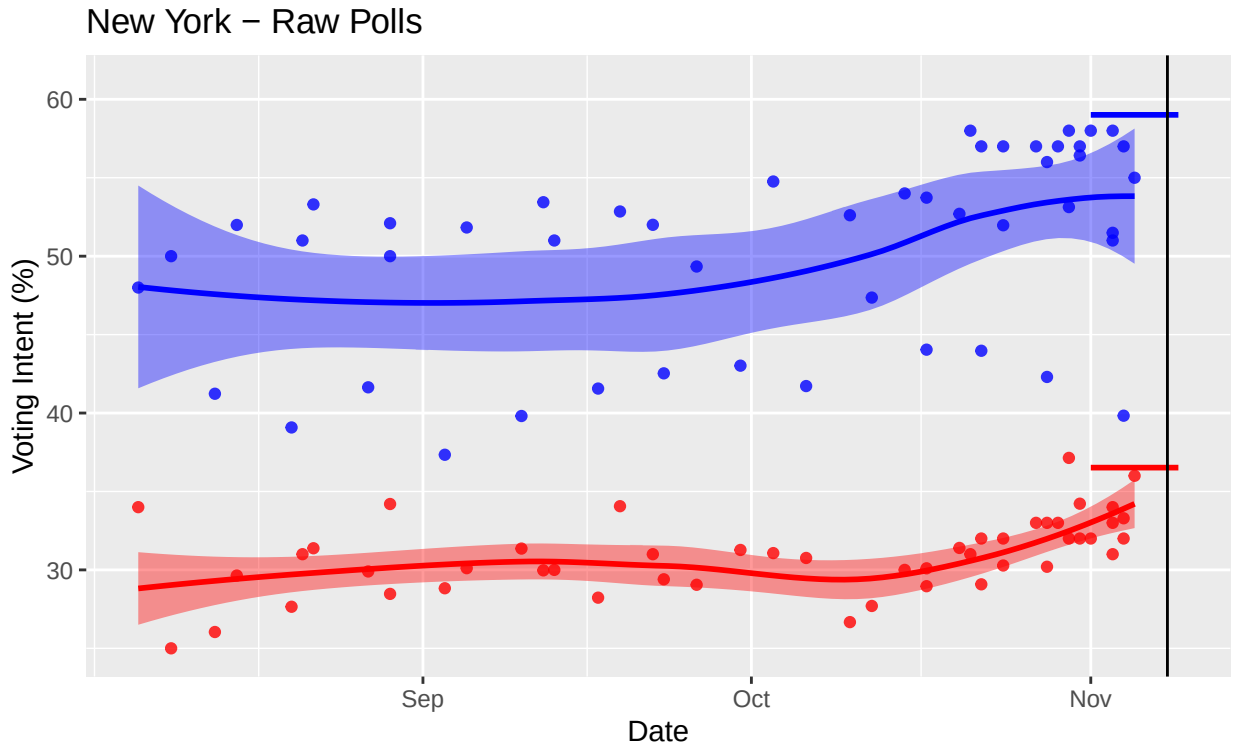

New Hampshire − Adjusted Polls
Adjusted Voting Intent (%)
60
40
20
0
Sep
Oct
Nov
Date

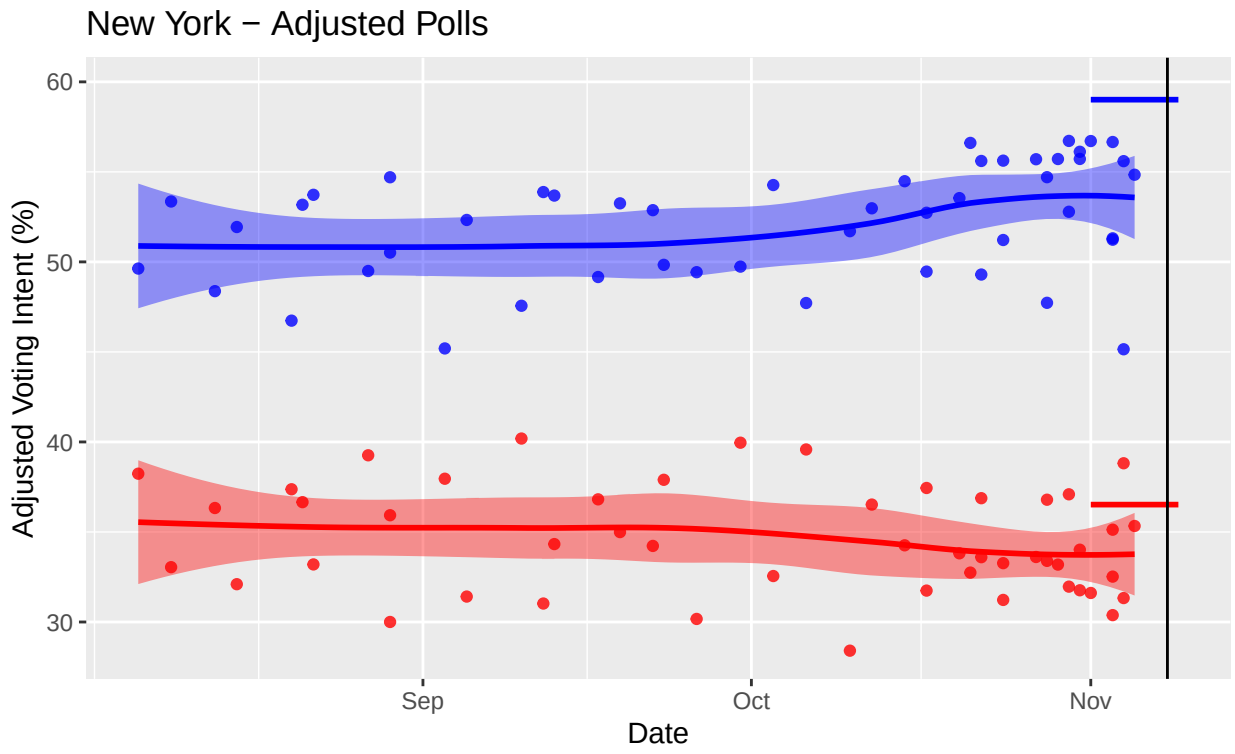

New Jersey − Raw Polls
Voting Intent (%)
50
40
30
Sep
Oct
Nov
Date

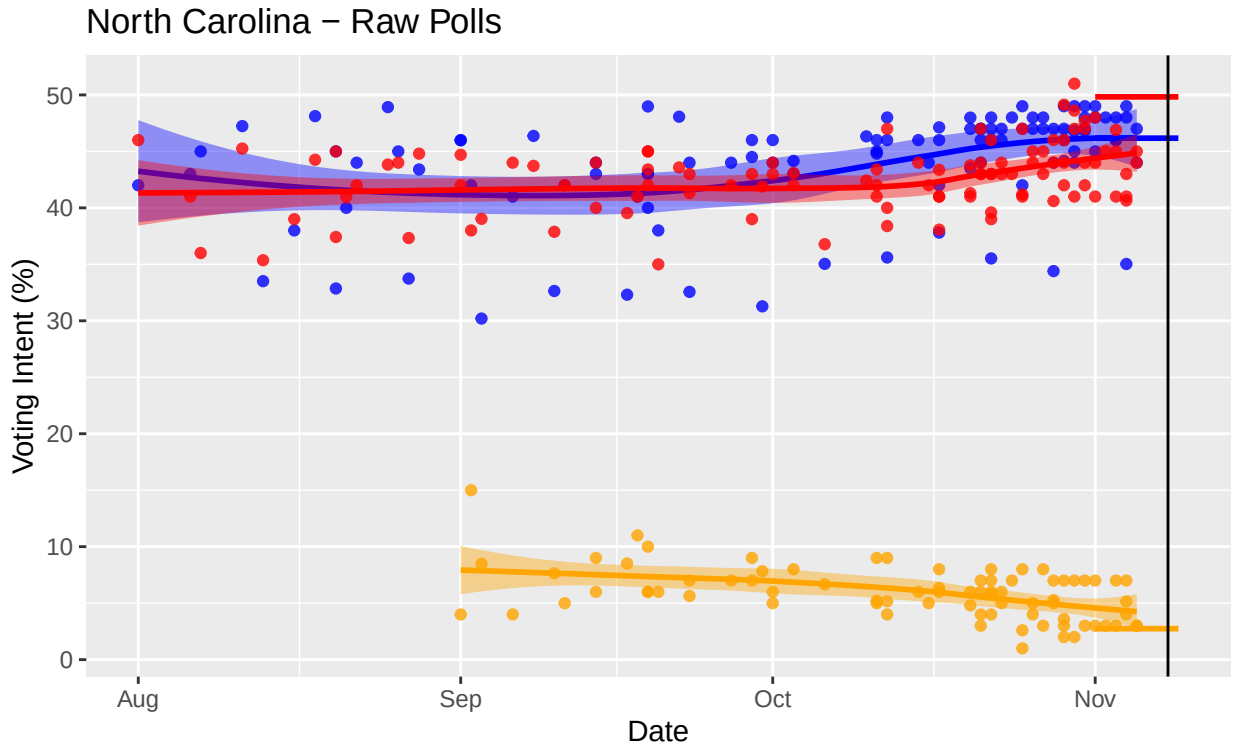

New Jersey − Adjusted Polls
Adjusted Voting Intent (%)
55
50
45
40
35
Sep
Oct
Nov
Date

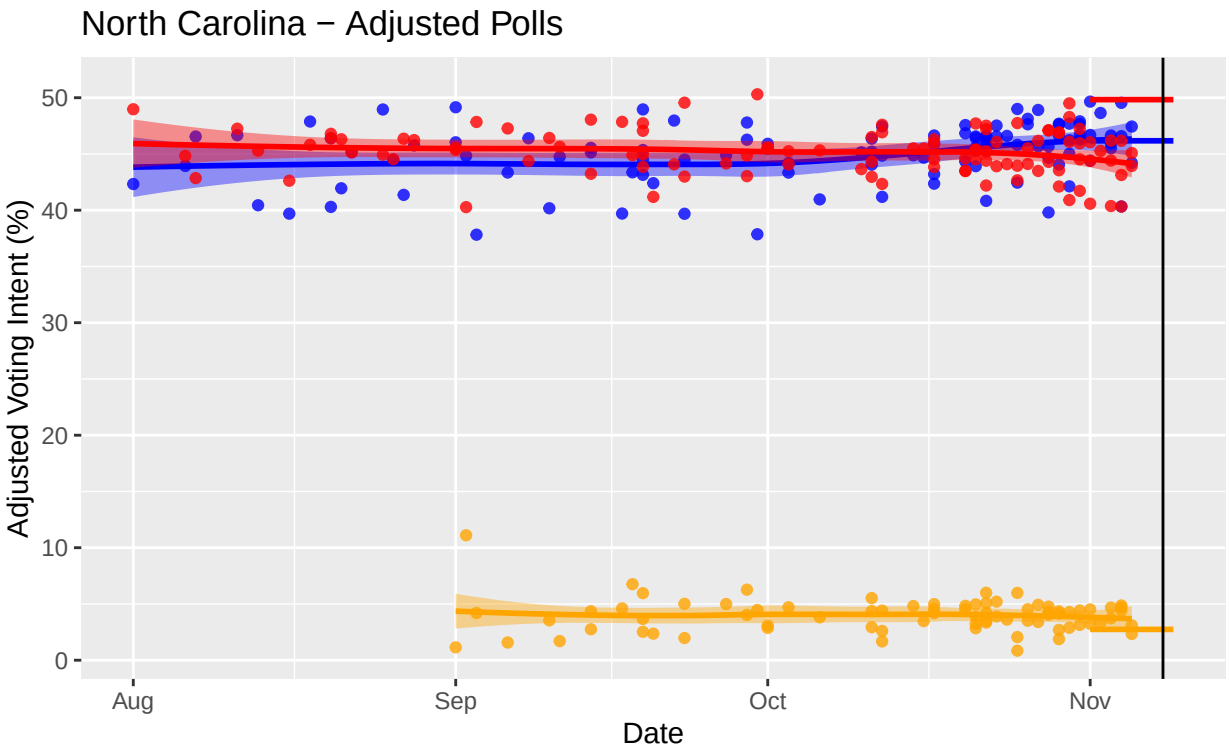

New Mexico − Raw Polls
Voting Intent (%)
50
40
30
20
10
Sep
Oct
Nov
Date

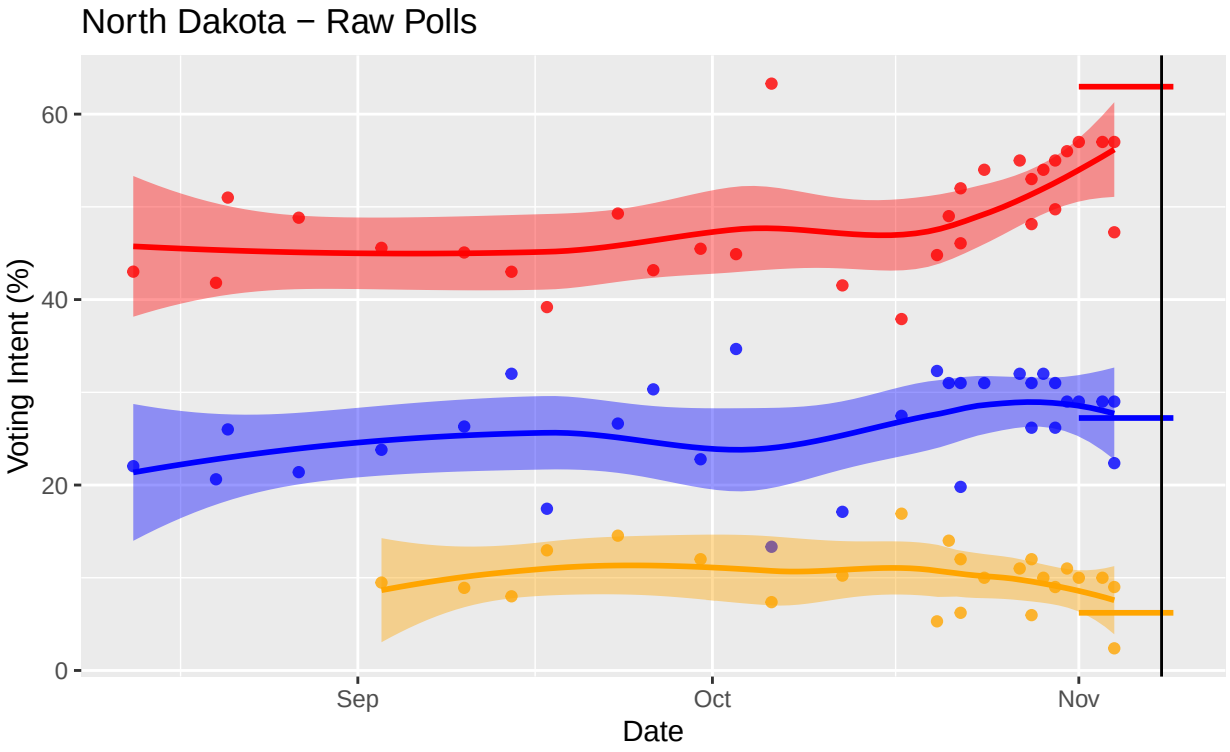

New Mexico − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
20
10
Sep
Oct
Nov
Date

