

A brief note on the effect of errors in the recount process, 2.0

Eric Lawrence
Department of Political Science
George Washington University

edl@gwu.edu

November 16, 2000

Thanks to Katherine Harris (my wife, not the Florida Secretary of State) for advice on setting up the Stata simulations. I was inspired to write this note after getting questions from students and from a reporter regarding why the recount numbers were “so different” from the original numbers. I want to thank Langche Zeng for useful suggestions and advice regarding the normal approximations, as well as Dave Rusin for also pointing out the normal approximation and lending advice.

Introduction

In the ongoing controversy regarding the presidential election results in Florida, political scientists, economists, and applied statisticians have been quick to analyze the distribution of votes in Florida.¹ The overwhelming consensus is that unless Palm Beach County is made up of voters different in fundamental ways from similar voters across Florida and across the country, something strange happened in Palm Beach County on Election Day. The most obvious culprit is the now nefarious “butterfly ballot”, which may be illegal under Florida state law.

While there has been considerable attention paid to the Buchanan vote, however, I have only found one statistical analyses of the problematic nature of recounting ballots when the election is this close.² And the stakes are tremendously high here—the presidency hangs in the balance. One side asserts the desire to obtain the most accurate count possible (the Gore team), implying the need for recounts. The other side, led by Jim Baker, has expressed concern that the Democrats want to keep counting until they get a result they like. Baker and the rest of the Bush team have been repeating the idea that machines don’t make mistakes (since they are neither Democrats nor Republicans) but that hand counting is fraught with subjectivity. This brief note does not take up the John Henry question of human versus machine, though the assertion that machines do not make mistakes seems dubious given the changes in the vote tallies after the first machine recount. We know that mechanical processes are not completely reliable; every now and then, we find a misshapen M & M, a malformed nail, or some other defective product from a manufacturing product.³

The statistical problem, as I am framing it here, is one of measurement error. In most cases measurement error does not matter if it is random and the election is not close. But in this case, what we are trying to estimate is the percentage of the two party vote for one candidate when the two percentages are extremely close. Even a small amount of random measurement error in the counting process may be enough to swing the election to one side or the other. The randomness in the counting process does not derive from sampling, but the implications of the stochastic component of the count are similar. We are trying to estimate the proportion of the two party vote received by each candidate. If counting were error free, we would not be *estimating* the proportions, we would just be *counting* the votes. But counting seems to have at least some errors, whether performed by humans or machines. If there is not *much* measurement error, we will end up with an estimate that is very close. But will it be close enough to get the count right? For that, I turn to Monte Carlo simulation and then normal approximations.

Methods I: Monte Carlo Simulations

The approach I take here is quite straightforward. First, I assume a distribution of votes. Then I assume an error rate in the counting process. Each voter draws an error from the error rate distribution. For each voter, the likelihood that the true vote will be recorded is the

¹ See the website set up by Greg Adams (CMU) and Chris Fastnow (Chatham College) for their analyses and links to several other analyses of the Florida vote, the Palm Beach vote, and the Buchanan outliers. See <http://madison.hss.cmu.edu> for Adams and Fastnow’s analyses and links to many other analyses.

² Dave Rusin of the Northern Illinois Math department kindly alerted me to his paper. You can access it at: <http://www.math.niu.edu/~rusin/recount>. The paper gives a more precise explanation for the normal approximation to the binomial in this context and also discusses recounting in Palm Beach County.

³ And given all the stories that have surfaced since the election regarding misplaced bags of ballots, various forms of chads, and poorly trained workers, it is reasonable to assume that counting ballots is more error prone than candy making. Not that I know anything about making candy, but the point should be clear.

complement of the error rate. The votes are then tallied and saved, with this process repeated 1000 times. In the end, we have a “sampling distribution” of the number of counted votes for each candidate.⁴

For this note, I assume a population of 600,000, with a 20-vote margin between the two candidates. Candidate A therefore gets 300,010 votes, and Candidate B gets 299,990 votes. This corresponds to percentages of 50.0017% for Candidate A and 49.9983% for candidate B, with rounding.⁵ I then run two Monte Carlo simulations, varying only the error rate. In the first simulation, I assume an error rate of 1 in 1000. Each voter’s gets an error draw from $u1 \sim (0,1)$, and their vote is tallied incorrectly if their draw is less than .001. In the second simulation, I assume an error rate of 1 in 10,000, and the voter’s vote is tallied incorrectly if their draw is less than .0001.

