A Stochastic Model of the 2020 Presidential Election

Raphael Xie

2022-11-14

1. Part I: Introduction

1a. Problem Description

The Monte Carlo simulations are used to model the probability of outcomes in a random process that cannot easily be predicted due to the intervention of random variables. This paper introduced the Monte Carlo methods to model the 2020 Presidential Election.

I collected polls data [1] [2] 30 days before the 2020 Presidential Election from each electoral state. The following table below shows the first five rows of the proportions of voters voted for Biden and Trump in each state summarized by the poll data.

State	Prop_Biden	Prop_Trump	Seats
Alabama	0.390	0.580	9
Alaska	0.438	0.494	3
Arizona	0.480	0.458	11
Arkansas	0.350	0.603	6
California	0.617	0.323	55

For each electoral state, the candidate that received the greatest proportion of votes would take all the electoral votes from the state. There are in total of 270 electoral votes from 56 electoral states (the 50 States, District of Columbia, Nebraska District $1, 2, 3$, and Maine District $1, 2$). Candidate that received a total electoral votes greater than 270 will win the election. With the poll data, I design a model to simulate the election.

1b. Election Model

For $i = 1, 2, \ldots, 56$, let's denote B_i and T_i as the proportion of voters who opted for Biden and Trump respectively in electoral State i reported from the poll. Denote dp_i as the true difference between the proportion of voters opting Biden and the proportion of voters oping Trump in State i. Let's also define $\hat{dp}_i = B_i - T_i$, where $\hat{dp}_i$ is the difference of proportions of voters opting Biden and the proportions of voters opting for Trump in Electoral State i reported from the poll. Since that polls are collected within 30 days of the election. This means there are lots of time for people to change their mind. So it is necessary to distinguish dp_i and $\hat{dp}_i$. To improve the accuracy, we introduce a margin of error for $\hat{dp}_i$. Let's denote mr as the margin of error and assume mr to be a fixed parameter. Then we assume in State i, the true difference

[1]The poll data can be accessed from the link: https://www.270towin.com/2020-polls-biden-trump/. The website summarized and averaged polls for the 2020 presidential election from different electoral states.

[2]Since there is no qualifying data from Nebraska District 1 and Nebraska District 3 available, I set the proportions for each candidates to 0.5 in both districts.

of proportions of votes between Biden and Trump, dp_i, would be uniformly distributed between $\hat{dp_i} - mr$ and $\hat{dp_i} + mr$.

In the simulations, for $i = 1, 2, \ldots, 56$, we firstly generate a random number from a uniform distribution with parameters $\hat{dp_i} - mr$ and $\hat{dp_i} + mr$. Let's denote this generated number as ${dp_i}^*$. If $dp_i^* > 1$, then Biden takes all the electoral votes from the Electoral State i. For example, the $2-$nd row of the table shows that in Alaska, the proportion of voters opting for Biden is 0.438 and the proportion of voters opting for Trump is 0.494. Then $\hat{dp_2} = 0.438 - 0.494 = -0.056$. We assume a margin of error $mr = 0.12$. Then we randomly generate ${dp_2}^*$ from $Unif[-0.064, 0.24]$. If the generated ${dp_2}^* > 0$, then we assume Biden would take all the 3 electoral votes from Alaska.
If the generated ${dp_2}^* < 0$, then we assume Trump would take all the 3 electoral votes from Alaska.
If the generated ${dp_2}^* = 0$, then there is a tie between Biden and Trump in Alaska.
In this case, we will regenerate a value of ${dp_2}^*$ until ${dp_2}^* \neq 0$.

With computer software, I can generate a random value of ${dp_i}^*$ for each electoral State i, where $i = 1, 2, \ldots, 56$. Then I checked the value of ${dp_i}^*$ and add the electoral votes to the winner of the state i. After we add all the electoral votes, I compare the total electoral votes to determine the winner of the election. By using Monte Carlo simulations, I can estimate the probability of wining the election for both candidates and the number of electoral votes they would received.

2. Monte Carlo Simulation

2a. Process

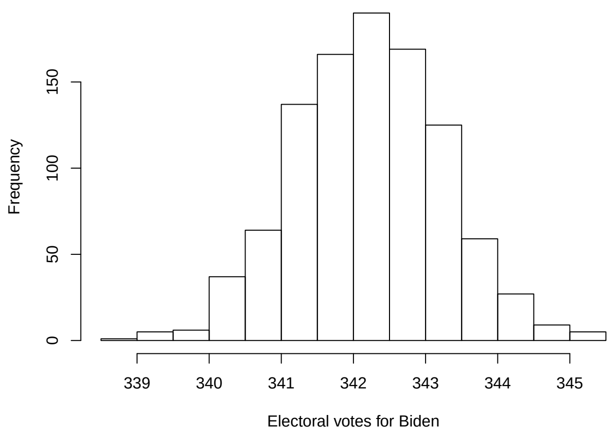
To estimate the probability of winning and the number of electoral votes for both candidates, I firstly assume a margin of error $mr = 0.12$. (We will experiment other values of mr in the next section). Let's denote π_B and π_T as the expected probability of winning the election for Biden and Trump respectively. Denote $\hat{\pi_B}$ and $\hat{\pi_T}$ as the estimate of the expected probability of winning the election for Biden and Trump respectively. Let also denote V_B and V_T as the expected total number of electoral votes received by Biden and Trump respectively. Denote $\hat{V_B}$ and $\hat{V_T}$ as the estimate of the expected total number of electoral votes received by Biden and Trump respectively. Then I use computer software to perform 1000 trials of simulations. For each trial, there are in total of 1000 simulations of election. After each simulation, I keep track of the number of times of winning and the total electoral votes for both candidates. After performing 1000 simulations, I use the ratio

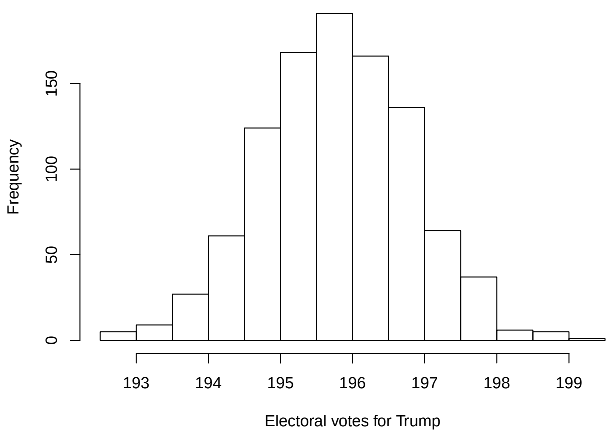
$$\frac{\text{number of times that Biden wins}}{1000}, \text{and } \frac{\text{number of times that Trump wins}}{1000}$$

as an empirical estimate of the expected probability of winning the election for Biden, $\hat{\pi_B}$, and Trump, $\hat{\pi_T}$ respectively. And I use the averaged total electoral votes received by each candidate as the estimate of the actual electoral votes for each candidate. After performing 1000 trials, I plot the results. (All the R codes can be viewed at the Appendix).