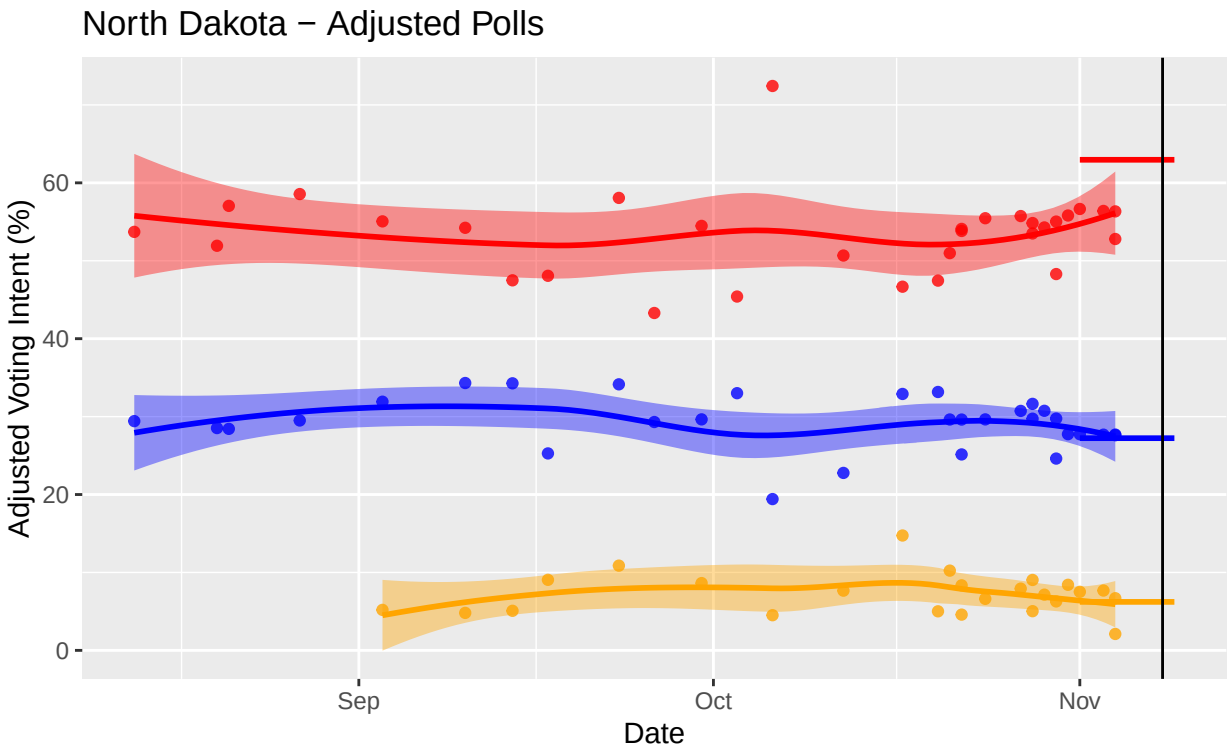

New York − Raw Polls
Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

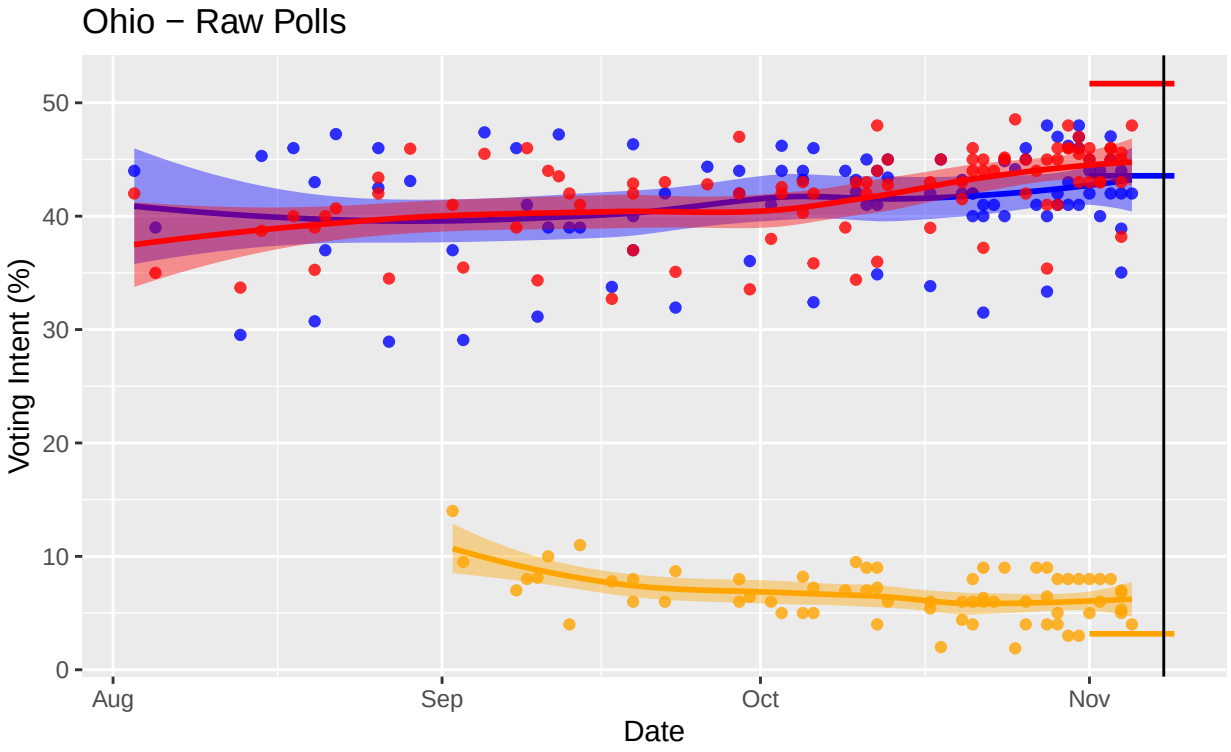

New York − Adjusted Polls
Adjusted Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

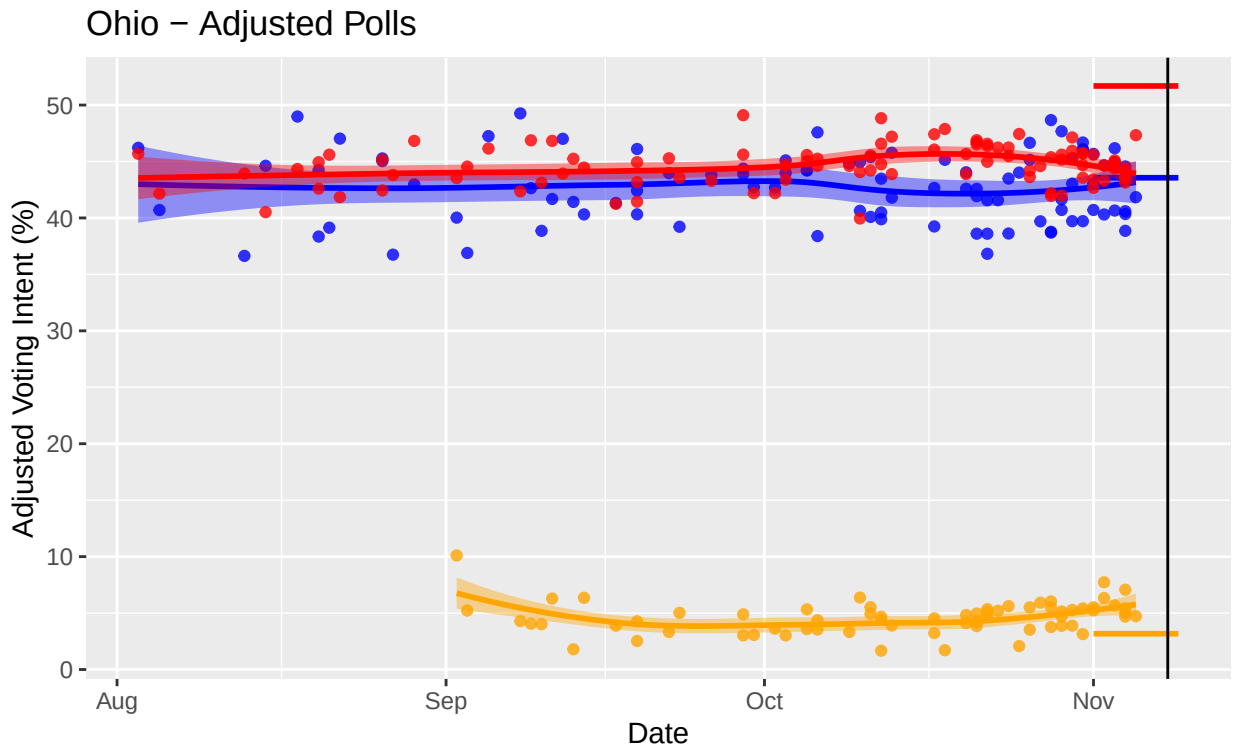

North Carolina − Raw Polls
Voting Intent (%)
50
40
30
20
10
0
Aug
Sep
Oct
Nov
Date

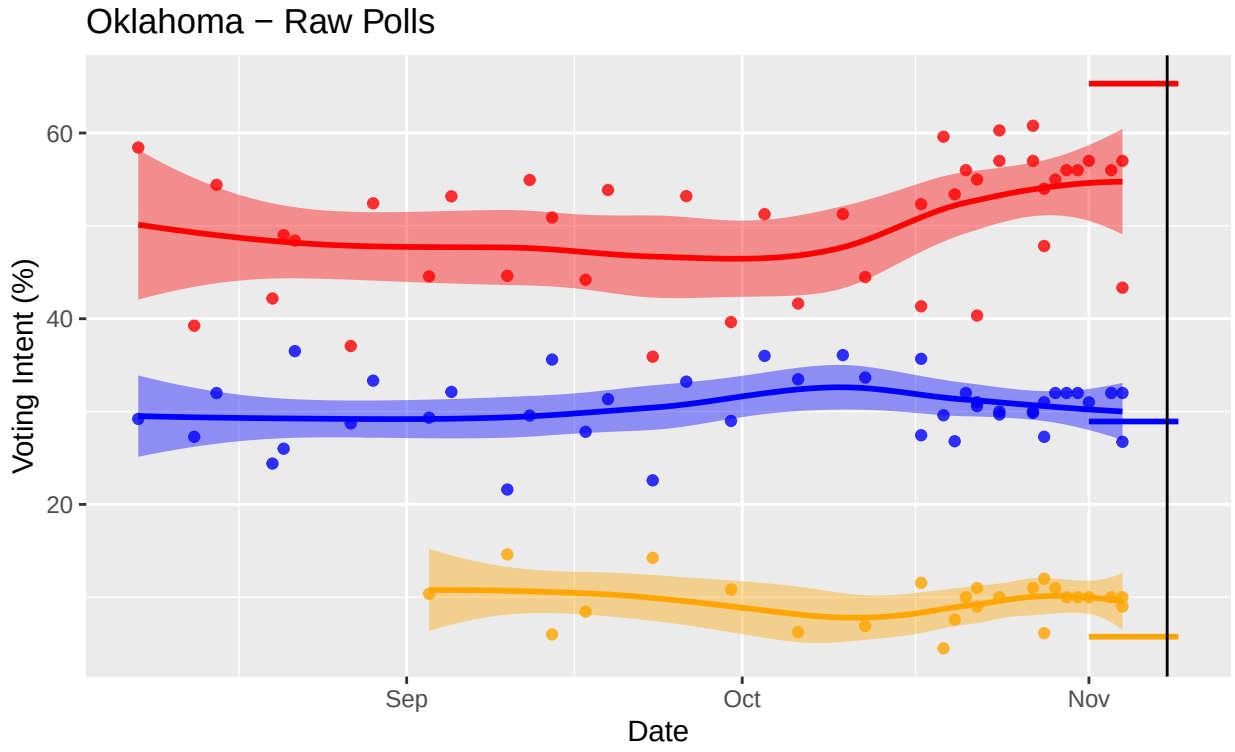

North Carolina − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
20
10
0
Aug
Sep
Oct
Nov
Date

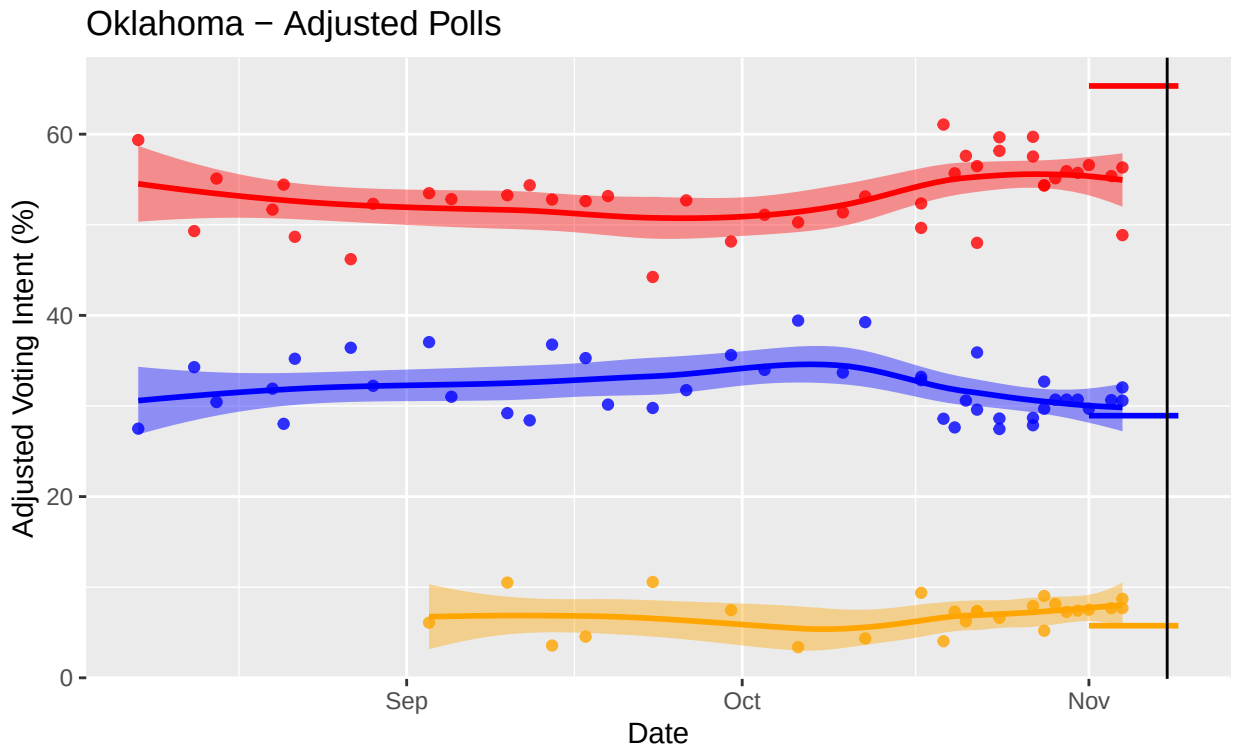

North Dakota − Raw Polls
Voting Intent (%)
60
40
20
0
Sep
Oct
Nov
Date

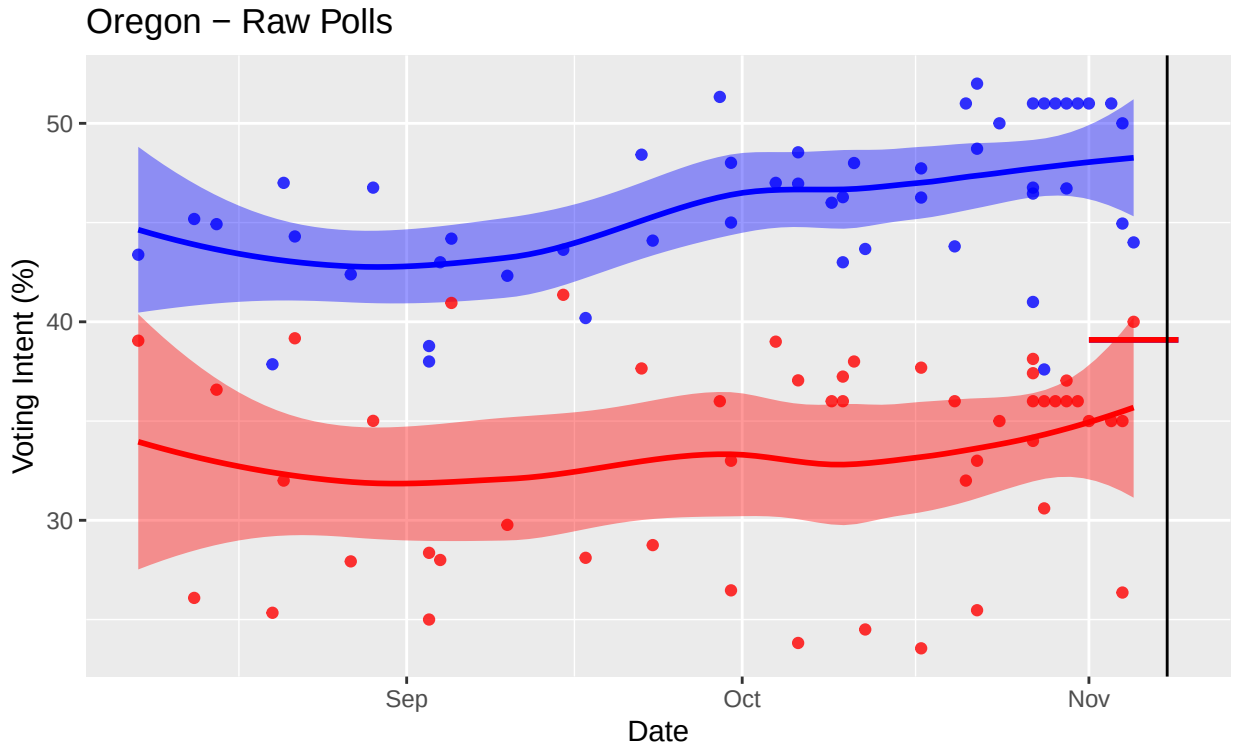

North Dakota − Adjusted Polls
Adjusted Voting Intent (%)
60
40
20
0
Sep
Oct
Nov
Date