Results I

The results of the simulation with an error rate of 1 in 1000 are shown in figure 1 below. The left vertical line, located at 299,990, denotes the true number of votes for the candidate and also happens to be the mean of the simulated distribution. Each observation to the right of the second vertical line, located at 300,000 votes, represents a case where the candidate with *more* votes in the true distribution gets *fewer* votes in the distribution with error. I call that a “wrong” result, and the frequency of the wrong result in this simulation are shown below. Of the 1000 simulation, 33.4% of them end up with the “wrong” winner. One in three times, the incorrect winner is declared. And this occurs even with the winning candidate’s votes to be nearly identically likely to suffer an error in tallying as the losing candidate.

Percentage of wrong results, by error rate
[both based on 1000 Monte Carlo simulations]

Wrong winner?	Error rate=.001	Error rate=.0001
No	66.60	91.00
Yes	33.40	9.00
Total	100.00	100.00

The results of the simulation with an error rate of 1 in 10,000 are shown in figure2 below. The situation improves, but the wrong winner is still declared 9% of the time. But consider that result. With a population of 600,000 and an error rate of 1 in 10,000, we should expect there to be 60 total errors, with 30 errors for each candidate. Nevertheless, 9 times out of 1000, the measurement error was sufficient to swing at least 20 votes in the direction of the losing candidate, making her the winner.

Methods II: normal approximations

After reading my paper, both Dave Rusin and Langche Zeng pointed out that one need not simulate to get the right answer here. Alternatively, one can use the normal approximation to

⁴ I put sampling distribution in quotes in case referring to it as such offends anyone’s statistical sensibilities.

⁵ Simulating a population of 6,000,000 with a vote margin of 200 would have been preferable, given the numbers reported so far in the media, but I will leave that to someone else with more RAM. These numbers are therefore an order of magnitude smaller than Florida. But see the normal approximations below, where I tackle a Florida sized population.

the binomial distribution, given the large n context. This approach has many advantages, the main one being the speed gains that are possible. By using the normal approximation, I am able to assess a broader range of scenarios in much less time. To frame the problem as a binomial counting process, consider each vote cast to be either a vote for or against the losing candidate.⁶ Think of a for vote as a 1, and an against vote as a 0. Variation is then generated by the errors in the counting process. Then, for the true losing candidate to be incorrectly named the winner the number of against vote errors minus the number of for vote errors must be greater than half the final vote margin. Why *half* the final margin? Because each counting error that produces a vote flip either reduces or increases the margin by two.⁷ By using the normal approximation, evaluating $P(\text{against errors} - \text{for errors} > \text{margin}/2)$ is relatively straightforward. The results are summarized both in tabular and graphical form below.

Results II: normal approximations

First, two tables are presented. The cells contain the probability that the candidate with fewer votes wins, strictly due to measurement error. The columns vary by error rates, and the rows vary by vote margins. The two bolded entries are the scenarios simulated above. The Monte Carlo simulations “worked”, as the results are consistent with each other. The second table presents a Florida sized voting population. Notice that the probability of a wrong outcome is greater in the larger population, controlling for error rate and margin. This makes sense, given the law of large numbers. But also notice that the probability of selecting the wrong winner is non trivial in large populations and higher error rates. In this table, the error rate ranges from .005 to .00001. In the figures below, however, I vary the error rates over a wider range, from .1 to .00001. The results for error rates of .1 and .01 should give us pause. Even with wide margins (by 2000 standards), a considerable proportion of the time, one should expect wrong winners. I have been told, but have not been able to confirm, that Tillie Fowler (R-FL) asserted on television recently that machine error rates are in the 2-5% range, and that the human error rate is higher. I am skeptical of those numbers, but what if she is right? A careful consideration of the lower two graphs in the population 6,000,000 scenario may suggest that the best course of action is to declare Florida a tie and let Gore and Bush settle things with a field goal kicking contest during half time of the Florida-Florida State game.

Conclusion

If we have learned anything from this election, we have learned that counting votes is a hard problem when the division between the two leading vote getters is extremely close. The results above suggest the problem in counting and recounting votes when the counting process is imperfect. If there is any measurement error in the process, we should expect that recounting will lead to different winners being declared, *even with the true distribution of votes remaining unchanged*. Does this apply to the current situation in Florida? Perhaps, but we have little way of knowing without additional information. In particular, extending the results above would need to consider:

⁶ Of course, one could also do this for the winning candidate as well.