2b. Plot

The following is the histogram for the estimated electoral votes for Biden and Trump.





Based on the plot, we could notice that the estimated electoral votes for Biden and Trump centered at about 342 and 195 respectively. The estimated electoral votes for Biden and Trump both appear to be approximately normally distributed.
In normal distributions, we know that about 99.73% of the values of the population will lie within 3 standard deviations of the population mean. Using `R`, we can solve that the mean value of electoral votes for Biden and Trump is 342.2171 and 195.7829. The standard deviation of the estimated electoral votes for Biden and Trump is 1.0188 and 1.0188 respectively. This makes sense since the total votes sum up to 538 when a tie does not exist, and it is very unlikely to witness a tie. Then we may calculate the 99.73% confidence interval of the electoral votes

for Biden,

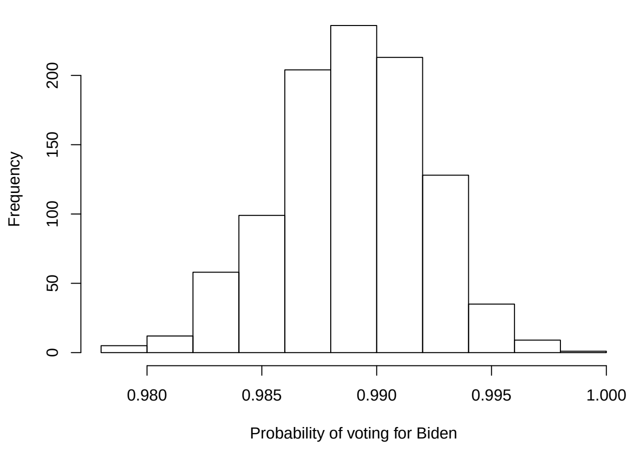
$$[342.2171 - 3 \cdot 1.0188, 342.2171 + 3 \cdot 1.0188] = [339.1607, 345.2735]$$

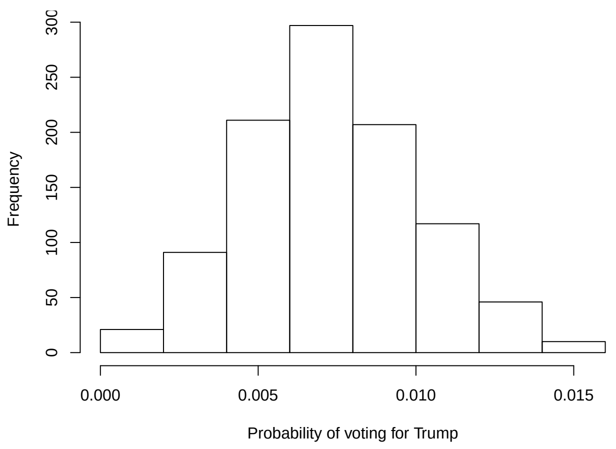
for Trump,

$$[195.7829 - 3 \cdot 1.0188, 195.7829 + 3 \cdot 1.0188] = [192.7265, 198.8393]$$

Based on the calculation, we can conclude that we have 99.73% confidence that the expected number of electoral votes for Biden V_B will be between 339.1607 and 345.2735. We have about more than 99.73% confidence that the expected number of electoral votes for Trump V_T will be between 192.7265 and 198.8393. Besides, we also noticed that the standard deviation of the estimated electoral votes for Biden and Trump is the same. This makes sense because we know that for a random variable X and an arbitrary constant c, we have $Var[X] = Var[c - X]$. Here, the votes for Biden and the votes for Trump both add up to 538. That is, $V_B = 538 - V_T$. Thus we have $Var[V_B] = Var[538 - V_B] = Var[V_T]$. While standard deviation is the square root of the variance, so this shows that our standard deviations of electoral votes for both candidates are equal.

Now let's focus on the estimated probability of winning for Biden, $\hat{\pi_B}$, and Trump, $\hat{\pi_T}$. The following is the histogram for $\hat{\pi_B}$ and $\hat{\pi_T}$.





From the plot, we could observed that the estimated probability of winning for Biden, $\hat{\pi_B}$, and Trump, $\hat{\pi_T}$, centered at about 0.988 and 0.075 respectively. We also noticed that both $\hat{\pi_B}$ and $\hat{\pi_T}$ appear to be normally distributed, even though the histogram for Trump shows a tiny right-skewness. We also apply the Central Limit Theorem to the estimated probability of winning. Using `R`, we can solve that the mean value of electoral votes for Biden and Trump is 0.9894 and 0.0078. The standard deviation of the 1000 estimated electoral votes for Biden and Trump is 0.0028 and 0.0032 respectively. Then we may calculate the 99.73% confidence interval of the electoral votes

for Biden,

$$[0.9894 - 3 \cdot 0.0028, 0.9894 + 3 \cdot 0.0028] = [0.981, 0.9978],$$

for Trump,

$$[0.0078 - 3 \cdot 0.0032, 0.0078 + 3 \cdot 0.0032] = [-0.0018, 0.0174]$$

We noticed that the confidence interval of the expected probability of winning the election for Trump, π_T, include negative values. However, probability must take nonzero value, so we need to remove the negative values. Then we could conclude that we have about 99.73% confidence that the expected probability that Trump wins the election, π_T, will be in the range $[0, 0.0174]$. For Biden, we have about 99.73% confidence that the expected probability of winning the election, π_B, will be in the range $[0.981, 0.9978]$.

2c. Compare with reality

Based on the results of the simulations, we noticed a great discrepancy between the estimated total electoral votes and the actual electoral votes received by both candidates. As we conclude from the previous section, we have at least 99.73% confidence interval of the expected electoral votes for Biden, V_B, is between 339.1607 and 345.2735. While in 2020 Presidential Election, Biden gained a total of 306 electoral votes and Trump gained 232 electoral votes. The simulated results overestimated more than thirty votes from the actual electoral votes received by Biden. There are many possible factors can lead to this discrepancy. For example, the poll samples may not be representative. Some likely voters may also changed their mind after taking the survey. The survey itself could also be biased. Besides, we should keep in mind that the elections are simulated based on a fixed margin of error, $mr = 0.12$. We will investigate how the results would change if we increase or decrease the margin of error.

3. Other Margin of Errors

3a. An easier method

To investigate the influence of margin of error to our model, I repeat Monte Carlo simulations with margin of error $mr = 0.04, 0.08, 0.12$ and 0.16. However, calculating the mean and standard deviation is too time-consuming. An easier and simple method is to do 10 runs for each margin of error, in which each run has 10000 simulations. Then we sort the ten estimates and apply a weak version of Central Limit Theorem. By using the fact that normal distribution implies symmetric about the true mean, we may estimate the probability that all of the 10 estimates are on one side of the true estimates as

$$\frac{2}{2^{10}} \approx 0.00195.$$

This means the probability that all of our 10 estimates are greater or less than the true population mean is approximately 0.00195. In other word, we the probability that the true population mean will lie between our 10 estimates is

$$1 - \frac{2}{2^{10}} \approx 0.998.$$

Thus, we may conclude that we have about 99.8% confidence that the population mean will be between the lowest and the highest value of our estimates. Let me use the simulation results with margin of error $mr = 0.08$ as an example.