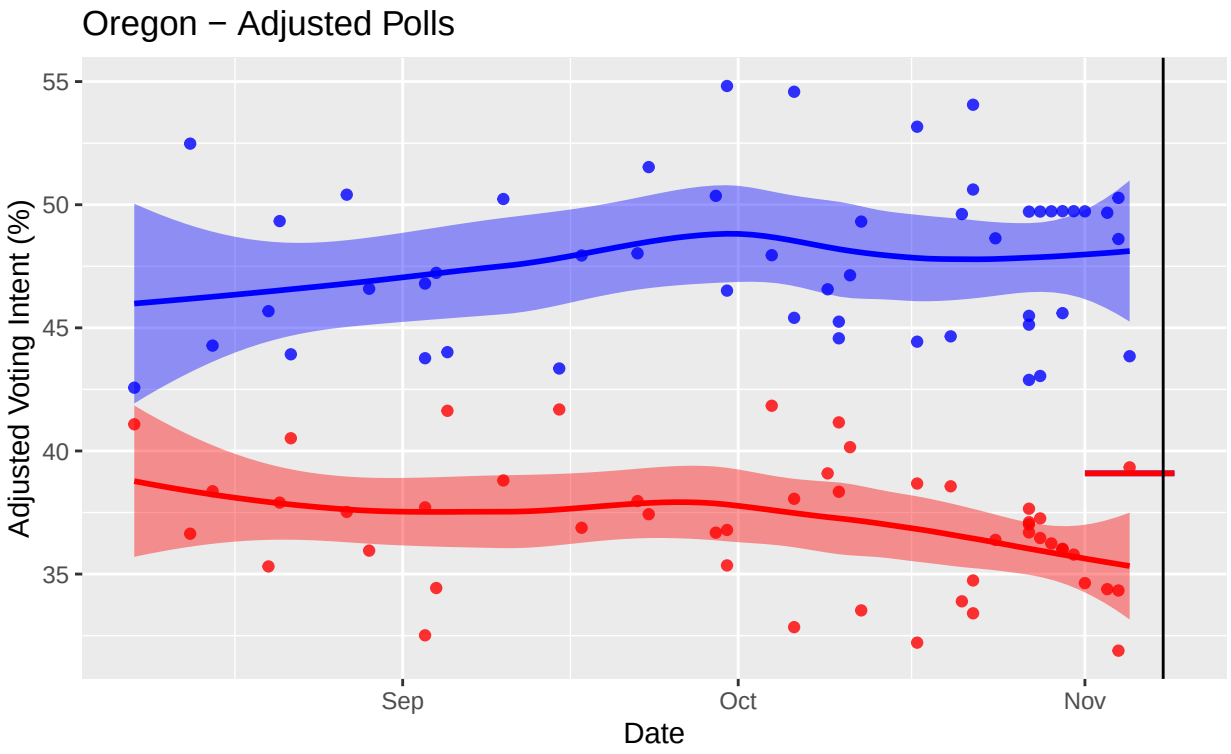

Ohio − Raw Polls
Voting Intent (%)
50
40
30
20
10
0
Aug
Sep
Oct
Nov
Date

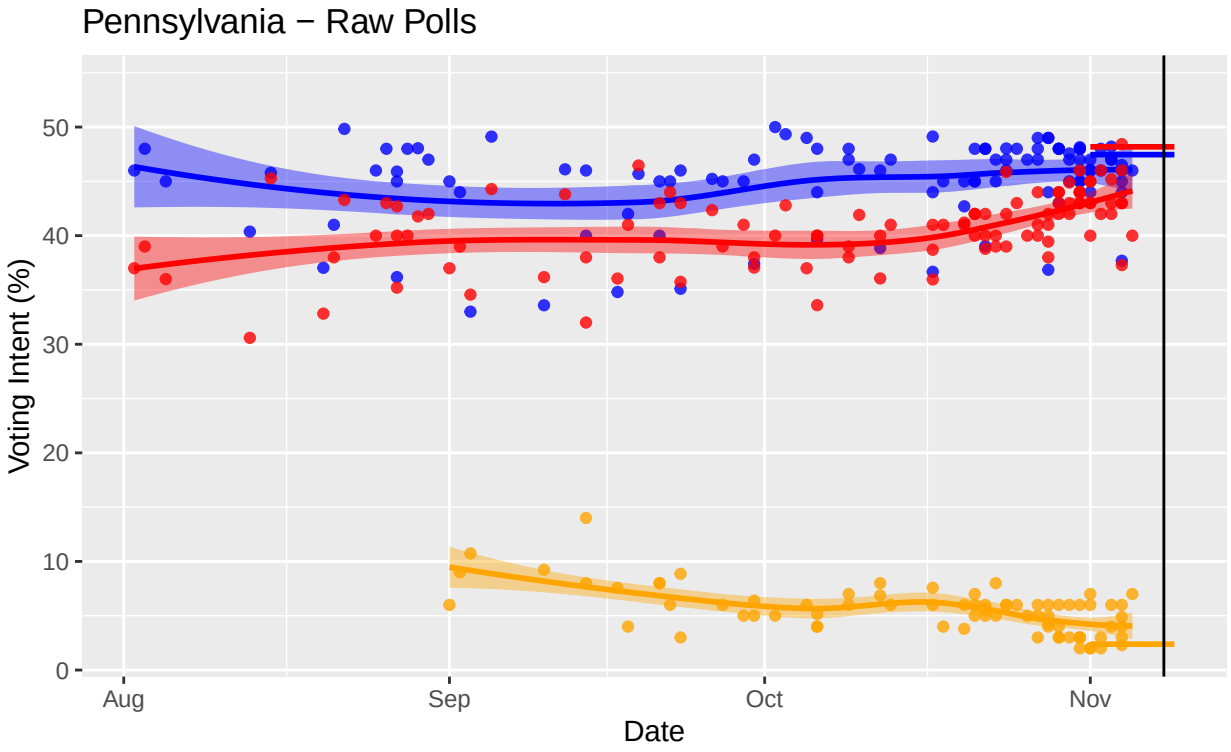

Ohio − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
20
10
0
Aug
Sep
Oct
Nov
Date

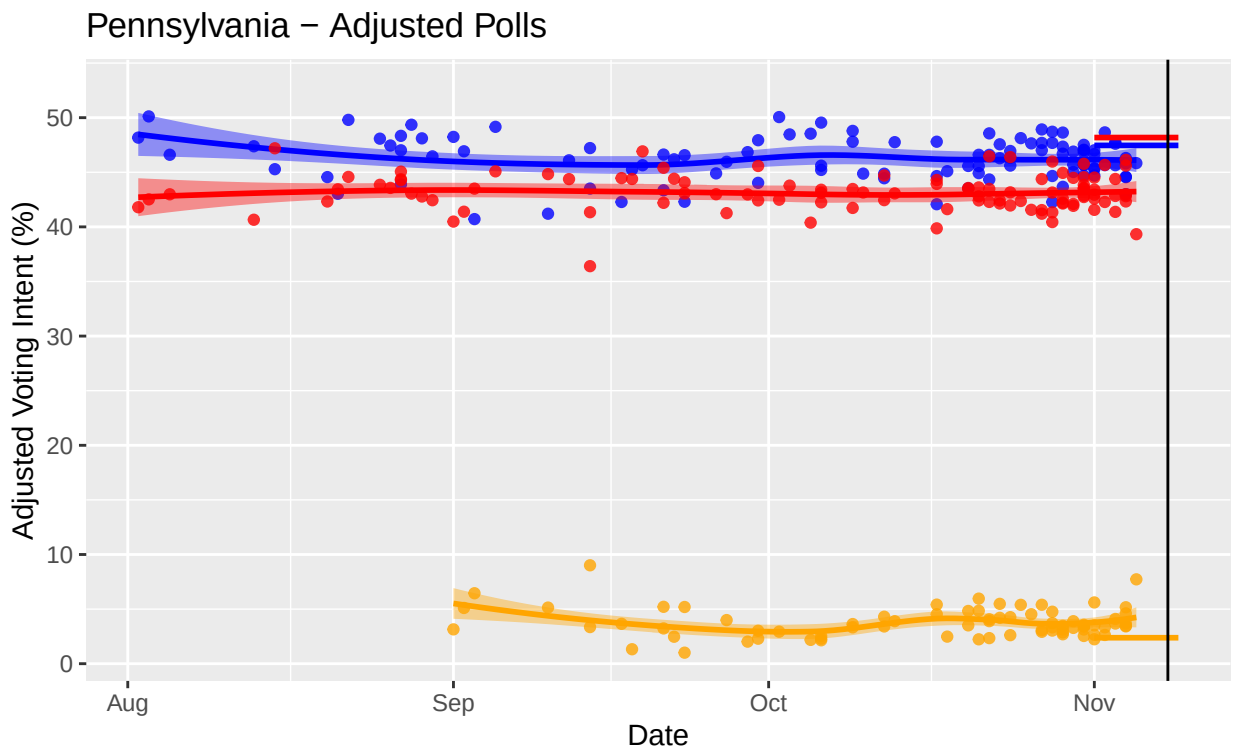

Oklahoma − Raw Polls
Voting Intent (%)
60
40
20
Sep
Oct
Nov
Date

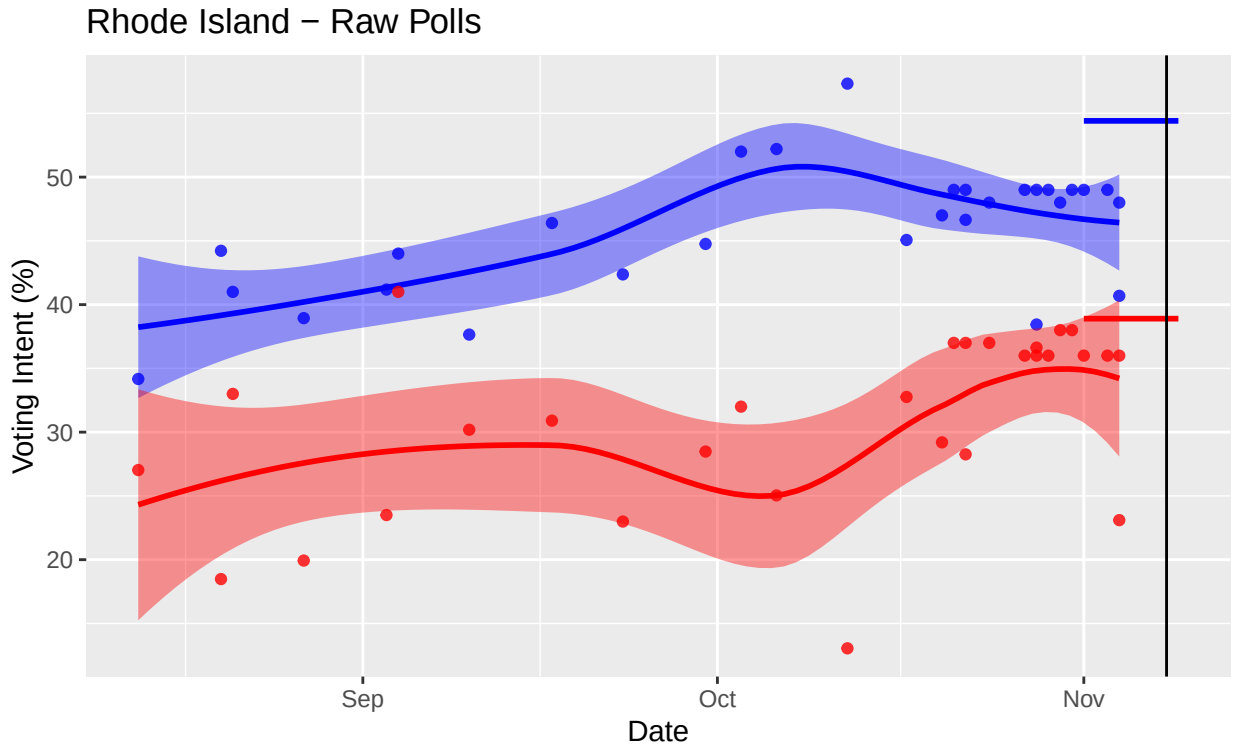

Oklahoma − Adjusted Polls
Adjusted Voting Intent (%)
60
40
20
0
Sep
Oct
Nov
Date

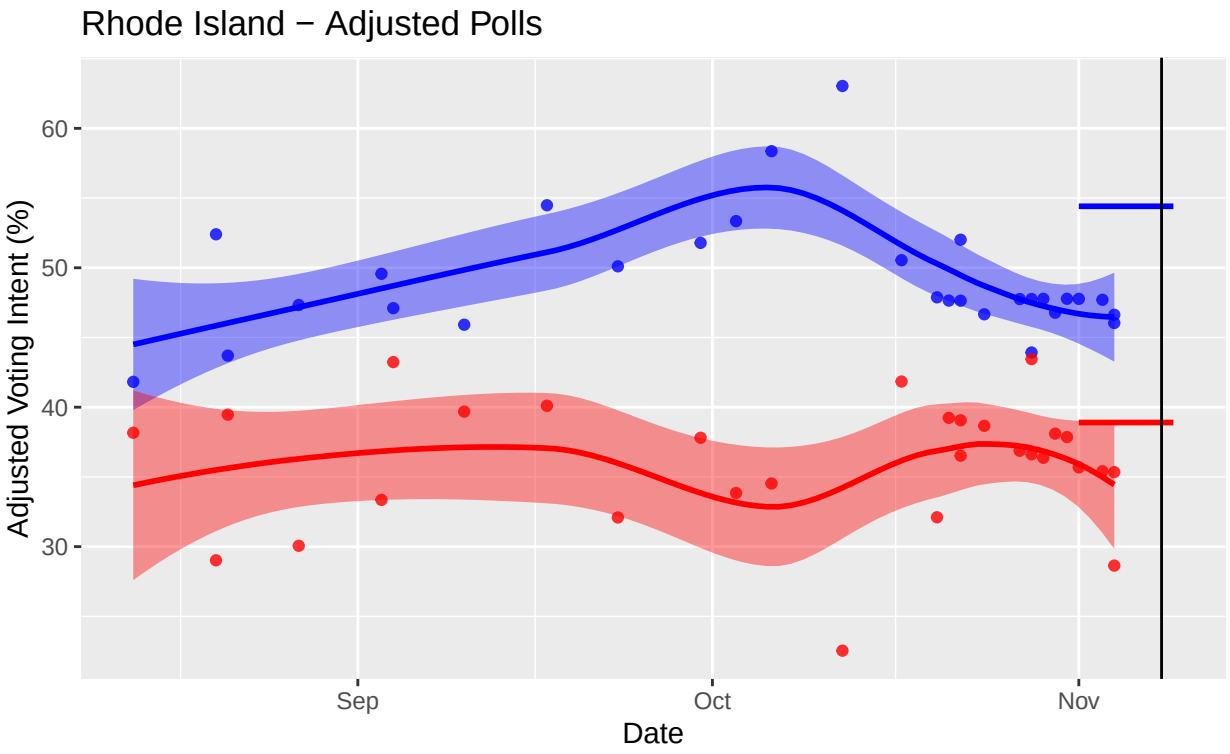

Oregon − Raw Polls
Voting Intent (%)
50
40
30
Sep
Oct
Nov
Date

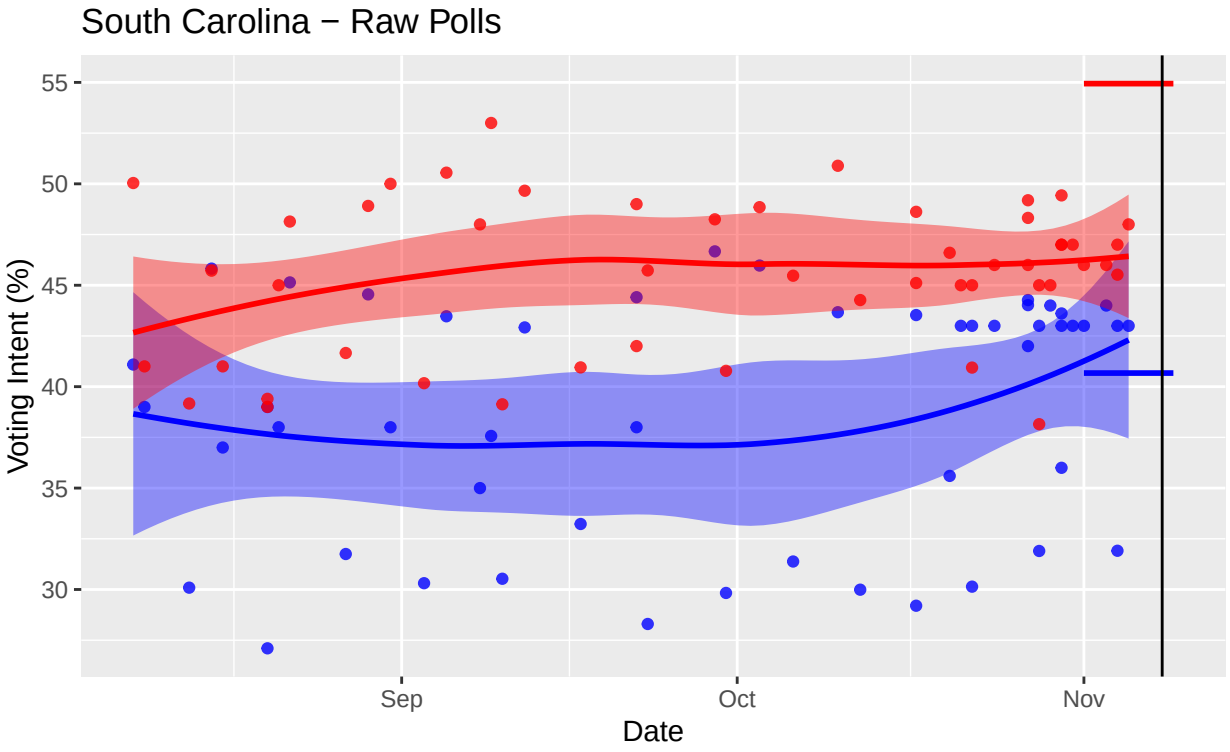

Oregon − Adjusted Polls
Adjusted Voting Intent (%)
55
50
45
40
35
Sep
Oct
Nov
Date