⁷ Again, thanks to Langche Zeng for illuminating this point to me.

- The possibility of different error rates for different kinds of ballots. Do punched ballots have higher measurement error rates than other types of ballots?
- The error rates. What is a reasonable estimate of the error rate? The anecdotal evidence we are witnessing suggests that it might be higher than we would initially estimate.

Population of 600,000 voters: “New Mexico”
 [normal approximations to binomial probabilities]

Margin/2	Error rates			
	.005	.001	.0001	.00001
10	0.42810451	0.34177059	0.09838638	.0000223
50	0.18246526	0.02076605	.	.
100	0.03499154	.0000229	.	.
200	0.00014502	.	.	.
300

Population of 6,000,000 voters :“Florida”
 [normal approximations to binomial probabilities]

Margin/2	Error rates			
	.005	.001	.0001	.00001
10	0.47715267	0.44871613	0.34156818	0.09835616
50	0.38724547	0.25961637	0.02062863	.
100	0.28331859	0.09868911	0.0000223	.
200	0.12589299	0.00496706	.	.
300	0.04280416	0.0000551	.	.
400	0.01095154	.	.	.
500	0.00208468	.	.	.
600	0.00029294	.	.	.
700	0.0000302	.	.	.
800

Figure 1: Kernel Density Plot for Monte Carlo distribution

Figure 1: Observed number of votes, with error rate of .001
True number of votes is 299,990 [left vertical line]