The following is the table that summarised the estimated probability of winning and the estimated total number of electoral votes for Biden and Trump.

Est_Prob_Biden	Est_Prob_Trump	Est_Votes_Biden	Est_Votes_Trump
0.9897	0.0086	342.2981	195.7019
0.9919	0.0067	342.2242	195.7758
0.9886	0.0085	341.8075	196.1925
0.9881	0.0096	342.0179	195.9821
0.9886	0.0083	341.8910	196.1090
0.9900	0.0079	341.9344	196.0656
0.9897	0.0081	341.8636	196.1364
0.9876	0.0099	341.6422	196.3578
0.9895	0.0077	342.2229	195.7771
0.9898	0.0077	342.5061	195.4939

From the table, we found that all the estimated probability of that Biden wins the election, $\hat{\pi_B}$, are greater than 0.98 while all the estimated probability of winning the election for Trump ,$\hat{\pi_T}$, is less than 0.01. We also noticed that $\hat{\pi_B} + \hat{\pi_T} \neq 1$. This happens when there is a tie between two candidates. That is, both Biden and Trump received 269 electoral votes.

Based on the method we explained above, we assume a weak version of the Central Limit Theorem. From the results, we can conclude with about 99.8% confidence that the expected number of electoral votes received by Biden is between `341.6422` and `342.5061`; and the expected probability for Biden to win the election is between `0.9876` and `0.9919`.

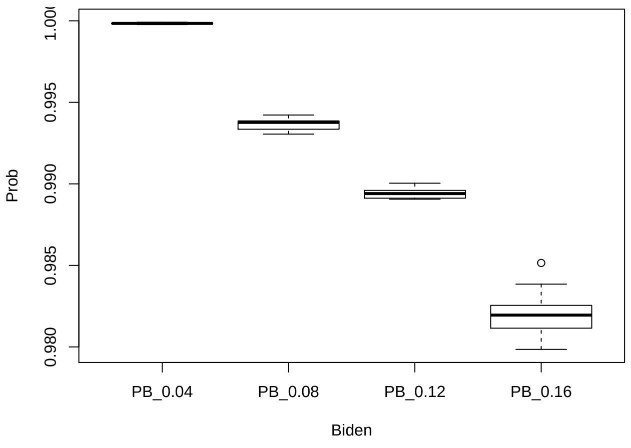
We can also have about 99.8% confidence that the expected number of electoral votes received by Trump is between `195.4939` and `196.3578`, and the expected probability for Trump to win the election is between `0.0067` and `0.0099`.

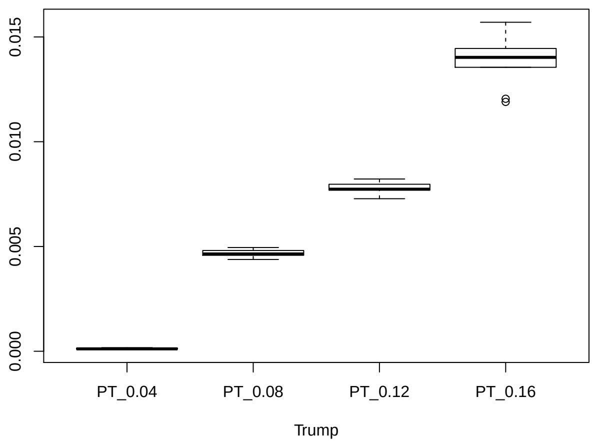
3b. Table

The following table summarised the sorted estimated probability of wining the election for both candidates with $mr = 0.04, 0.08, 0.12,$ and 0.16. Note that I use "$PB_{0.04}$" as the abbreviation for "probability of winning for Biden when we set $mr = 0.04$", "$PT_{0.04}$" as the correspond for Trump.

PB_0.04	PB_0.08	PB_0.12	PB_0.16	PT_0.04	PT_0.08	PT_0.12	PT_0.16
0.99977	0.99305	0.98906	0.97985	0.00009	0.00438	0.00728	0.01190
0.99979	0.99334	0.98908	0.98095	0.00009	0.00450	0.00759	0.01205
0.99982	0.99335	0.98912	0.98115	0.00010	0.00458	0.00769	0.01355
0.99983	0.99350	0.98929	0.98130	0.00010	0.00461	0.00770	0.01395
0.99984	0.99377	0.98938	0.98185	0.00011	0.00463	0.00772	0.01400
0.99984	0.99377	0.98943	0.98205	0.00012	0.00466	0.00775	0.01405
0.99985	0.99377	0.98956	0.98225	0.00013	0.00479	0.00783	0.01405
0.99986	0.99386	0.98960	0.98255	0.00015	0.00481	0.00797	0.01445
0.99988	0.99390	0.98963	0.98385	0.00015	0.00491	0.00811	0.01475
0.99991	0.99422	0.99004	0.98515	0.00017	0.00495	0.00822	0.01570

Based on the table, we observed that the estimated probability for Biden to win, $\hat{\pi_B}$, decreases as the margin of error increases. Also, for the ten estimates of probability for Biden to win, $\hat{\pi_B}$ are all greater than 0.975, which is extremely close to 1. At the same time, we found that all the estimated probability for Trump to win, $\hat{\pi_T}$, are all smaller than 0.01 despite the case when margin of error is 0.16. This makes sense since a greater margin of error could bring greater potential of randomness to the results. To visualize how the $\hat{\pi_B}$ and $\hat{\pi_T}$ changed as we changed the value of mr, I use a boxplot to compares the probability of winning for both candidates with margin of error 0.04, 0.08, 0.12, and 0.16. The following is a boxplot that compare the estimated probability for each candidate to win, when simulating with different mr.