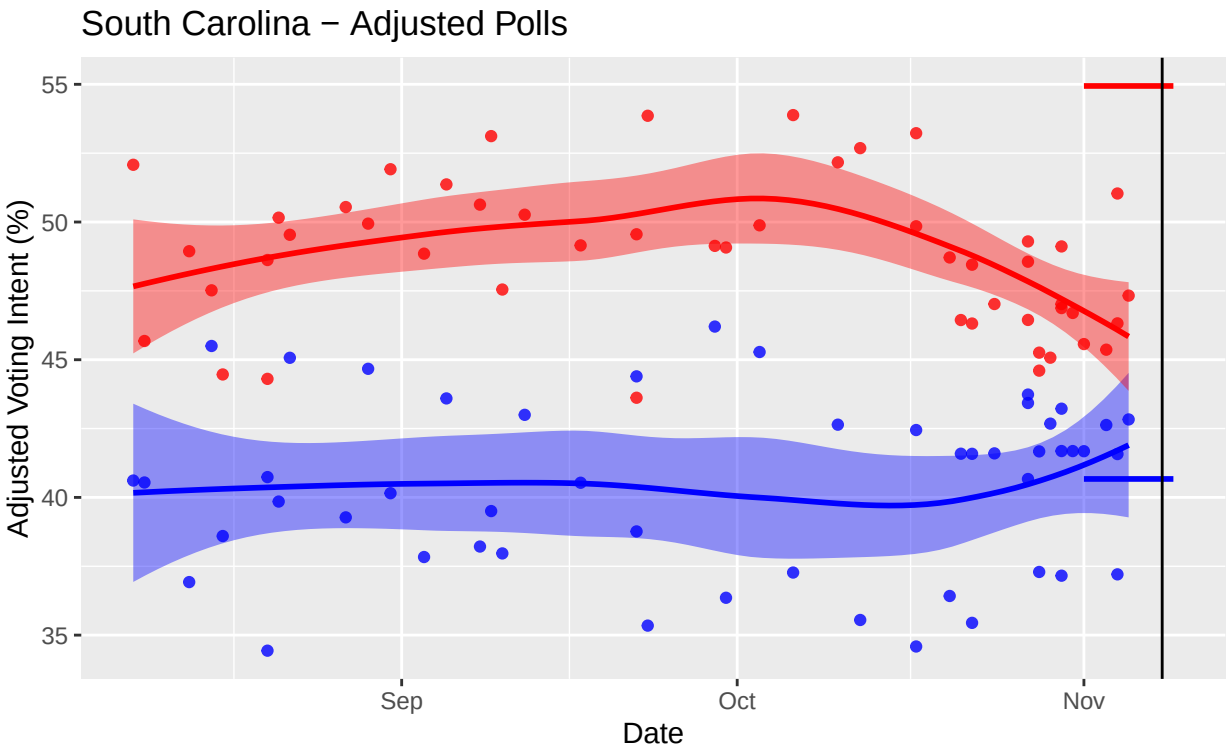

Pennsylvania − Raw Polls
Voting Intent (%)
50
40
30
20
10
0
Aug
Sep
Oct
Nov
Date

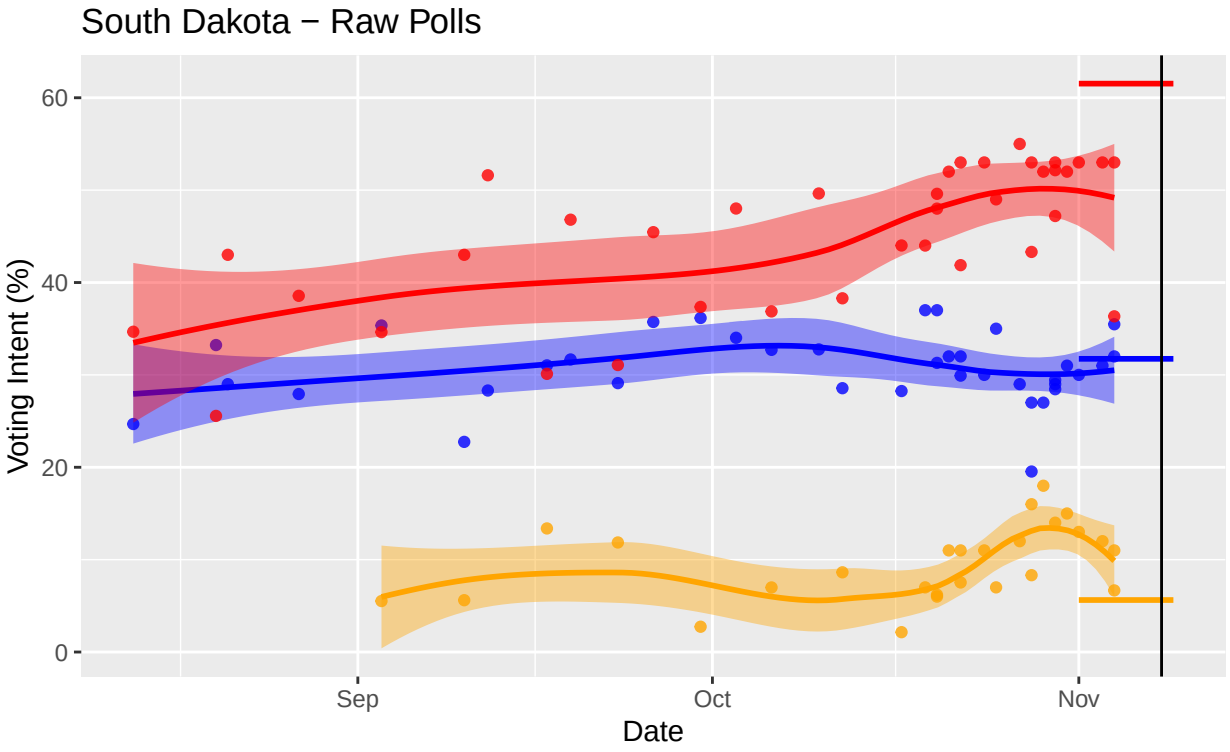

Pennsylvania − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
20
10
0
Aug
Sep
Oct
Nov
Date

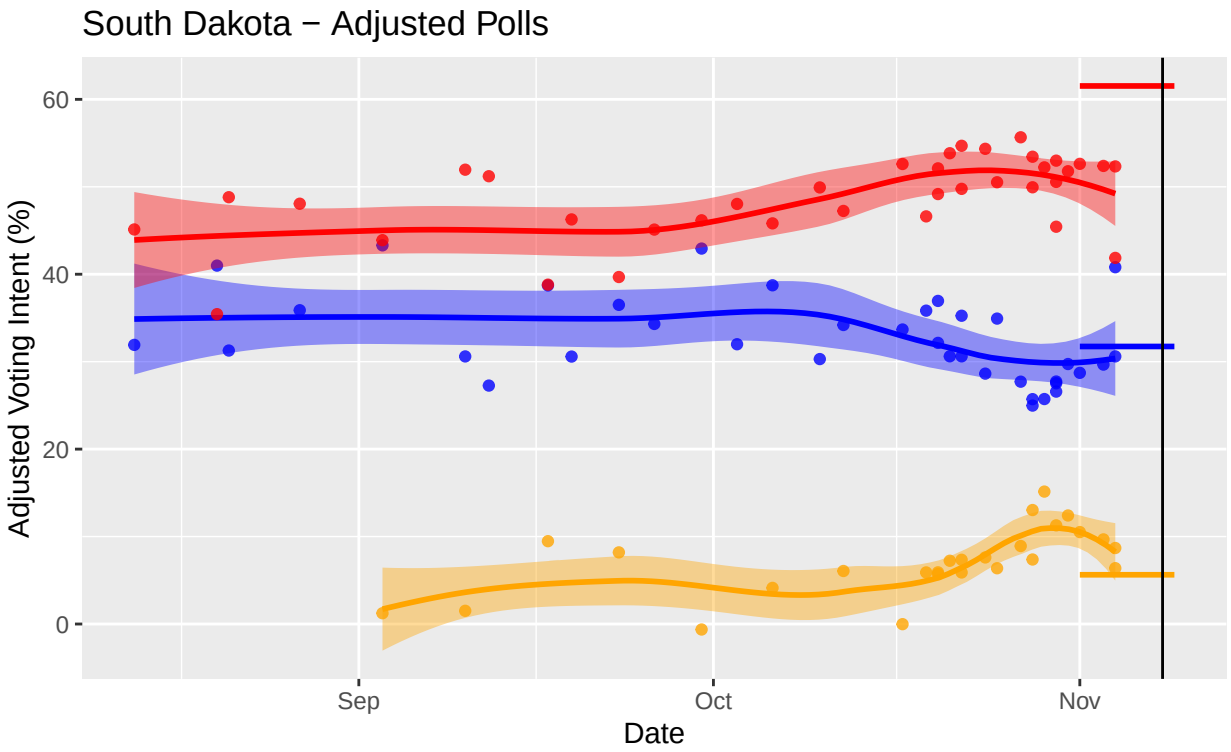

Rhode Island − Raw Polls
Voting Intent (%)
50
40
30
20
Sep
Oct
Nov
Date

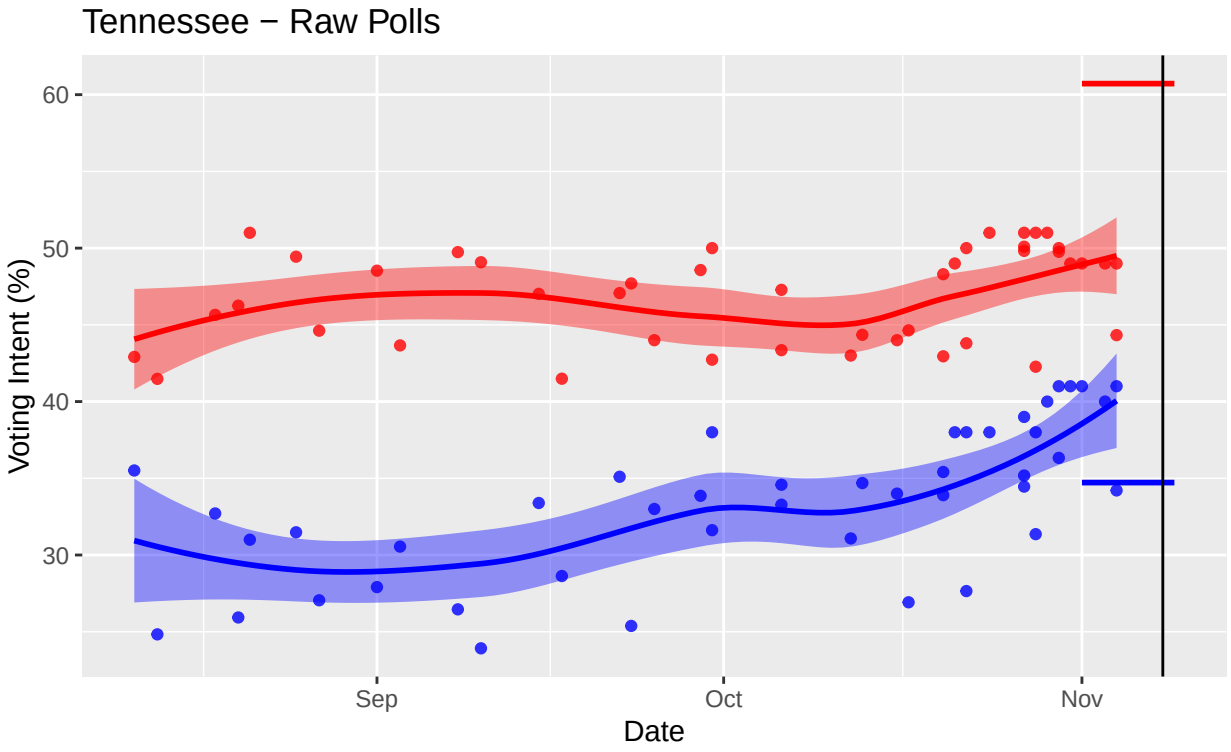

Rhode Island − Adjusted Polls
Adjusted Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