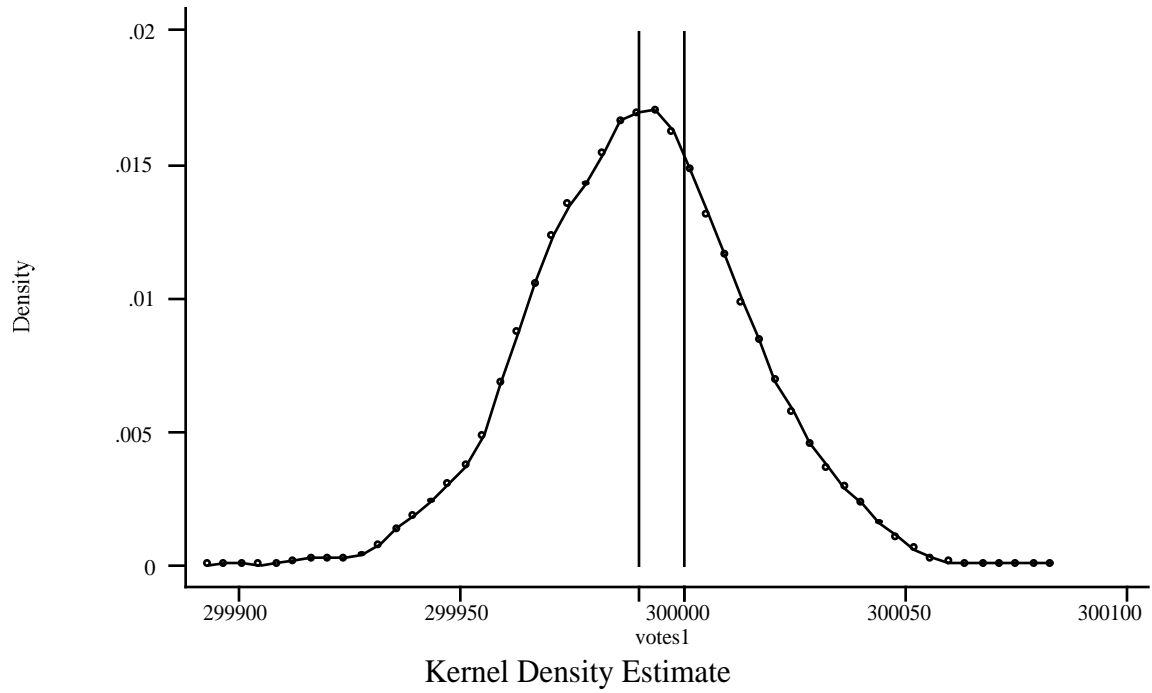
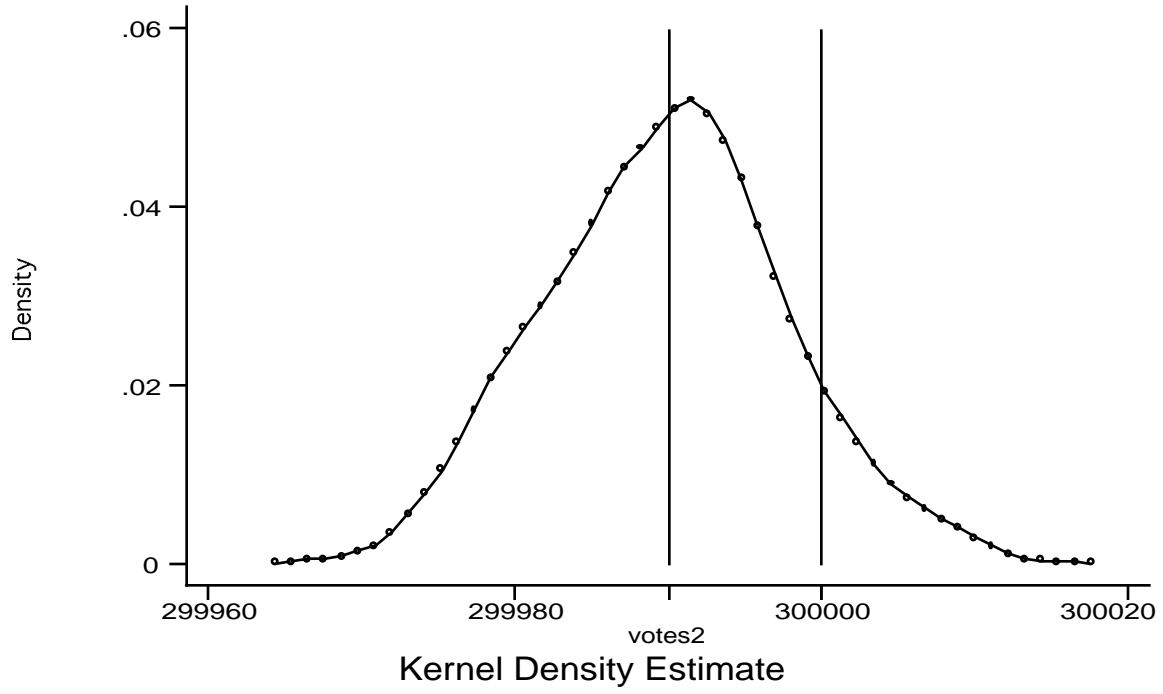
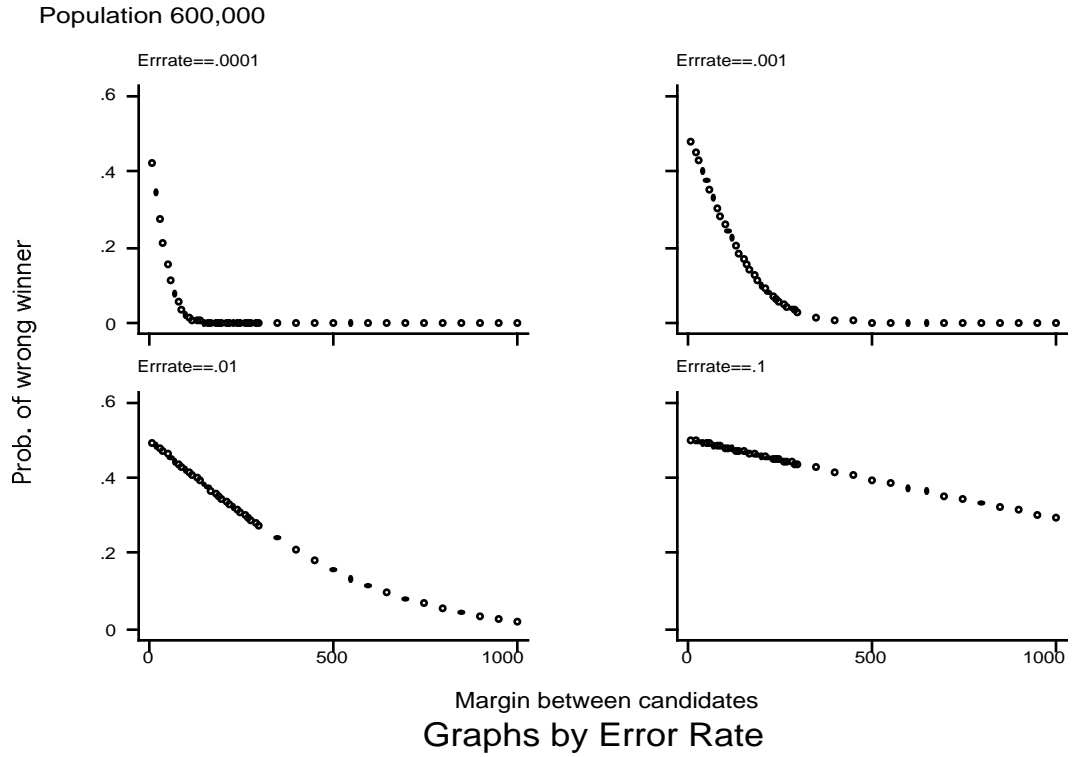