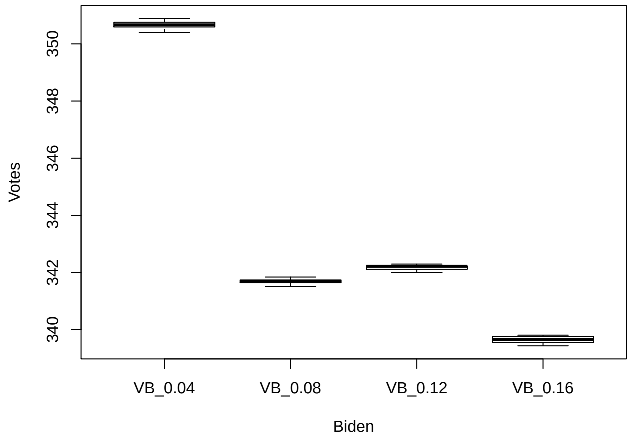


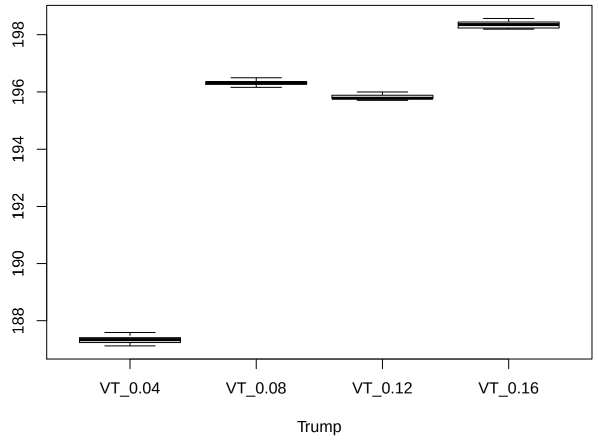
From the boxplot, we could observed that all the boxes are very narrow. When we set $mr = 0.04$, the boxplots for $\hat{\pi_B}$ and $\hat{\pi_T}$ are displayed as a thick line. This indicates an extremely small interquartile range. For $mr = 0.04, 0.08, 0.12,$ and 0.16, the estimated probability for Biden to win,$\hat{\pi_B}$, decreased as mr increased. On the contrary, the estimated probability for Trump to win, $\hat{\pi_T}$, increased as mr increased. Though we may continue to increase the value of margin of error, in practice, polls with large margin of errors are usually not reliable and accurate.

The following table summarises the estimated electoral votes Biden and Trump received, with margin of error 0.08, 0.12, and 0.16. Note that in the table, I used "$VB_{0.04}$" as an abbreviation for "estimated electoral votes for Biden when we set $mr = 0.04$", "$VT_{0.04}$" as the correspond for Trump.

VB_0.04	VB_0.08	VB_0.12	VB_0.16	VT_0.04	VT_0.08	VT_0.12	VT_0.16
350.4056	341.5062	342.0009	339.4333	187.1193	196.1591	195.7055	198.1933
350.4321	341.5576	342.0810	339.5240	187.1337	196.2362	195.7529	198.2209
350.5919	341.6404	342.1138	339.5563	187.2415	196.2611	195.7712	198.2346
350.6340	341.6406	342.1405	339.5668	187.2606	196.2684	195.7781	198.3161
350.6587	341.6625	342.2170	339.6225	187.3302	196.3094	195.7784	198.3285
350.6698	341.6906	342.2216	339.6715	187.3413	196.3375	195.7830	198.3775
350.7394	341.7316	342.2219	339.6839	187.3660	196.3594	195.8595	198.4332
350.7585	341.7389	342.2288	339.7654	187.4081	196.3596	195.8862	198.4437
350.8662	341.7638	342.2471	339.7790	187.5679	196.4424	195.9190	198.4760
350.8807	341.8409	342.2945	339.8067	187.5944	196.4938	195.9991	198.5667

The following boxplot summarised the electoral votes for Biden and Trump with margin of errors from 0.08 to 0.16.





From the boxplot, we found that among all the estimated number of electoral votes, $\hat{V_B}$ has the greatest value and $\hat{V_T}$ has the lowest value when we set $mr = 0.16$. We also observed that all the range of estimated number of electoral votes for Biden, $\hat{V_B}$, and Trump, $\hat{V_T}$, are very narrows. Combined the table with the boxplot, we find that there is no overlap between each confidence interval. Among the range of the averaged votes received by Biden, the lower bound is 339.43 and the upper bound is 350.88. These tables and boxplots show that the value of margin of error has very great influence to the simulated results. Setting the values of margin of error are crucial to the results of simulations.

Different from the estimated probability for Biden to win the election, here, $\hat{V_B}$ firstly decreased from $mr = 0.04$ to $mr = 0.08$, and then increased from $mr = 0.08$ to $mr = 0.12$, and then decreased. This reflected the randomness of Monte Carlo simulations. While from the previous boxplot, we found that the probability of winning for Biden decreased as margin of error increased. This shows that higher estimated probability of winning the elections does not imply greater expected number of electoral votes received.

Recall from the reality, Biden and Trump 306 and 232 electoral votes in 2020 Presidential Election respectively. There is a great discrepancy between the results and reality. Though we may increase the margin of error to manually make the resulted votes closer to the reality, we should keep in mind that polls with a large margin of error are not reliable in reality, and we should never use these polls.

The discrepancy can be influenced by various factors, including but not limited to unrepresentative sample size, sampling bias, uniform distribution, and even bias in the poll experts... To minimize the influence of unrepresentative sample size, we shall discuss the techniques in the next section.

4. Reelection

4a. Using real election results

Based on the simulations in previous section, we also noticed that all the simulated results overestimate the electoral votes for Biden. This great discrepancy made me question the accuracy of the polls. What will the results be if the poll is very close to the real election results? Let's put it into an extreme case, can we just use the statistics from real election as the poll? In this case, what we do is using the results[3] of 2020 Presidential Election as the poll to "reelect" the president.

The following table is the results from the 2020 Presidential Election. In the table, the attribute `"dem_this_margin"` represents the proportion of votes for Biden minus the proportion of votes for Trump. Note that I only shows the first fifteen rows, which summarised the fifteen swing states in the Election.

state	dem_percent	rep_percent	EV	dem_this_margin
Arizona	0.494	0.491	11	0.003
Florida	0.479	0.512	29	-0.034
Georgia	0.495	0.493	16	0.002
Iowa	0.449	0.531	6	-0.082
Maine 2nd District	0.448	0.523	1	-0.074
Michigan	0.506	0.478	16	0.028
Minnesota	0.524	0.453	10	0.071
Nebraska 2nd District	0.522	0.457	1	0.065
Nevada	0.501	0.477	6	0.024
New Hampshire	0.527	0.454	4	0.074
North Carolina	0.486	0.499	15	-0.013
Ohio	0.452	0.533	18	-0.080
Pennsylvania	0.500	0.488	20	0.012
Texas	0.465	0.521	38	-0.056
Wisconsin	0.494	0.488	10	0.006

Based on the `dem_this_margin` column, we found that many of the swing states has a small difference of proportion of votes, dp. For example, in Georgia, $dp = 0.002$, which is very small. With this small proportion, Biden took all the 15 electoral votes from Georgia. If we set a margin of error greater than 0.002, then the results of estimated total votes that Biden received will change. Note that including the margin of error is necessary because it takes into account the chance that some voters may change their mind and vote for others.

In this section, I repeat the procedures in `Section 3` to estimate the probability of winning and total electoral votes for Biden and Trump. I simulated the results with margin of error $mr = 0.02, 0.04, \ldots, 0.12$

4b. Results

By Monte Carlo method, I use R to simulate 10 runs, in which one run has 10000 simulations. The following tables summarised the estimates of expected probability of winning the election for Biden and Trump, with $mr = 0.02, 0.04, \ldots, 0.12$ based on the real election results.