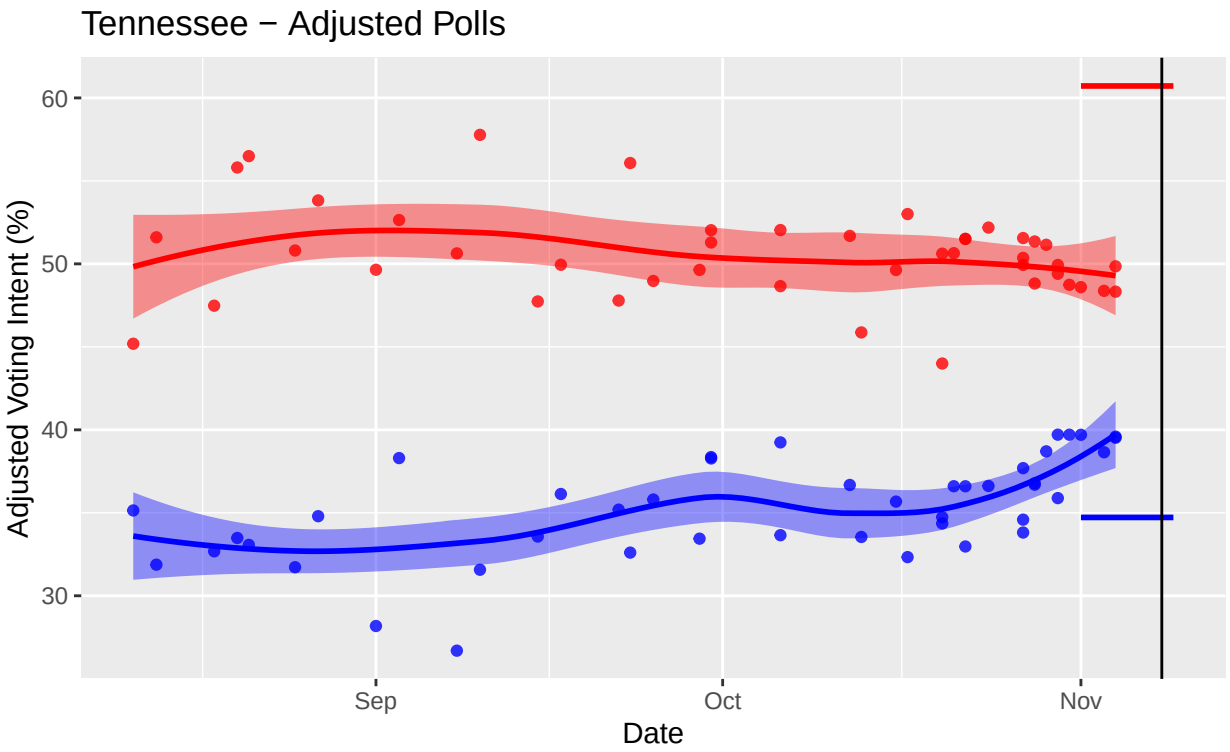

South Carolina − Raw Polls
Voting Intent (%)
55
50
45
40
35
30
Sep
Oct
Nov
Date

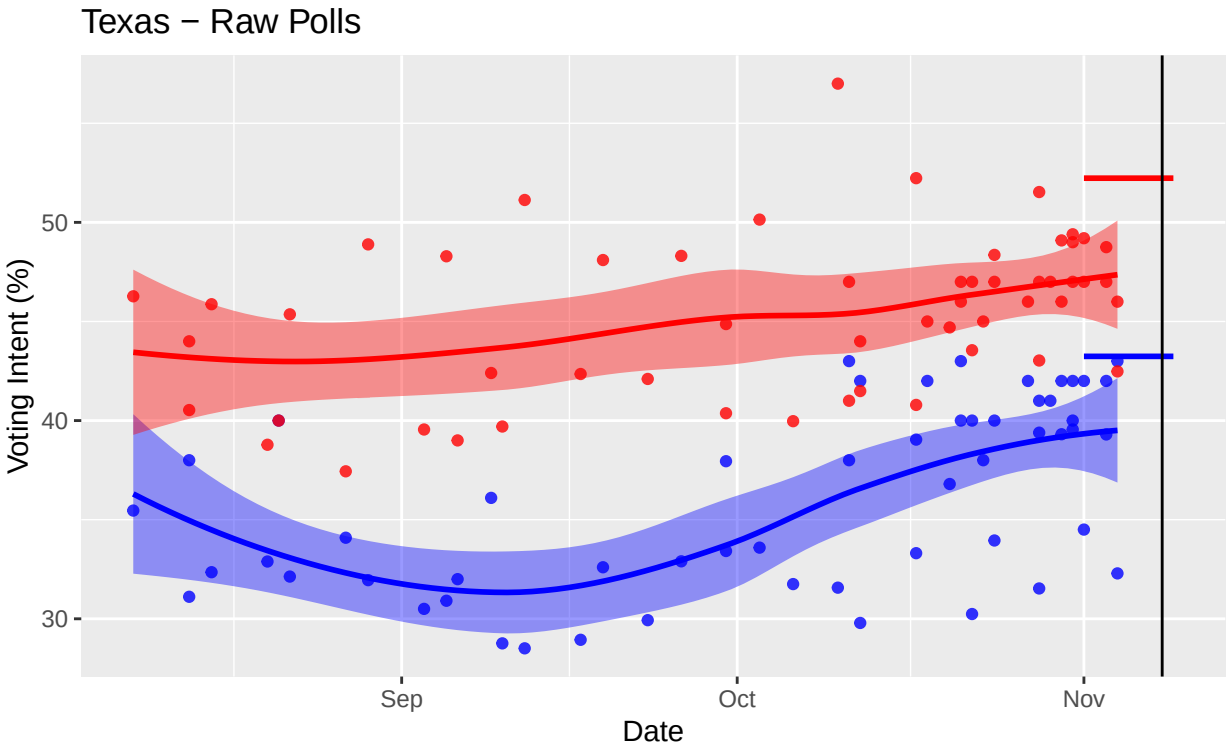

South Carolina − Adjusted Polls
Adjusted Voting Intent (%)
55
50
45
40
35
Sep
Oct
Nov
Date

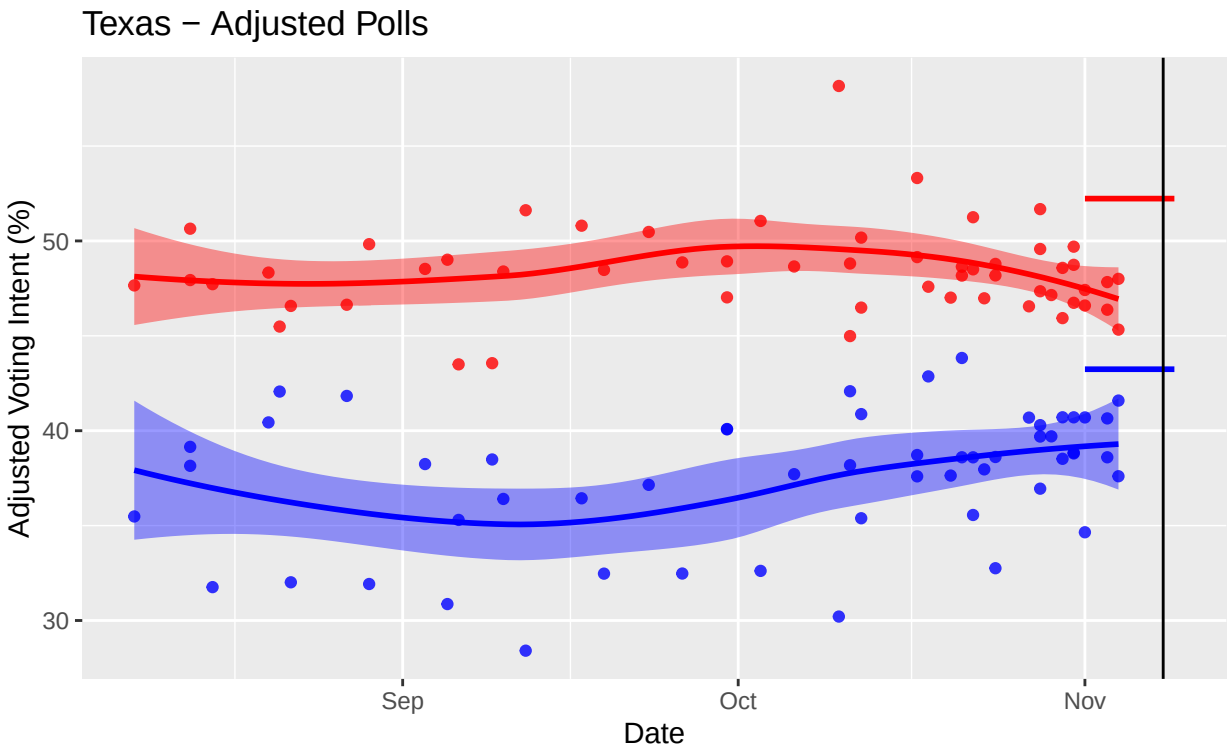

South Dakota − Raw Polls
Voting Intent (%)
60
40
20
0
Sep
Oct
Nov
Date

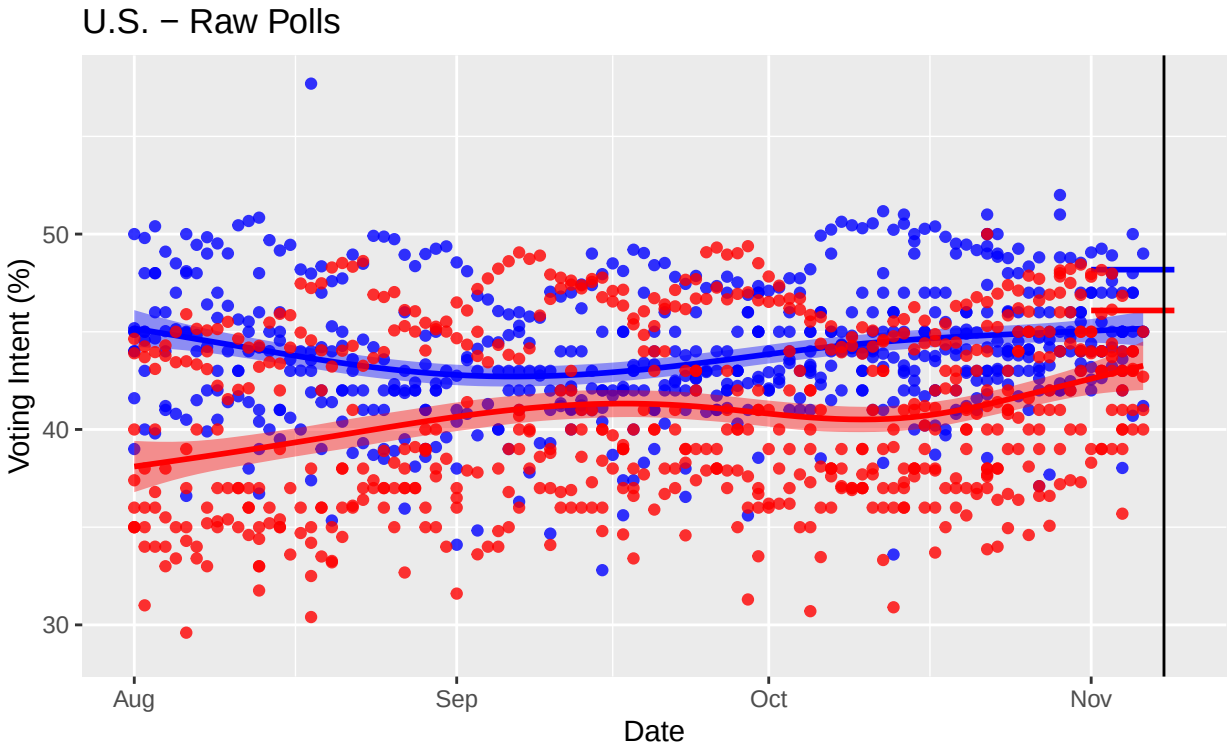

South Dakota − Adjusted Polls
Adjusted Voting Intent (%)
60
40
20
0
Sep
Oct
Nov
Date

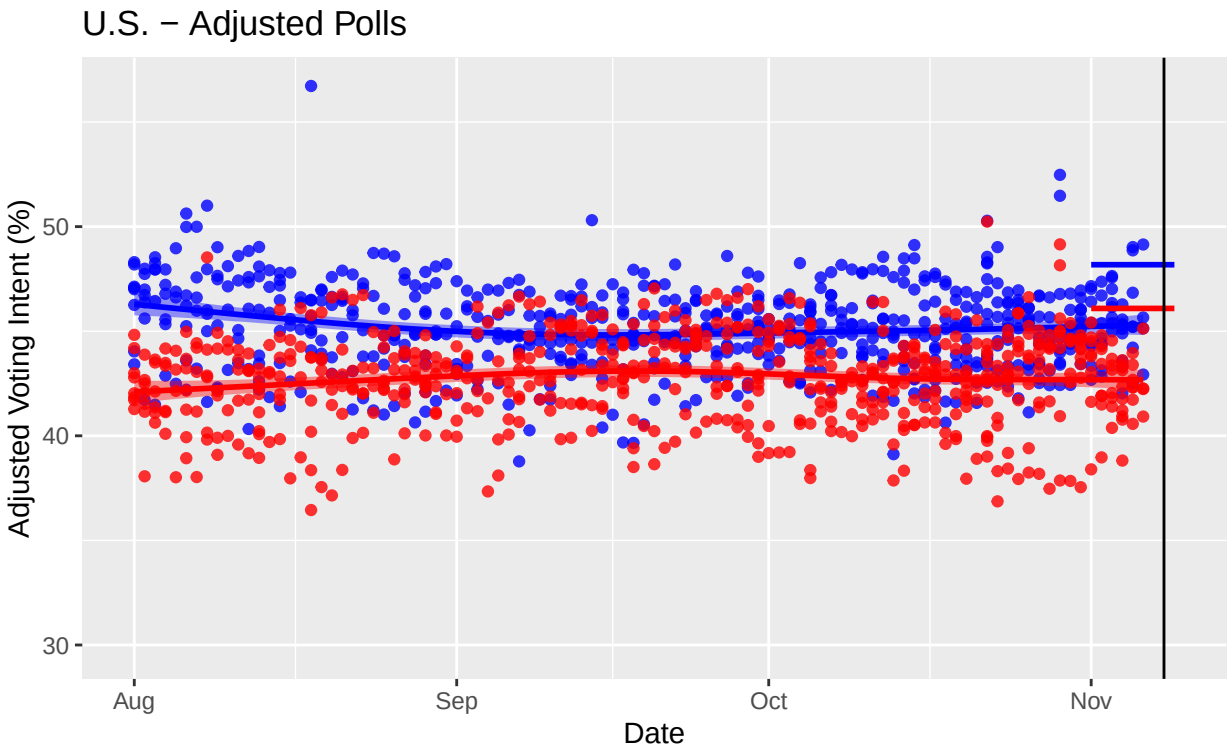

Tennessee − Raw Polls
Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

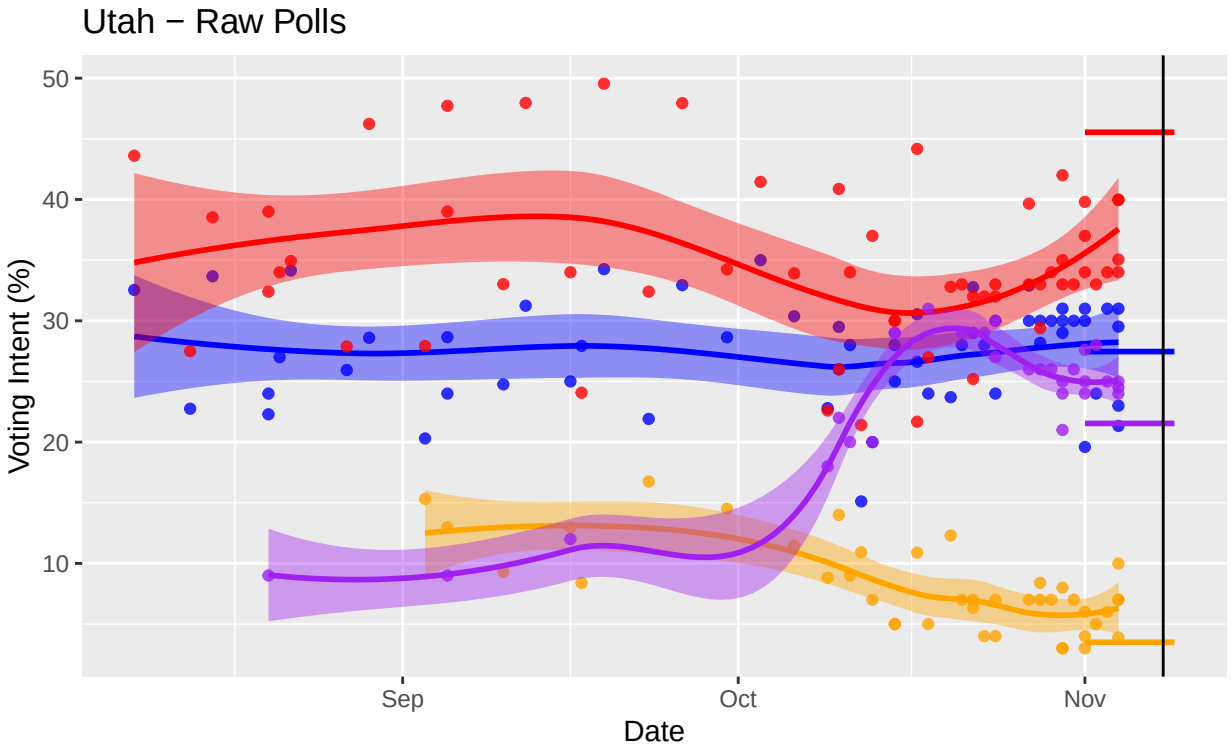

Tennessee − Adjusted Polls
Adjusted Voting Intent (%)
60
50
40
30
Sep
Oct
Nov
Date