Figure 2: Kernel Density Plot for Monte Carlo distribution

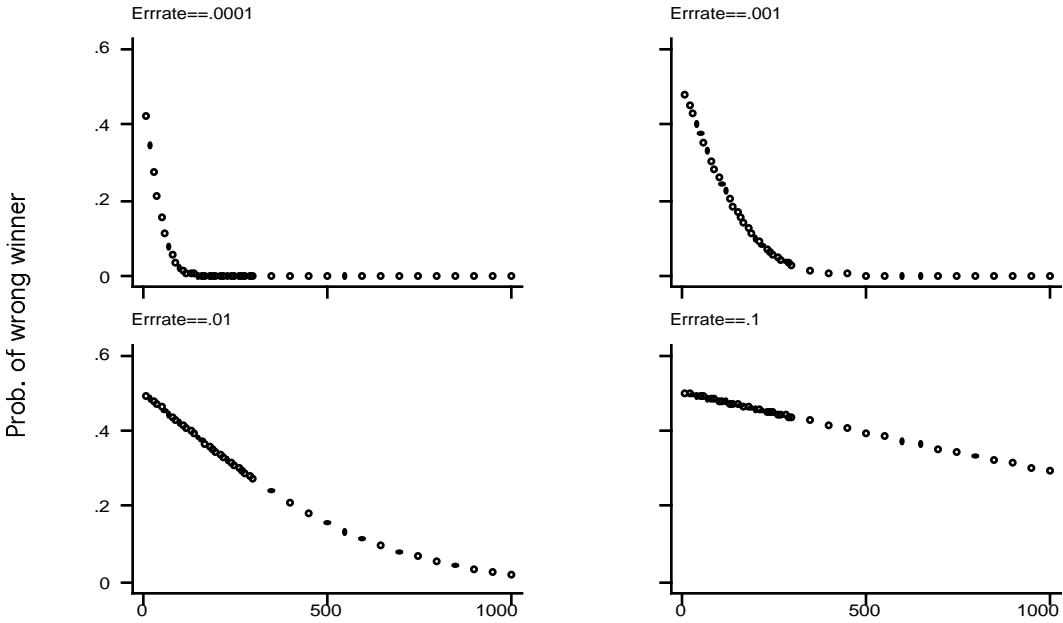
Figure 2: Observed number of votes, with error rate of .0001
True number of votes is 299,990 [left vertical line]



Normal Approximations for Probability of Wrong Winner selected, By voting population, margin, and error rate



Population 6,000,000



Margin between candidates
Graphs by Error Rate

Appendix A: Stata 6.0 code for the Monte Carlo simulations

```
* On recounting, Eric D. Lawrence, 11/14/2000
* George Washington University, Dept. of Political Science
* edl@gwu.edu
*
* This is a set of programs to evaluate the perils of estimating a
* percentage, when the percentage is close to .5 and there is noise
* in the counting process.
*
* This was inspired by a reporter's question regarding the statistical
* issues involved in the Florida recounts.
*
* Any suggestions, critiques, or complaints are welcome.
* This was jiggered up in an hour, so it isn't perfect.
*
* Below are 2 programs.
* Both assume a true vote margin of 20 votes, and a voting population
* of 600,000
* A 200 vote margin and a voting population of 6,000,000 would have been
* preferable but someone with more RAM can do that if they like.
* The general problem is the same. Think New Mexico if you like.
*
* No assumptions are made about the nature of the errors.
* Rather, I merely assume that they occur at the following rates.
*
* handcnt1 assumes an error rate of 1 in 1000
* handcnt2 assumes an error rate of 1 in 10000
*
*
program define handcnt1
    if "`1'"=="?" {
        global S_1 "mean var"
        exit
    }
    drop _all
    set obs 600000
    gen id = _n
    gen true = id > 300010
    gen u = uniform()
    gen counted = true if u > .001
    replace counted = abs(1-true) if counted==.
    summ counted
    post `1' r(mean) r(var)
end

*
simul handcnt1, reps(1000) dots saving(handcnt1.dta) replace
*
gen votes1 = mean*600000
kdensity votes1, xline(299990 300000) xtick(299990 300000) ylab xlab /*
    /* t1(Figure 1: Observed number of votes, with error rate of .001) /*
    /* t2(True number of votes is 299,990 [left vertical line])
gen wrong1 = votes1 > 300000
tab wrong1
```

```

*
program define handcnt2
    if "`1'"=="?" {
        global S_1 "mean var"
        exit
    }
    drop _all
    set obs 600000
    gen id = _n
    gen true = id > 300010
    gen u = uniform()
    gen counted = true if u > .0001
    replace counted = abs(1-true) if counted==.
    summ counted
    post `1' r(mean) r(var)
end

*
simul handcnt2, reps(1000) dots saving(handcnt2.dta) replace
*
gen votes2 = mean*600000
kdensity votes2, xline(299990 300000) xtick(299990 300000) ylab xlab /*
    */ t1(Figure 2: Observed number of votes, with error rate of .0001) /*
    */ t2(True number of votes is 299,990 [left vertical line])
gen wrong2 = votes2 > 300000
tab wrong2