PB_0.02	PB_0.04	PB_0.06	PB_0.08	PB_0.10	PB_0.12
0.8612	0.76527	0.75674	0.77383	0.79722	0.80450
0.8622	0.76592	0.75740	0.77395	0.79742	0.80562

[3]The data can be accessed from the link:https://docs.google.com/spreadsheets/d/e/2PACX-1vS3Z8Rq9xqOLISwoKdK0n6CFLBuPSCoXbbLeY8vhi-rzFS3ZFNEtR0BCdEbHcS-2Tlh5aPcnZbwBLao/pub?output=csv

PB_0.02	PB_0.04	PB_0.06	PB_0.08	PB_0.10	PB_0.12
0.8640	0.76598	0.75747	0.77449	0.79898	0.80595
0.8641	0.76674	0.75809	0.77509	0.79938	0.80645
0.8644	0.76678	0.75826	0.77573	0.79966	0.80710
0.8656	0.76693	0.75836	0.77604	0.80038	0.80739
0.8656	0.76722	0.75846	0.77620	0.80082	0.80740
0.8660	0.76743	0.75887	0.77621	0.80137	0.80816
0.8674	0.76883	0.75942	0.77696	0.80253	0.80824
0.8694	0.76904	0.76029	0.77726	0.80381	0.80861

PT_0.02	PT_0.04	PT_0.06	PT_0.08	PT_0.10	PT_0.12
0.0576	0.15435	0.17356	0.16745	0.15556	0.15903
0.0596	0.15464	0.17367	0.16826	0.15653	0.15911
0.0605	0.15612	0.17375	0.16843	0.15803	0.15975
0.0606	0.15656	0.17392	0.16862	0.15809	0.16026
0.0626	0.15665	0.17398	0.16916	0.15824	0.16053
0.0630	0.15673	0.17404	0.16970	0.15875	0.16092
0.0640	0.15717	0.17444	0.17006	0.15913	0.16092
0.0653	0.15753	0.17454	0.17033	0.15930	0.16124
0.0656	0.15766	0.17465	0.17098	0.16111	0.16133
0.0659	0.15784	0.17562	0.17118	0.16127	0.16206

From the table, we found that the estimates of expected probability for Biden to win, $\hat{\pi_B}$ is the greatest when we set $mr = 0.02$. We also noticed that there are overlapping confidence intervals. For the estimates with $mr = 0.04$ and 0.06, the confidence intervals for π_B overlap with each other. The confidence interval of π_T with 0.10 also overlaps with the confidence interval of π_T with $mr = 0.04$ and $mr = 0.12$. Among all the values of mr from 0.02 to 0.12, we found that the values of $\hat{\pi_B}$ ranged from `0.75674` to `0.8694`. This also wide range can show that different values of margin of error can produce very different results in simulations. When $mr = 0.02$, the confidence interval of π_T are the highest among other confidence intervals of π_T. From $mr = 0.06$ to 0.12, we found that the confidence interval for π_T increases as mr increases. As the same time, we also noticed that for the same value of mr, the sum of any one value of $\hat{\pi_T}$ and any one value of $\hat{\pi_B}$ can not be 1. This happens when both Biden and Trump get 269 electoral votes. Now, let's shift our focus from the estimates of expected probability of winning to the estimates of the electoral votes.

The following tables summarised the estimated number of electoral votes for Biden and Trump, with $mr = 0.02, 0.04, \ldots, 0.12$ based on the real election results.

VB_0.02	VB_0.04	VB_0.06	VB_0.08	VB_0.10	VB_0.12
289.0138	285.6365	287.7350	291.8679	295.7645	297.5399
289.0883	285.6405	287.8058	291.9580	295.8005	297.5702
289.1310	285.6416	287.8335	291.9594	295.8406	297.5817
289.1392	285.6488	287.8530	291.9620	295.8495	297.6145
289.2568	285.6675	287.8845	291.9688	295.9124	297.6358
289.2707	285.6916	287.8932	291.9987	295.9227	297.6530
289.3393	285.7235	287.9297	292.0283	296.0177	297.6763
289.3762	285.7250	287.9339	292.0542	296.0226	297.7157
289.3812	285.7341	287.9530	292.0590	296.0272	297.7228
289.4151	285.7705	287.9998	292.1414	296.0597	297.7507

VT_0.02	VT_0.04	VT_0.06	VT_0.08	VT_0.10	VT_0.12
248.5849	252.2295	250.0002	245.8586	241.9403	240.2493
248.6188	252.2659	250.0470	245.9410	241.9728	240.2772
248.6238	252.2750	250.0661	245.9458	241.9774	240.2843
248.6607	252.2765	250.0703	245.9717	241.9822	240.3237
248.7293	252.3084	250.1068	246.0013	242.0773	240.3470
248.7432	252.3325	250.1155	246.0312	242.0876	240.3642
248.8608	252.3512	250.1470	246.0380	242.1505	240.3855
248.8690	252.3584	250.1665	246.0406	242.1594	240.4183
248.9117	252.3595	250.1942	246.0420	242.1995	240.4298
248.9862	252.3635	250.2650	246.1321	242.2355	240.4601

Compared the table with the results in `Section 2` and `3`, we found that the results in this section are much closer to the reality. (Recall that Biden and Trump gained 306 and 232 electoral votes in 2020 Presidential Election respectively). This shows that the accuracy of polls can have great influence on the accuracy of the estimation.

When $mr = 0.02$, the ten estimates of expected electoral votes for Biden, $\hat{V_B}$ is larger then the ten values of $\hat{V_B}$ with mr set as 0.04. When $mr = 0.04$, the confidence intervals for V_B overlaps with the confidence interval for V_B with $mr = 0.06$. Then from $mr = 0.06, 0.08, \ldots, 0.12$, there are no overlaps between the confidence interval for V_B. And we found that the value of the expected electoral votes V_B increases as mr increases during this range. From $mr = 0.02$ to 0.04 and 0.06, we found that the expected number of electoral votes for Biden decreases. This shows that increasing the margin of error mr can bring more chances for some Biden voters to change their minds and vote to Trump. From $mr = 0.04$ and 0.06 to 0.12, we found that the expected number of electoral votes for Biden increases. This shows that increasing the margin of error can also bring more chances for other Trump voters to change their mind and vote for Biden.

Based on the table, we also found that all the estimates of the expected electoral votes for Biden, $\hat{V_B}$, are smaller than 298, and all the expected electoral votes for Trump, $\hat{V_T}$, are greater than 240. This shows that the simulated results underestimate the total electoral votes for Biden and overestimate the expected total electoral votes for Trump. Since none of the confidence interval for the expected electoral votes for Biden include 306, and none of the confidence interval for the expected electoral votes for Trump include 232, we still can not conclude that our new model is consistent with the reality.