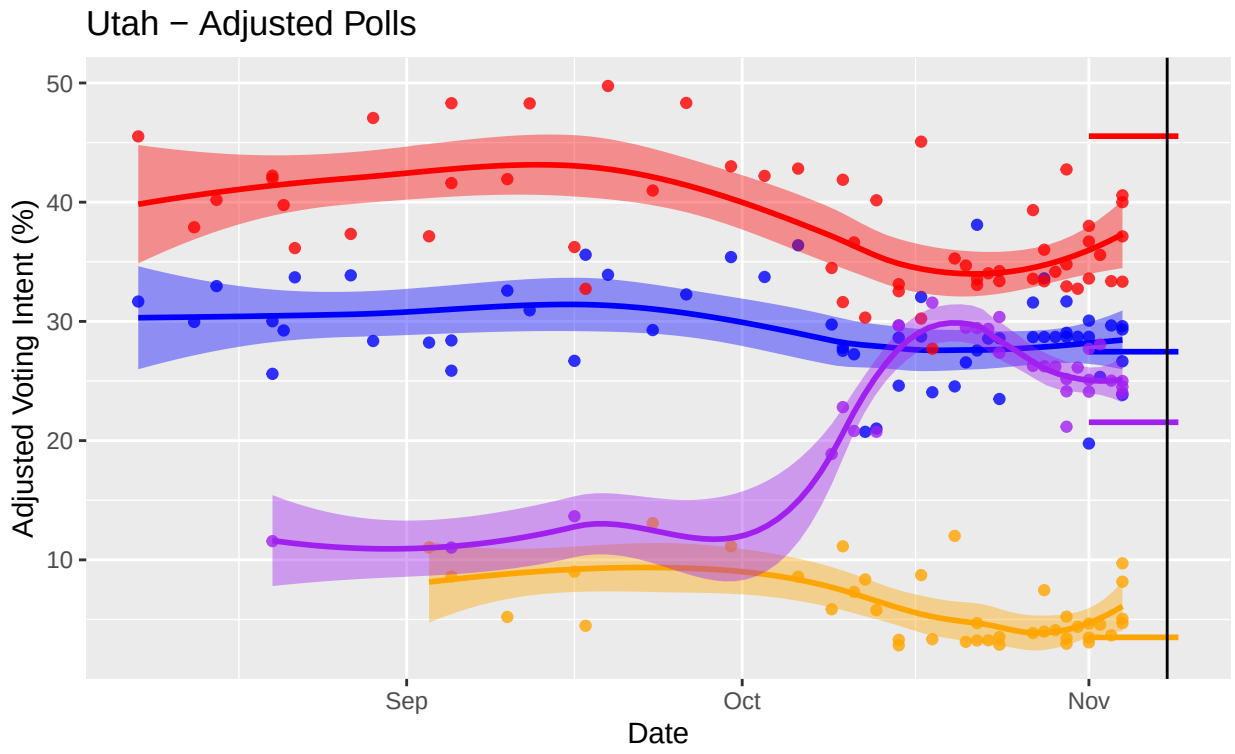

Texas − Raw Polls
Voting Intent (%)
50
40
30
Sep
Oct
Nov
Date

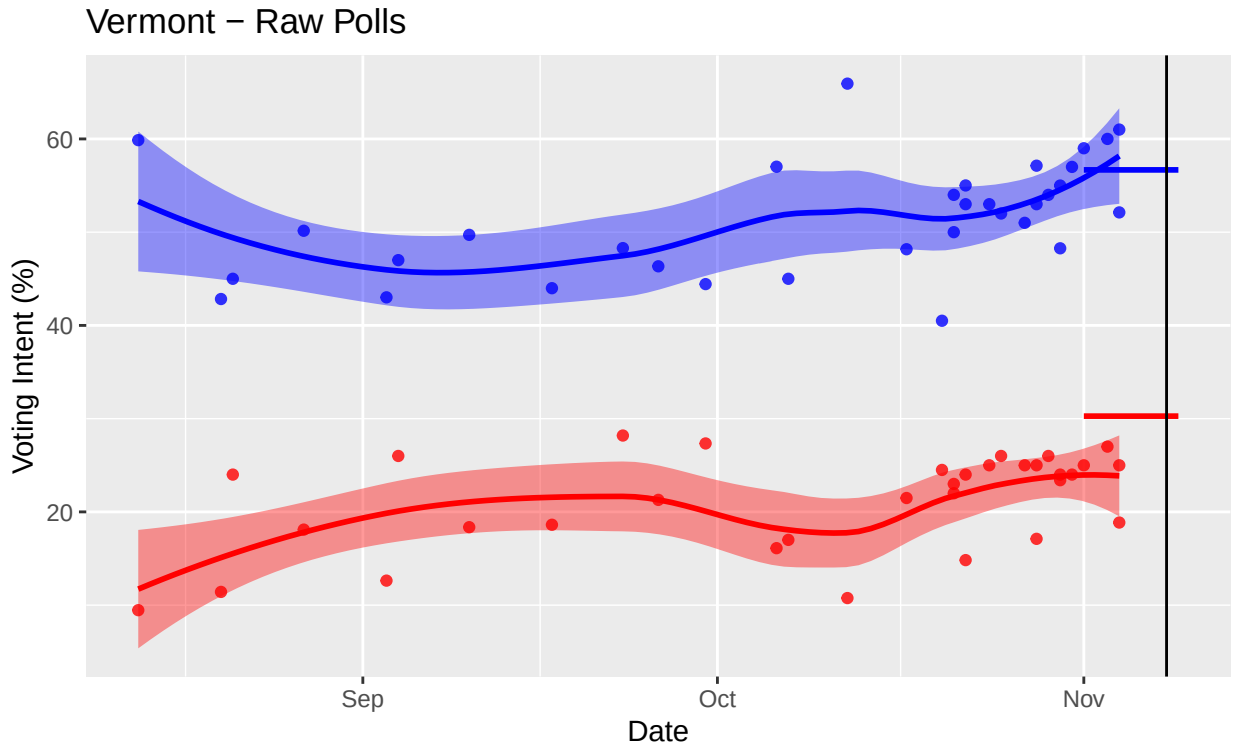

Texas − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
Sep
Oct
Nov
Date

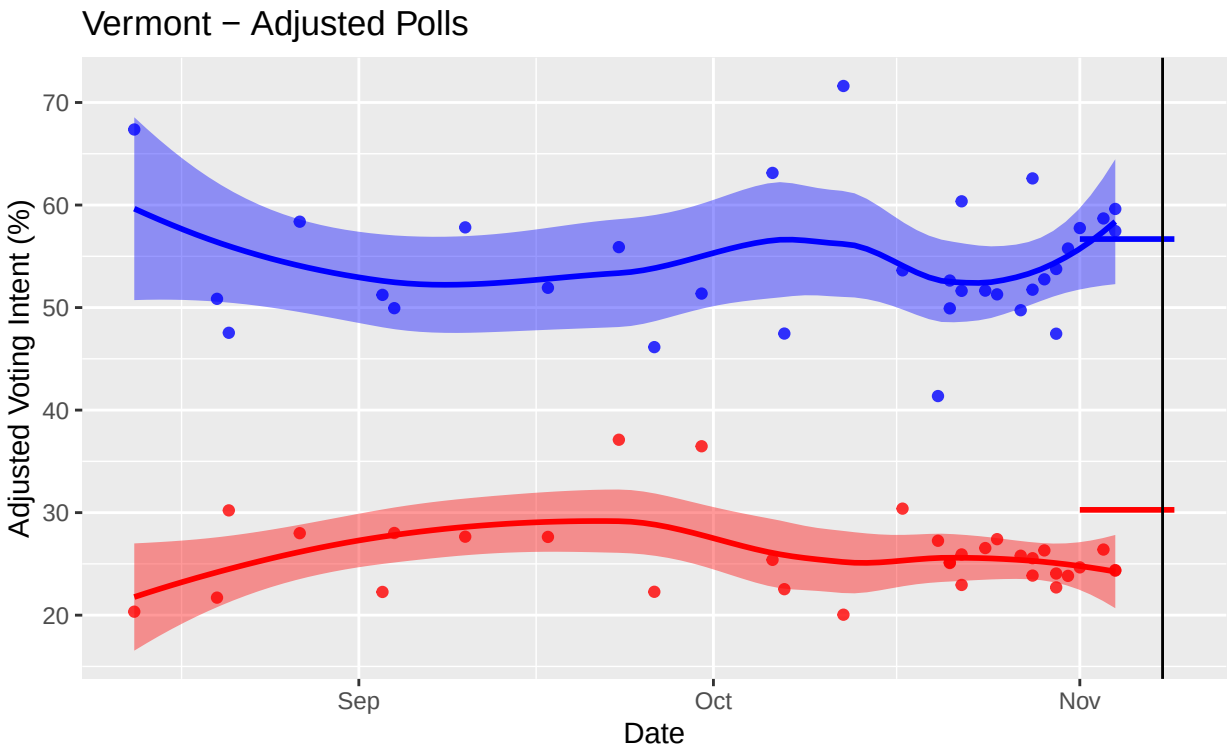

U.S. − Raw Polls
Voting Intent (%)
50
40
30
Aug
Sep
Oct
Nov
Date

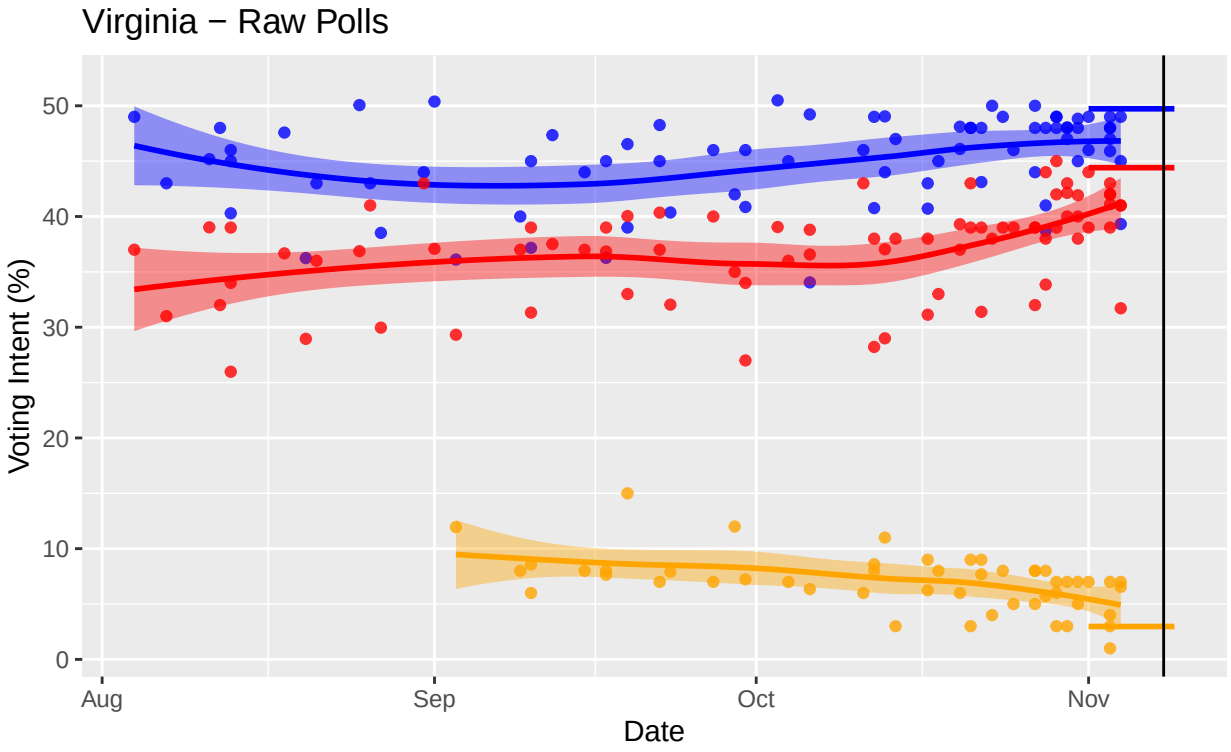

U.S. − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
Aug
Sep
Oct
Nov
Date

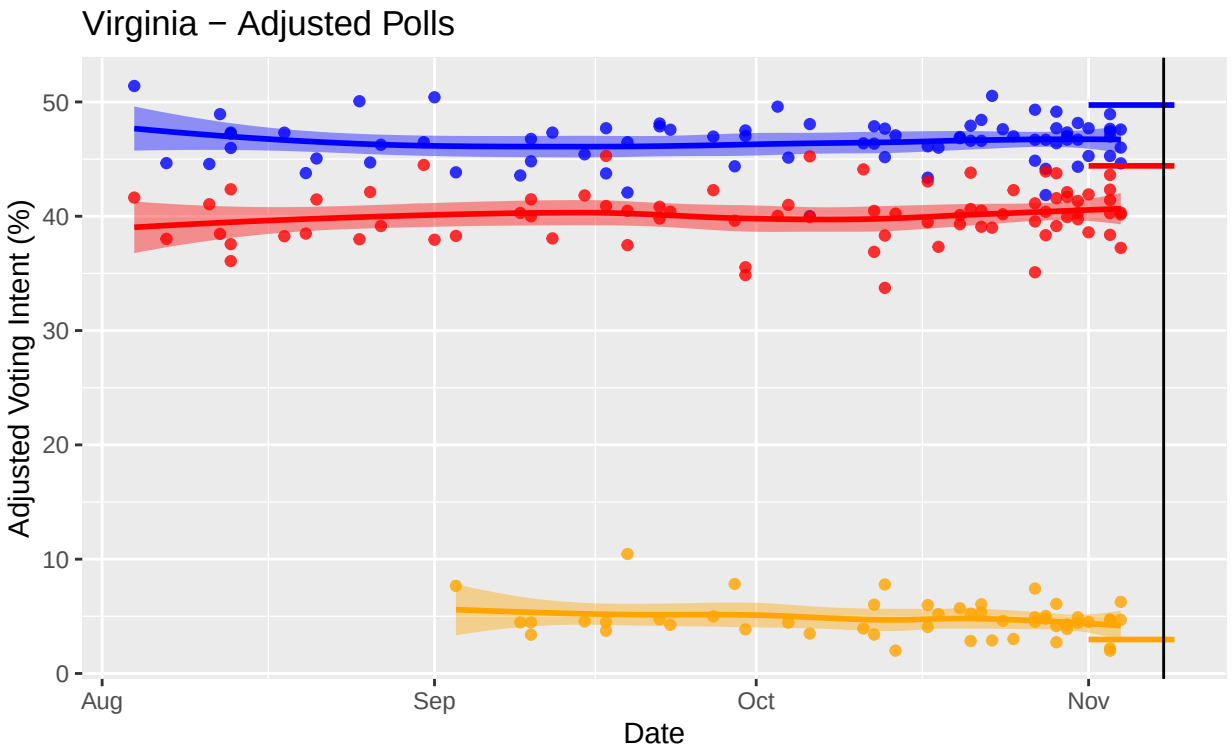

Utah − Raw Polls
Voting Intent (%)
50
40
30
20
10
Sep
Oct
Nov
Date

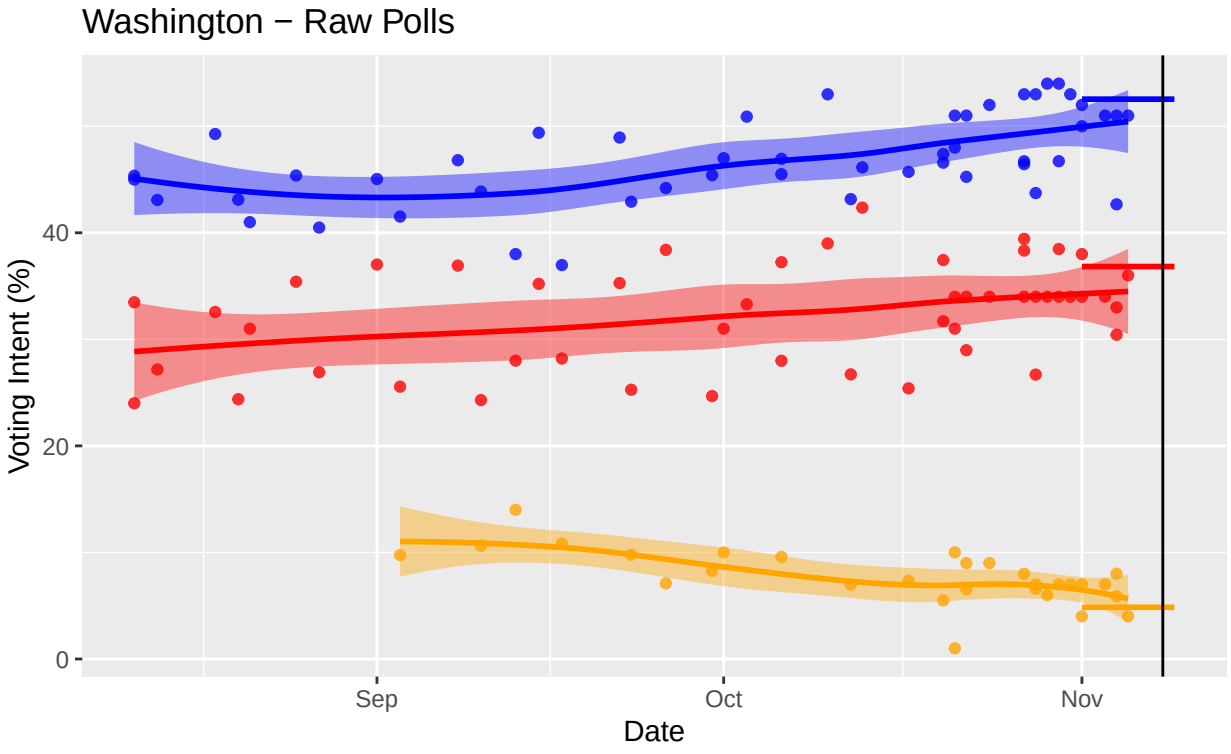

Utah − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
20
10
Sep
Oct
Nov
Date