```

Appendix B: Stata 6.0 code for the normal approximations

```

* on counting, take 2
* eric lawrence, 11/15/2000
* dept. of political science, GWU
*
* Thanks to Langche Zeng, GWU Political Science and
* Dave Rusin, Northern Illinois Math Department for useful suggestions and pointing
* the way away from Monte Carlo toward solving this with normal approximations
* to the binomial
*
* normal approximation to the binomial
* for the likelihood of the wrong winner getting more votes
* strictly due to random measurement error symmetric across the two candidates
*
*
* DEFINITIONS
* total votes == V
*
* number of votes separating the candidates == margin
* number of vote switches needed to flip the election == margin/2
* true losing candidate == L
* true winning candidate == W
*
* number of votes against L == Lno == (V + margin)/2
* number of votes for L == Lyes == V - (V + margin)/2 == (V - margin)/2
*
* but mistakes are made at rate e,  $0 < e < 1$ 

```

```

*
* what needs to happen for the true loser to win is for the loser
* to get more errors in the No votes than in the yes votes, and the loser must
* get at least margin/2 more favorable errors
*
* so call Eno the no vote errors and Eyes the yes vote errors
*
* We need to find P(Eno-Eyes>margin/2)
*
* Eno and Eyes ~ binomial, with error rate e
* but with large V, we can approximate with the normal, so
*
* Eno ~ N(Lno*e, Ln*e*1-e)
* Eyes ~ N(Lyes*e, Lyes*e*1-e)
*
* Given those assumptions, we can calculate the probability of calling the election
the wrong way
* strictly due to measurement error (machine error, wrinkled ballots, etc.)
*
* P(wrong winner) = P(Eno-Eyes > margin/2) = 1 - normprob(Eno-Eyes<margin)
*
* where the latter equals 1 - normprob((margin/2 - (Lno-Lyes)*e)/sqrt((Lno-Lyes)*e*(1-
e)))
*
*
local v =600000
while `v' <= 6000000 {
    local m = 10
    while `m' <= 300 {
        local e = .1
        while `e' > .00001 {
            local Lno = (`v' + `m')/2
            local Lyes = (`v'- `m')/2
            local pwrongz = (`m'/2 - (`Lno'-
`Lyes')*`e')/((`Lno'+`Lyes')*`e'*(1-`e'))^.5
            local pwrong = 1 - normprob(`prongz')
            di _col(5) `v' _col(15) `m' _col(25) `e' _col(35) `prong'
            local e = `e'/10
        }
        local m = `m' + 10
    }
    local v = `v'*10
}
local v =6000000
while `v' <= 6000000 {
    local m = 350
    while `m' <= 1000 {
        local e = .1
        while `e' > .00001 {
            local Lno = (`v' + `m')/2
            local Lyes = (`v'- `m')/2
            local pwrongz = (`m'/2 - (`Lno'-
`Lyes')*`e')/((`Lno'+`Lyes')*`e'*(1-`e'))^.5
            local pwrong = 1 - normprob(`prongz')
            di _col(5) `v' _col(15) `m' _col(25) `e' _col(35) `prong'
            local e = `e'/10
        }
    }
}

```

```
        local m = `m' + 50  
    }  
local v = `v'*10  
}
```