Based on the result, the author believe that Uniform distribution may not be a workable model for election. I would recommend other probability distribution such as normal distribution.

5. Appendix

The poll can be accessed from the link: https://www.270towin.com/2020-polls-biden-trump/. The results of the election can be accessed from the link: https://docs.google.com/spreadsheets/d/e/2PACX-1vS3Z8Rq9xqOLISwoKdK0n6CFLBuPSCoXbbLeY8vhi-rzFS3ZFNEtR0BCdEbHcS-2Tlh5aPcnZbwBLao/pub?output=csv. The following is the R code that simulates the elections from the polls.

```r
library(knitr)
library(tidyverse)

# Preparing and cleaning the data
dat <- read.csv("/Users/mac/Desktop/PresidentialElection/2020polls_MC.csv")
head(dat, 5)

diff_prop <- dat$Prop_Biden - dat$Prop_Trump
polls_dat <- cbind(dat, diff_prop)
```

```r
head(polls_dat, 5)

# Function to randomly generate a value uniformly distributed
#   with specified input parameters
# Return the generated value
sim_adj_diff <- function(min_diff, max_diff) {

  adj_diff <- runif(n = 1, min = min_diff, max = max_diff)

  # Assume the State will rerun the election when a tie happens
  while (adj_diff == 0) {
    adj_diff <- runif(n = 1, min = min_diff, max = max_diff)
  }

  return (adj_diff)
}


# Function to set up the original poll data based on the margin of error.
# This greatly improved the computational efficiency
new_poll_df <- function(poll_df, mr_error) {

  # Calculate the lower and upper bound of the proportion of votes in the poll
  min_diff <- poll_df$diff_prop - mr_error
  max_diff <- poll_df$diff_prop + mr_error
  poll_with_mr <- cbind(poll_df, min_diff, max_diff)

  return (poll_with_mr)
}


# Function to simulate an election in all electoral States
# Return an array of votes received by both candidates
sim_vote_usa <- function(poll_with_mr) {
  count_B_votes <- 0
  count_T_votes <- 0

  # For each State, simulate an election
  for (i in 1:dim(poll_with_mr)[1]) {
    min_unif <- poll_with_mr$min_diff[i]
    max_unif <- poll_with_mr$max_diff[i]

    # Check if the electoral State has strong partisanship
    # If the proportion of votes received by Biden adjusted by margin of error
    #   still indicate no chance for Biden to win the State (max_unif < 0),
    #   then Trump wins the State directly.
    #   This reduced computational time
    if (max_unif < 0) {
      count_T_votes <- count_T_votes + poll_with_mr$Seats[i]
    } else if (min_unif > 0) {
      count_B_votes <- count_B_votes + poll_with_mr$Seats[i]
    } else {
      # Assume the adjusted difference in proportions to be uniformly distributed
      #   in [prop_B - prop_T margin of error, prop_B - prop_T + margin of error]
      adj_diff <- sim_adj_diff(min_unif, max_unif)

      # We assume Biden wins a State if adj_diff > 0
      # Note that adj_diff is a non-zero value
      if (adj_diff > 0) {
        count_B_votes <- count_B_votes + poll_with_mr$Seats[i]
      } else if (adj_diff < 0) {
        count_T_votes <- count_T_votes + poll_with_mr$Seats[i]
      }
    }
  }

  return (array(c(count_B_votes, count_T_votes)))
}


# Set seed to ensure the results can be reproducible
set.seed(538)

# Number of iterations per trial
N <- 10000

# Total number of trials
total_trial <- 10

# Function to perform 10 trials, with each trial has
#   10000 Monte Carlo simulations
# Return a data frame that record the averaged probability
#   of winning and the averaged total votes for two candidates
MC_simulate <- function(poll_df, mr_err) {

  # Step 1: Getting the Poll Data Frame with lower and upper bound
  poll_with_mr <- new_poll_df(poll_df, mr_err)

  # Matrix to record the averaged probability of winning and
  #   the averaged total votes for two candidates
  montecarlo_results <- matrix(nrow = total_trial, ncol = 4)

  # Step 2: For each trail, perform 10000 simulations of Presidential Election
  #         Keep track of the winning times of both candidates
  for (i in 1:total_trial) {

    # Initialize the count of winning times for both candidates
    count_B_trial <- 0
    count_T_trial <- 0
    count_tie_trial <- 0

    # Array to record the total votes in each simulation
    record_biden_votes <- array()
    record_trump_votes <- array()
```

```r
    # Step 3: For each Electoral State, simulate an election;
    #         Record the votes of both candidates;
    #         Repeat for 10000 times
    for (j in 1:N) {
      temp_election <- sim_vote_usa(poll_with_mr)

      # Winner needs to have more than 270 electoral votes
      if (temp_election[1] > 270) {
        count_B_trial <- count_B_trial + 1
      } else if (temp_election[2] > 270) {
        count_T_trial <- count_T_trial + 1
      } else {
        # A tie
        count_tie_trial <- count_tie_trial + 1
      }

      # Arrays to record the total votes received from each simulation
      record_biden_votes <- append(record_biden_votes, temp_election[1])
      record_trump_votes <- append(record_trump_votes, temp_election[2])
    }

    # Step 4: Calculate the averaged probability of winning for both candidates
    montecarlo_results[i, 1] <- count_B_trial / N
    montecarlo_results[i, 2] <- count_T_trial / N
    montecarlo_results[i, 3] <- mean(record_biden_votes, na.rm = TRUE)
    montecarlo_results[i, 4] <- mean(record_trump_votes, na.rm = TRUE)

    # Rename the columns
    colnames(montecarlo_results) <- c("Est_Prob_Biden", "Est_Prob_Trump",
                                      "Est_Votes_Biden", "Est_Votes_Trump")
  }

  # Return the resulted data frame
  montecarlo_results <- as.data.frame(montecarlo_results)

  return(montecarlo_results)
}


############################################################################
###  The following code simulates a Monte Carlo election with MR = 0.16  ###
############################################################################

# Initialize the parameters, margin of error = 0.16
mr_err <- 0.16
example_mr_016 <- new_poll_df(polls_dat, mr_err)
```