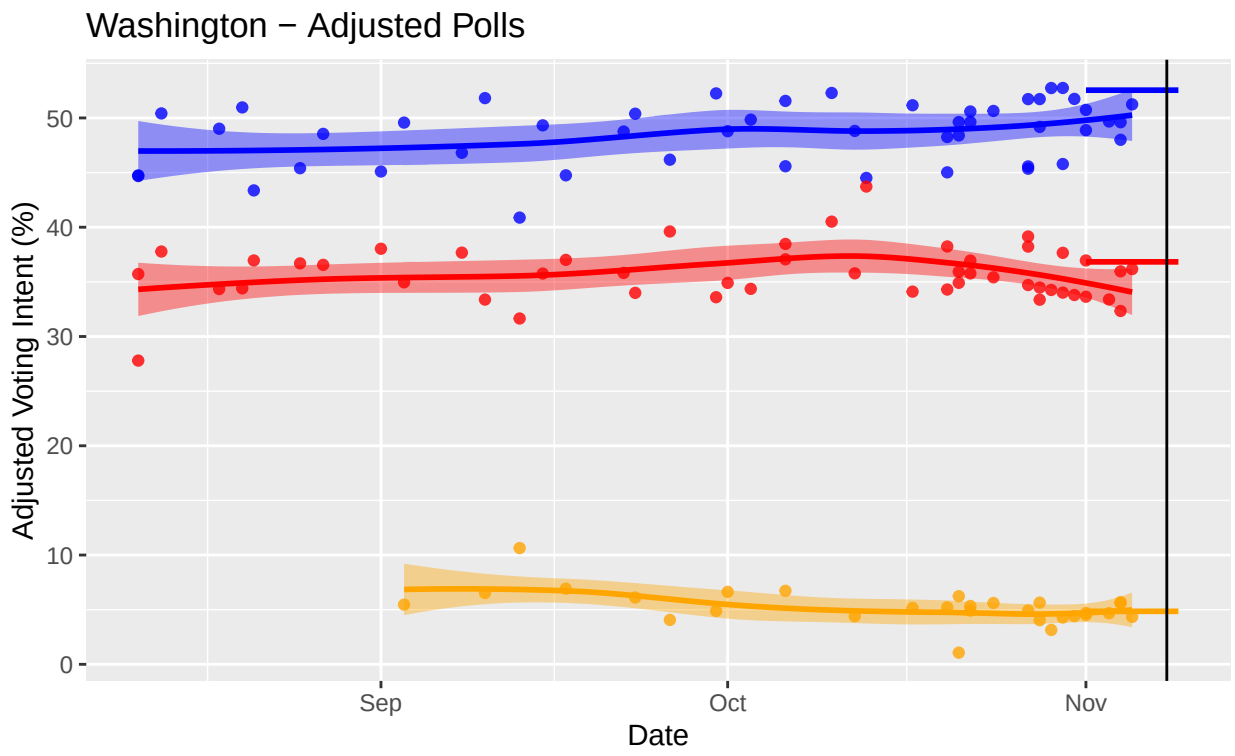

Vermont − Raw Polls
Voting Intent (%)
60
40
20
Sep
Oct
Nov
Date

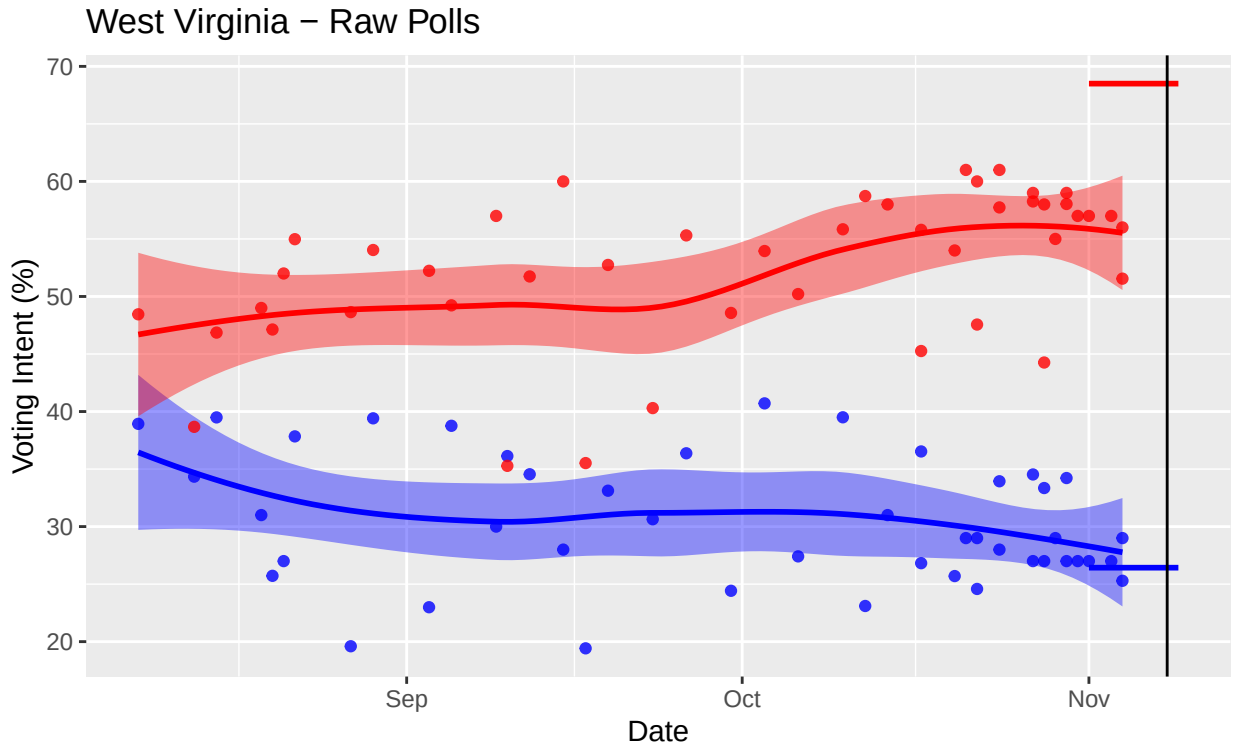

Vermont − Adjusted Polls
Adjusted Voting Intent (%)
70
60
50
40
30
20
Sep
Oct
Nov
Date

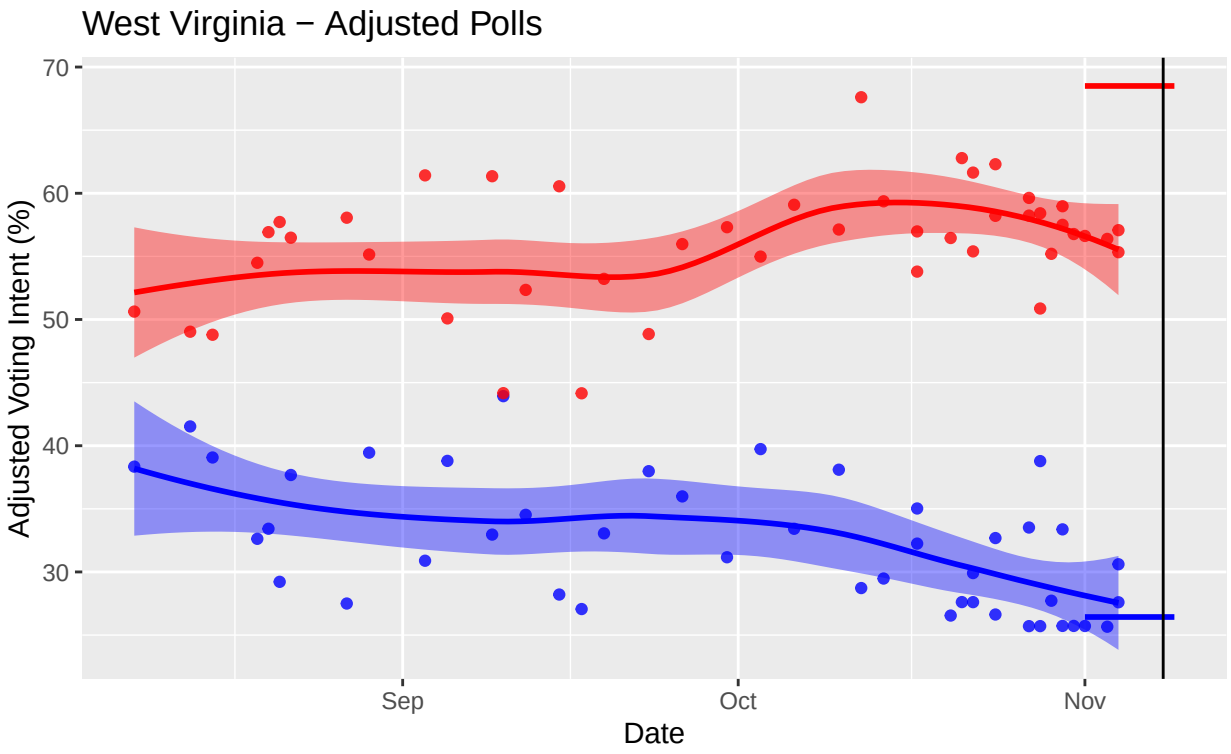

Virginia − Raw Polls
Voting Intent (%)
50
40
30
20
10
0
Aug
Sep
Oct
Nov
Date

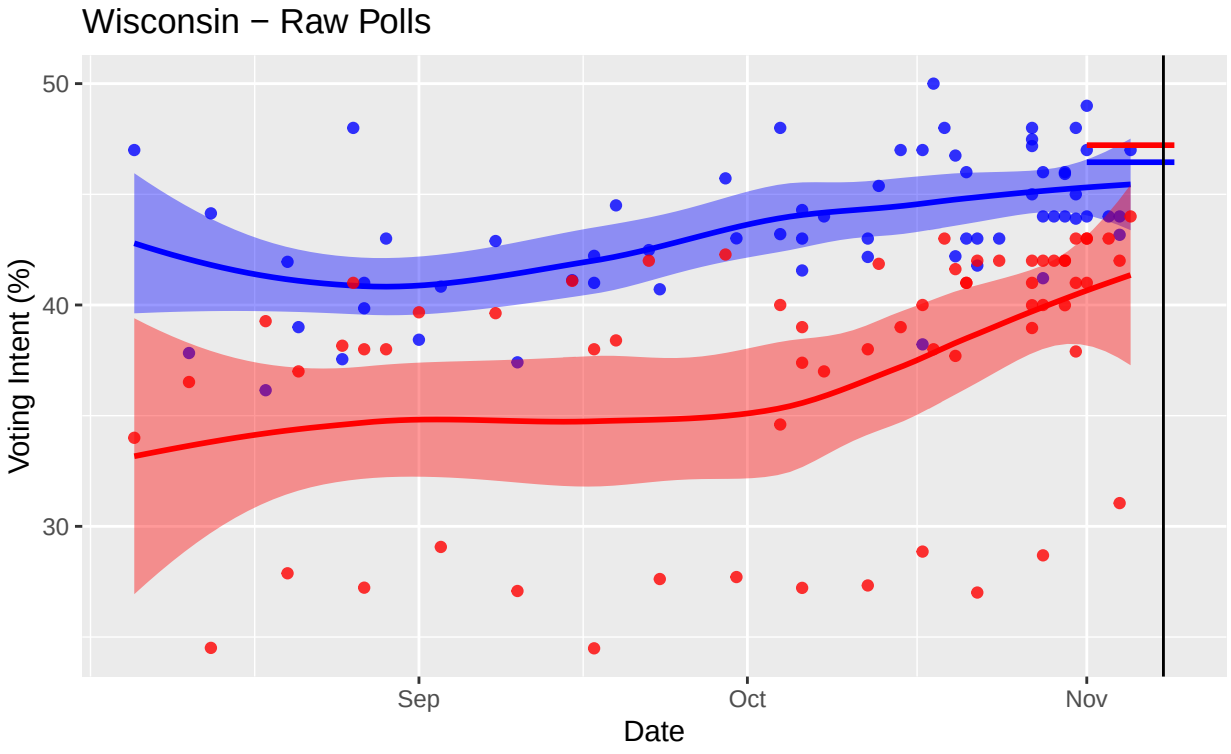

Virginia − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
20
10
0
Aug
Sep
Oct
Nov
Date

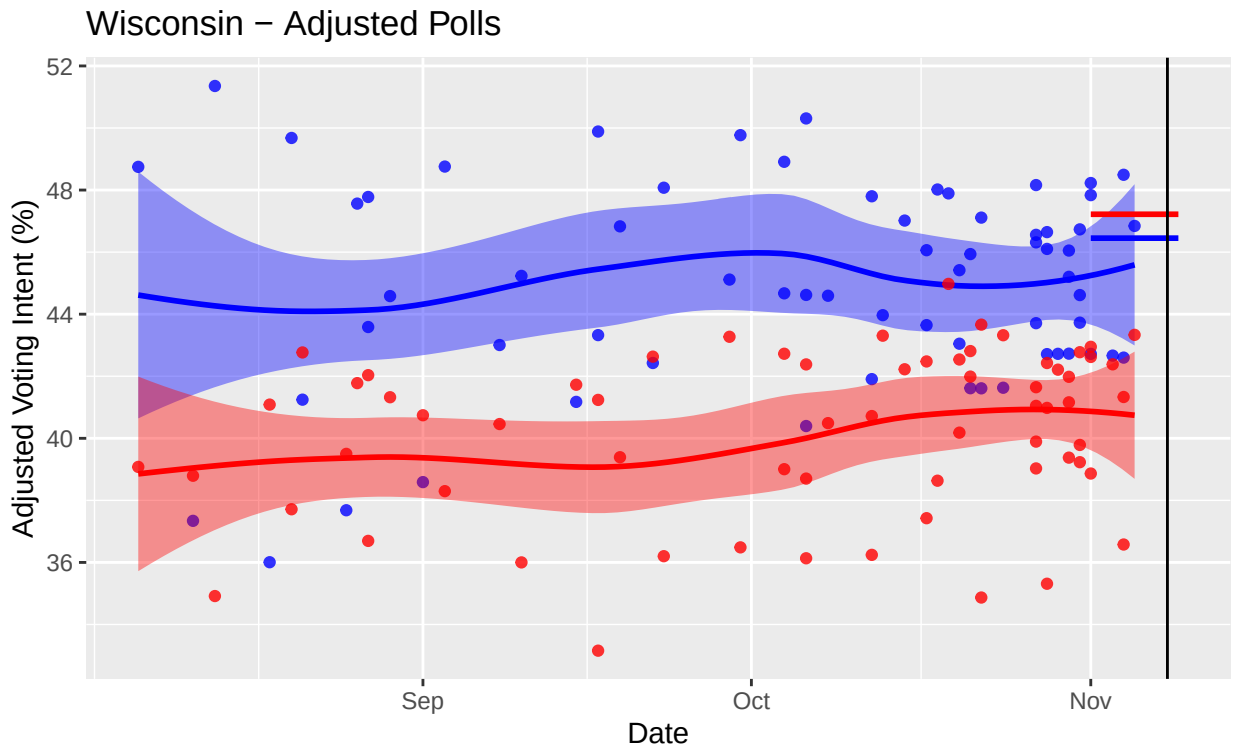

Washington − Raw Polls
Voting Intent (%)
40
20
0
Sep
Oct
Nov
Date

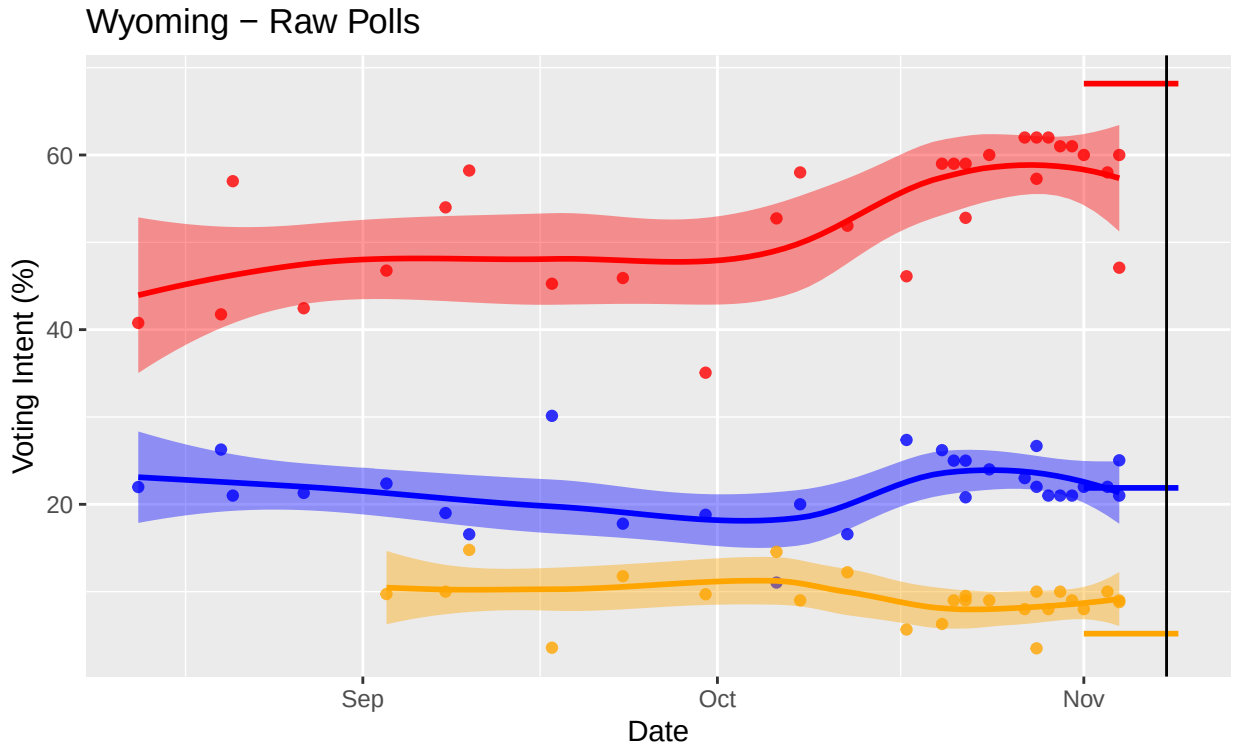

Washington − Adjusted Polls
Adjusted Voting Intent (%)
50
40
30
20
10
0
Sep
Oct
Nov
Date