The resulted data frame includes the lower and upper bound of proportion of votes. Since the function $MC_simulate(mr_err)$ already include the code, we don't need to perform this code for each margin of error.

```r
# Now perform the Monte Carlo simulations,
#   for 10 trail, with 10000 simulated Presidential election in each trail
df_mr_016 <- MC_simulate(polls_dat, mr_err)
```

```
# This is a very computational expensive process
# This code (already improved) took about 5 minutes to complete the task
# Information about the Version of R will be provided in the Appendix 5.2
# Now we save the resulted file, and retrieved later
saveRDS(df_mr_016, "/Users/mac/Desktop/PresidentialElection/election_MR_0.16_CLT.csv")
# df_mr_016 <- readRDS("/Users/mac/Desktop/PresidentialElection/election_MR_016.csv")

head(df_mr_016)

# Save the results in to local file
# Note that the result is a data.frame, which may not be opened directly
df_mr_016

hist(df_mr_016$Est_Votes_Biden,
     xlab = "Total Electoral Votes",
     main = "Histogram of electoral votes for Biden")

mean(df_mr_016$Est_Votes_Biden)   # 339.6409
mean(df_mr_016$Est_Votes_Trump)   # 198.3591
mean(df_mr_016$Est_Prob_Biden)    # 0.982095
mean(df_mr_016$Est_Prob_Trump)    # 0.013845

############################################################################
###                       End of The Demonstration                       ###
############################################################################
```

```
############################################################################
###  Repeat the same process, with Margin of error 0.04, 0.08, and 0.12  ###
############################################################################

df_mr_004 <- MC_simulate(polls_dat, 0.04)
df_mr_008 <- MC_simulate(polls_dat, 0.08)
df_mr_012 <- MC_simulate(polls_dat, 0.12)
saveRDS(df_mr_004, "/Users/mac/Desktop/PresidentialElection/election_MR_0.04_CLT.csv")
saveRDS(df_mr_008, "/Users/mac/Desktop/PresidentialElection/election_MR_0.08_CLT.csv")
saveRDS(df_mr_012, "/Users/mac/Desktop/PresidentialElection/election_MR_0.12_CLT.csv")

result_004 <- readRDS("/Users/mac/Desktop/PresidentialElection/election_MR_0.04_CLT.csv")
result_008 <- readRDS("/Users/mac/Desktop/PresidentialElection/election_MR_0.08_CLT.csv")
result_012 <- readRDS("/Users/mac/Desktop/PresidentialElection/election_MR_0.12_CLT.csv")
result_016 <- readRDS("/Users/mac/Desktop/PresidentialElection/election_MR_0.16_CLT.csv")

# Matrix to store the averaged votes for both candidate for each margin of error
votes_all_mr <- cbind(sort(result_004$Est_Votes_Biden), sort(result_008$Est_Votes_Biden),
                      sort(result_012$Est_Votes_Biden), sort(result_016$Est_Votes_Biden),
                      sort(result_004$Est_Votes_Trump), sort(result_008$Est_Votes_Trump),
                      sort(result_012$Est_Votes_Trump), sort(result_016$Est_Votes_Trump))
colnames(votes_all_mr) <- c("VB_0.04", "VB_0.08",
                            "VB_0.12", "VB_0.16",
                            "VT_0.04", "VT_0.08",
                            "VT_0.12", "VT_0.16")
votes_all_mr <- as.data.frame(votes_all_mr)
```

```r
head(votes_all_mr, 5)
# saveRDS(votes_all_mr, "/Users/mac/Desktop/PresidentialElection/election_all_MR.csv")

par(mfrow = c(1, 2))
vote_boxplot1 <- boxplot(votes_all_mr[, 1:4],
                         xlab = "Votes received by each candidate")

vote_boxplot1 <- boxplot(votes_all_mr[, 2:3],
                         xlab = "Votes received by Trump")

vote_boxplot2 <- boxplot(votes_all_mr[, 4:6],
                         xlab = "Votes received by Trump")

# Matrix to store the averaged prob of winning for both candidate for each margin of error
prob_all_mr <- cbind(sort(result_004$Est_Prob_Biden), sort(result_008$Est_Prob_Biden),
                     sort(result_012$Est_Prob_Biden), sort(result_016$Est_Prob_Biden),
                     sort(result_004$Est_Prob_Trump), sort(result_008$Est_Prob_Trump),
                     sort(result_012$Est_Prob_Trump), sort(result_016$Est_Prob_Trump))
colnames(prob_all_mr) <- c("PB_0.04", "PB_0.08",
                           "PB_0.12", "PB_0.16",
                           "PT_0.04", "PT_0.08",
                           "PT_0.12", "PT_0.16")

prob_all_mr <- as.data.frame(prob_all_mr)
prob_all_mr
# saveRDS(prob_all_mr, "/Users/mac/Desktop/PresidentialElection/election_prob_all_MR.csv")
kable(prob_all_mr)

par(mfrow = c(1, 2))

prob_boxplot1 <- boxplot(prob_all_mr[, 1:3],
                         xlab = "Margin of Error",
                         main = "Estimated probability that Biden wins",
                         ylab = "Prob")

prob_boxplot2 <- boxplot(prob_all_mr[, 4:6],
                         xlab = "Margin of Error",
                         main = "Estimated probability that Trump wins",
                         ylab = "Prob")


##########################################################################
###  The following code simulates 1000 elections for 1000 times with MR = 0.12  ###
##########################################################################
total_trial <- 1000
N <- 1000
clt_montecarlo <- MC_simulate(polls_dat, 0.12)
saveRDS(clt_montecarlo, "/Users/mac/Desktop/PresidentialElection/election_MR_0.12_clt_1000.csv")

# The R code for 1000 runs with 1000 simulations for each run
clt_montecarlo <- readRDS("/Users/mac/Desktop/PresidentialElection/election_MR_0.12_clt_1000.csv")

# The histogram of electoral votes for Biden and Trump
# This shows the Central Limit Theorem
clt_hist1 <- hist(clt_montecarlo$Est_Votes_Biden,
                  xlab = "Electoral votes",
                  main = "Electoral votes for Biden")

clt_hist2 <- hist(clt_montecarlo$Est_Votes_Trump,
                  xlab = "Electoral votes",
                  main = "Electoral votes for Trump")

clt_hist3 <- hist(clt_montecarlo$Est_Prob_Biden,
                  main = "Probability of voting for Biden",
                  xlab = "Prob")

clt_hist4 <- hist(clt_montecarlo$Est_Prob_Trump,
                  main = "Probability of voting for Trump",
                  xlab = "Prob")

# Calculate the confidence interval
mean(clt_montecarlo$Est_Votes_Biden)    # 342.2171
mean(clt_montecarlo$Est_Votes_Trump)    # 195.7829
mean(clt_montecarlo$Est_Prob_Biden)     # 0.989433
mean(clt_montecarlo$Est_Prob_Trump)     # 0.00783

sd(clt_montecarlo$Est_Votes_Biden)      # 1.018773
sd(clt_montecarlo$Est_Votes_Trump)      # 1.018773
sd(clt_montecarlo$Est_Prob_Biden)       # 0.002803026
sd(clt_montecarlo$Est_Prob_Trump)       # 0.003221784

# [mean - 3 * sd, mean + 3 * sd]
# # 339.1607
c(mean(clt_montecarlo$Est_Votes_Biden) - 3 * sd(clt_montecarlo$Est_Votes_Biden))
# 192.7266
c(mean(clt_montecarlo$Est_Votes_Trump) - 3 * sd(clt_montecarlo$Est_Votes_Trump))
# 0.9797676
c(mean(clt_montecarlo$Est_Prob_Biden) - 3 * sd(clt_montecarlo$Est_Prob_Biden))
# -0.0005790792
c(mean(clt_montecarlo$Est_Prob_Trump) - 3 * sd(clt_montecarlo$Est_Prob_Trump))
```

The following is the R code for simulating the reelection.

```r
real_dat <- read.csv("/Users/mac/Desktop/PresidentialElection/Popular vote backend - Sheet1.csv")

N <- 10000
total_trial <- 10

# Clean the data set
real_election <- real_dat %>%
  select(state, dem_percent, rep_percent, EV, dem_this_margin)

real_election$dem_percent <- as.numeric(
  sub("%", "", real_election$dem_percent, fixed=TRUE)) / 100
real_election$rep_percent <- as.numeric(
  sub("%", "", real_election$rep_percent, fixed=TRUE)) / 100
```

```r
real_election$dem_this_margin <- as.numeric(
  sub("%", "", real_election$dem_this_margin, fixed=TRUE)) / 100

real_election <- real_election %>%
  arrange(sort(abs(dem_this_margin)))

kable(head(real_election, 15))

# Set seed to ensure the results can be reproducible
set.seed(270)

# Number of iterations per trial
N <- 10000

# Total number of trials
total_trial <- 10

# Repeat the process, with MR = 0.02, 0.04,..., 0.12
re_df_mr_002 <- MC_simulate(real_election, 0.02)
re_df_mr_004 <- MC_simulate(real_election, 0.04)
re_df_mr_006 <- MC_simulate(real_election, 0.06)
re_df_mr_008 <- MC_simulate(real_election, 0.08)
re_df_mr_010 <- MC_simulate(real_election, 0.10)
re_df_mr_012 <- MC_simulate(real_election, 0.12)

# Save and Retrieve Results
saveRDS(re_df_mr_002, "/Users/mac/Desktop/PresidentialElection/reelection_MR_0.02.csv")
saveRDS(re_df_mr_004"/Users/mac/Desktop/PresidentialElection/reelection_MR_0.04.csv")
saveRDS(re_df_mr_006"/Users/mac/Desktop/PresidentialElection/reelection_MR_0.06.csv")
saveRDS(re_df_mr_008"/Users/mac/Desktop/PresidentialElection/reelection_MR_0.08.csv")
saveRDS(re_df_mr_010"/Users/mac/Desktop/PresidentialElection/reelection_MR_0.10.csv")
saveRDS(re_df_mr_012"/Users/mac/Desktop/PresidentialElection/reelection_MR_0.12.csv")

reelect_results_002 <- readRDS("/Users/mac/Desktop/PresidentialElection/reelection_MR_0.02.csv")
reelect_results_004 <- readRDS("/Users/mac/Desktop/PresidentialElection/reelection_MR_0.04.csv")
reelect_results_006 <- readRDS("/Users/mac/Desktop/PresidentialElection/reelection_MR_0.06.csv")
reelect_results_008 <- readRDS("/Users/mac/Desktop/PresidentialElection/reelection_MR_0.08.csv")
reelect_results_010 <- readRDS("/Users/mac/Desktop/PresidentialElection/reelection_MR_0.10.csv")
reelect_results_012 <- readRDS("/Users/mac/Desktop/PresidentialElection/reelection_MR_0.12.csv")


# Matrix to store the averaged votes received by Biden with each margin of error
votes_biden_re <- cbind(sort(reelect_results_002$Est_Votes_Biden),
                        sort(reelect_results_004$Est_Votes_Biden),
                        sort(reelect_results_006$Est_Votes_Biden),
                        sort(reelect_results_008$Est_Votes_Biden),
                        sort(reelect_results_010$Est_Votes_Biden),
                        sort(reelect_results_012$Est_Votes_Biden))
colnames(votes_biden_re) <- c("VB_0.02", "VB_0.04",
                              "VB_0.06", "VB_0.08",
                              "VB_0.10", "VB_0.12")
votes_biden_re <- as.data.frame(votes_biden_re)
saveRDS(votes_biden_re, "/Users/mac/Desktop/PresidentialElection/votes_biden_re.csv")
```

```
# Matrix to store the averaged votes received by Trump with each margin of error
votes_trump_re <- cbind(sort(reelect_results_002$Est_Votes_Trump),
                        sort(reelect_results_004$Est_Votes_Trump),
                        sort(reelect_results_006$Est_Votes_Trump),
                        sort(reelect_results_008$Est_Votes_Trump),
                        sort(reelect_results_010$Est_Votes_Trump),
                        sort(reelect_results_012$Est_Votes_Trump))
colnames(votes_biden_re) <- c("VT_0.02", "VT_0.04",
                              "VT_0.06", "VT_0.08",
                              "VT_0.10", "VT_0.12")
votes_trump_re <- as.data.frame(votes_trump_re)
saveRDS(votes_trump_re, "/Users/mac/Desktop/PresidentialElection/votes_trump_re.csv")



# Matrix to record the averaged probability of winning for Biden
prob_biden_re <- cbind(sort(reelect_results_002$Est_Prob_Biden),
                       sort(reelect_results_004$Est_Prob_Biden),
                       sort(reelect_results_006$Est_Prob_Biden),
                       sort(reelect_results_008$Est_Prob_Biden),
                       sort(reelect_results_010$Est_Prob_Biden),
                       sort(reelect_results_012$Est_Prob_Biden))
colnames(prob_biden_re) <- c("PB_0.02", "PB_0.04",
                             "PB_0.06", "PB_0.08",
                             "PB_0.10", "PB_0.12")
prob_biden_re <- as.data.frame(prob_biden_re)
saveRDS(prob_biden_re, "/Users/mac/Desktop/PresidentialElection/prob_biden_re.csv")

# Matrix to store the averaged votes received by Trump with each margin of error
prob_trump_re <- cbind(sort(reelect_results_002$Est_Prob_Trump),
                        sort(reelect_results_004$Est_Prob_Trump),
                        sort(reelect_results_006$Est_Prob_Trump),
                        sort(reelect_results_008$Est_Prob_Trump),
                        sort(reelect_results_010$Est_Prob_Trump),
                        sort(reelect_results_012$Est_Prob_Trump))
colnames(prob_trump_re) <- c("PT_0.02", "PT_0.04",
                             "PT_0.06", "PT_0.08",
                             "PT_0.10", "PT_0.12")
prob_trump_re <- as.data.frame(prob_trump_re)
saveRDS(prob_trump_re, "/Users/mac/Desktop/PresidentialElection/prob_trump_re.csv")
```

5.2 Computer Software Version

The Version of R
R version 3.6.3 (2020-02-29) – "Holding the Windsock" Copyright (C) 2020 The R Foundation for Statistical Computing Platform: x86_64-apple-darwin15.6.0 (64-bit)

The Version of RStudio
2022.07.2+576