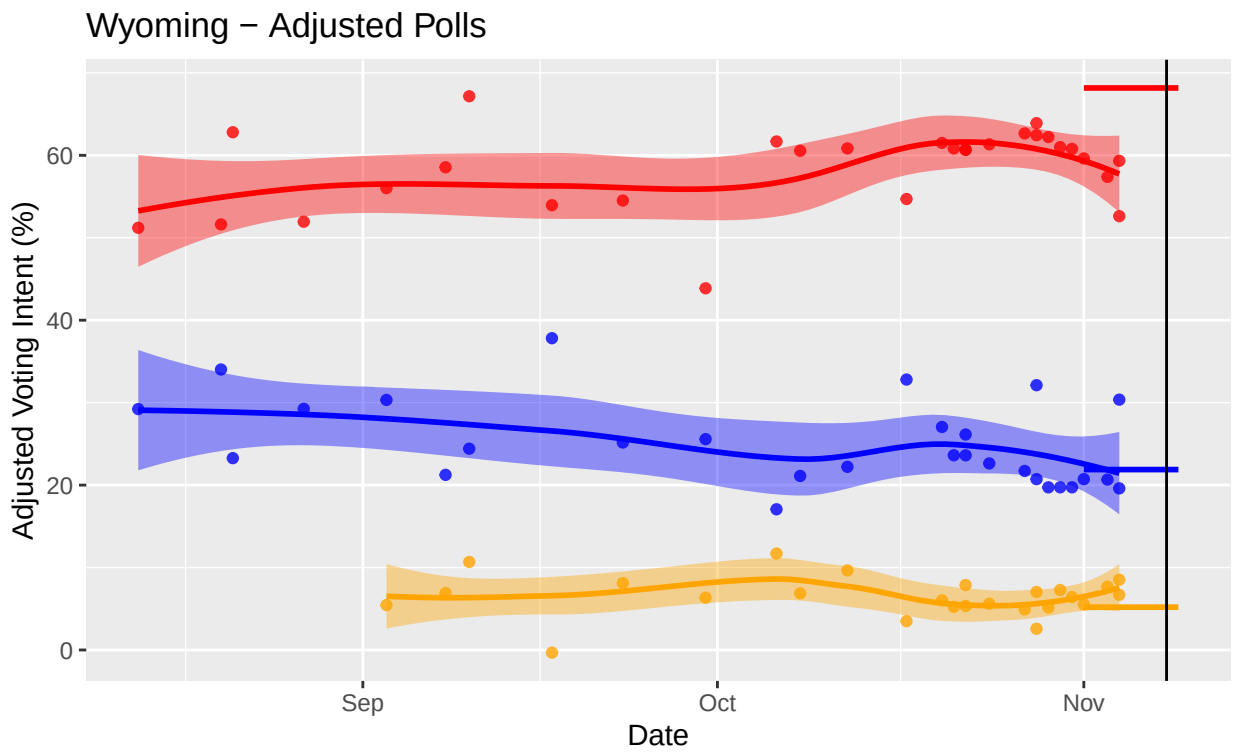

West Virginia − Raw Polls
Voting Intent (%)
70
60
50
40
30
20
Sep
Oct
Nov
Date


West Virginia − Adjusted Polls
Adjusted Voting Intent (%)
70
60
50
40
30
Sep
Oct
Nov
Date


Wisconsin − Raw Polls
Voting Intent (%)
50
40
30
Sep
Oct
Nov
Date


Wisconsin − Adjusted Polls
Adjusted Voting Intent (%)
52
48
44
40
36
Sep
Oct
Nov
Date


Wyoming − Raw Polls
Voting Intent (%)
60
40
20
Sep
Oct
Nov
Date



Maine has two congressional districts; as such, from the data of one congressional district and of the state as a whole, we can expect to reconstruct the data for the other congressional district (as long as the poll dates are relatively close together).

```
yaxes = c(0, 30)

    for (cand in c(clinton, trump, johnson)) { # get common y-limits to use on axes
        for (dist in c("Maine", "Maine CD-1", "Maine CD-2")){
            voteresult = filter(cand$finalresults, state == dist)[1,2] |>
                as.numeric()

            candpolls = cand$polls |>
                filter(state == dist)

            datevsraw = aes(
                x = candpolls$middate,
                y = candpolls$rawpolls
            )
            plot = ggplot() + geom_point(
                mapping = datevsraw,
                colour = cand$colour,
                alpha = 0.8,
                na.rm = TRUE
            ) + geom_smooth(
                mapping = datevsraw,
                colour = cand$colour,
                fill = cand$colour,
                alpha = 0.4,
                na.rm = TRUE
```

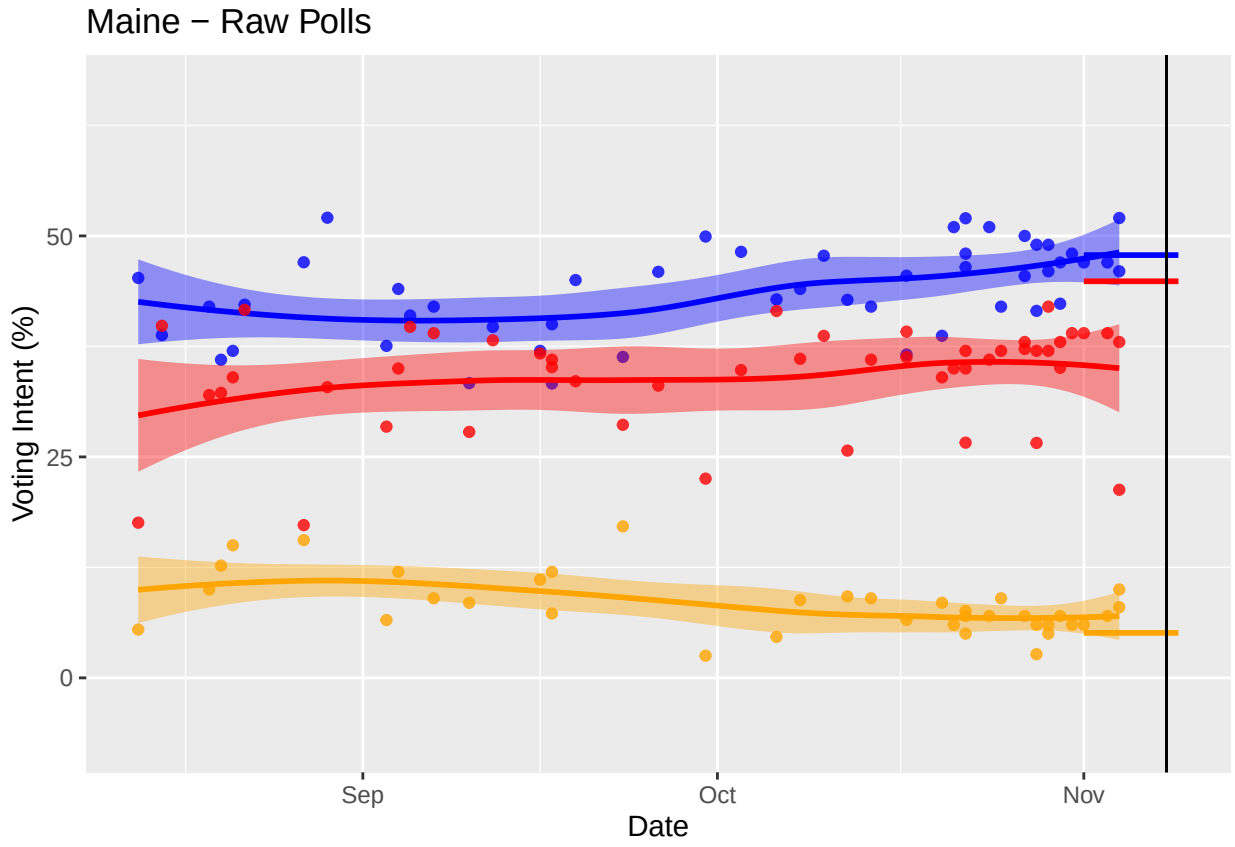
```
            ) + geom_segment(
                mapping = aes(
                x = ymd("2016-11-01"),
                y = voteresult,
                xend = dayafter,
                yend = voteresult
                ),
                colour = cand$colour,
                linewidth = 1.0
            )

        yaxes = c(min(yaxes[1], layer_scales(plot)$y$range$range[1]), max(yaxes[2], layer_scales(plot)$y
        }
    }
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

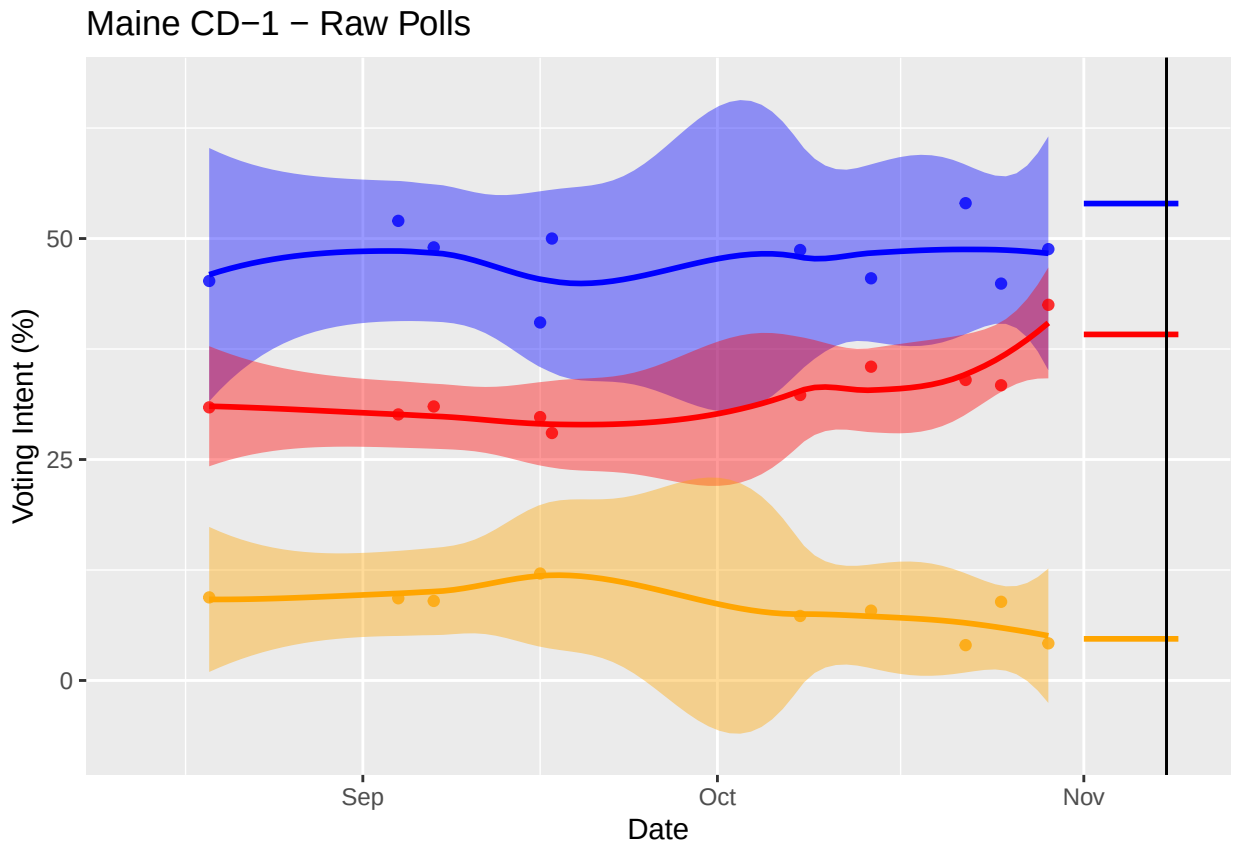
```
for (state in c("Maine", "Maine CD-1", "Maine CD-2")){
    plot = stateplotraw(state, begindate)

    stateraw = state |>
        paste("Raw Polls", sep = " - ")

    plot = plot + geom_vline(xintercept = finaldate) + labs(
        title = stateraw,
        x = "Date",
        y = "Voting Intent (%)"
    ) + xlim(begindate, dayafter)+ ylim(yaxes+c(0,2))

    print(plot)
}
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

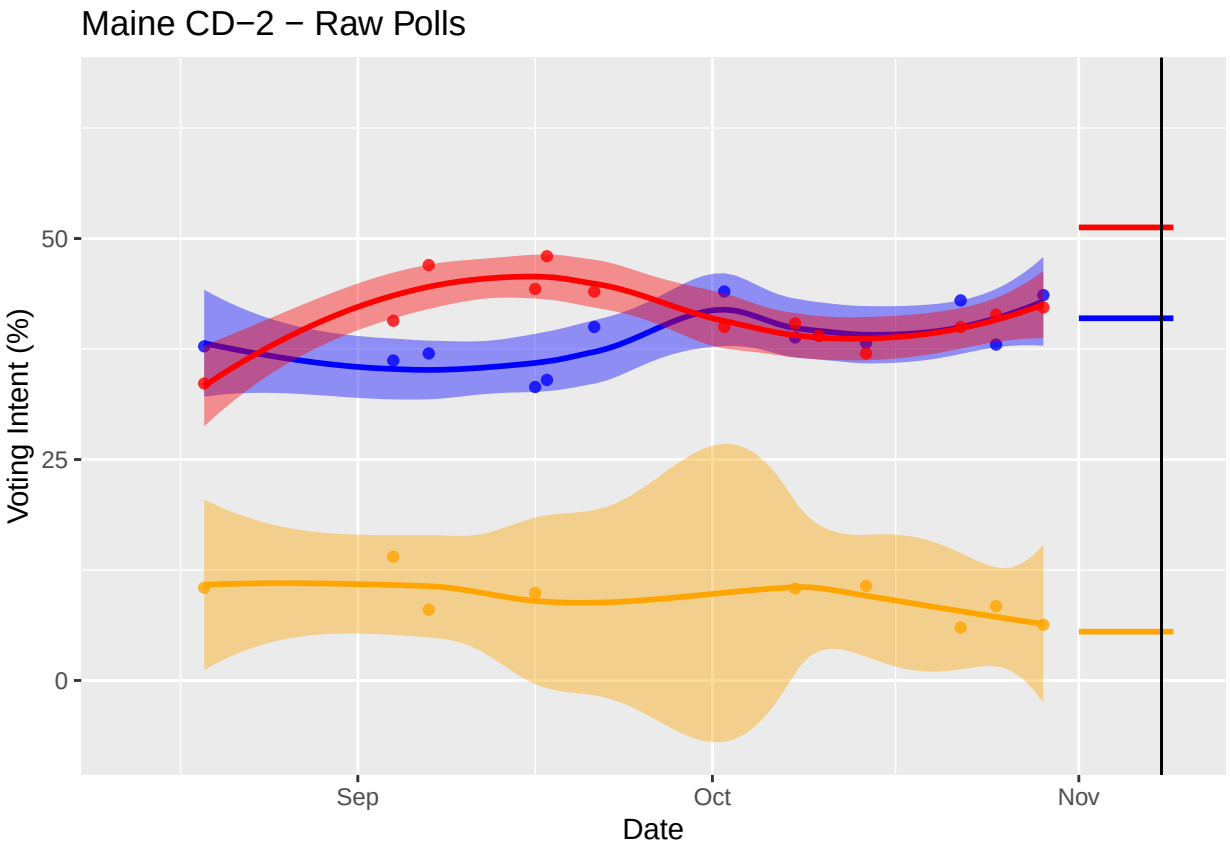
```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```




```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```




```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



Seeing as the polls for the Nebraskan congressional districts are so sparse, it's difficult to supplement the data with data from pan-Nebraskan polls – the CD polls are so few in number that they can't serve to support or refute any supplemental data. Although Clinton and Trump polled comparably in the 2nd Nebraskan congressional district, it makes the most sense (for the sake of electoral simulation) to consider Nebraska's electoral votes as a whole, rather than allocating electoral votes by district.

```
suppstates = c("Maine", "Maine CD-1", "Maine CD-2")
```