A SIMULATION APPROACH TO OPTIMIZING SELECTION OF THE STANDARD ERROR SPECIFICATION IN COUNT DATA MODELING

by
Robert Aaron Nathenson

A thesis submitted to Johns Hopkins University in conformity with the requirements for the degree of Master of Science in Engineering

Baltimore, Maryland
April, 2014

Abstract

Quantitative social scientists assume their model fit is appropriate to the data, especially the theoretical distribution of choice. However, researchers spend less time justifying the standard error specification. This step is critical as a misspecification of the standard error can lead to an incorrect interpretation of the independent variables, or parameters, of the model. Because researchers derive further research agendas and policy implications directly from the significance of their results, a misspecification of the standard error has large real world ramifications. This research, therefore, examines the validity of typically applied standard error techniques in Poisson and Negative Binomial regression in a case study framework, such as the observed information matrix, outer product of the gradient, clustering, nonparametric bootstrapping, and the jackknife procedure. A dataset of 2005 to 2011 state-based pro-/neutral and anti-immigration legislation is employed. In order to assess the validity of these standard error techniques I sample from the fitted conditional Poisson or Negative Binomial model to create a Monte Carlo (MC) simulation, which yields an estimate of the 'true' standard error. A relative error calculation then compares the commonly utilized standard error techniques to the MC 'true' standard error. The results indicate that the observed information matrix performs particularly well for small sample sizes. The jackknife procedure also performs quite well. Results for the nonparametric bootstrap, however, vary tremendously across iterations. Though the conclusions of this research are unlikely to generalize to other datasets, the approach taken may easily be adapted to other situations and other model formulations in which researchers are concerned with which standard error method to use. I include sample Stata code to illustrate the approach.

Acknowledgements

I would like to thank Professor Daniel Naiman for his care, guidance, and countless hours of rather enjoyable delving in Stata's technical documentation.

For Ali.

TABLE OF CONTENTS

List of Tables

List of Figures

Introduction

This research investigates the validity of commonly used standard error techniques in Poisson and Negative Binomial modeling. Using a publicly available dataset from the National Conference of State Legislatures, I create a dataset of pro-/neutral immigrant legislation passed per U.S. state per year from 2005 to 2011 and another dataset across the same time period, likewise, for anti-immigrant legislation. Conditioning on state level data, such as the state unemployment rate and the foreign-born population, I fit the Poisson and Negative Binomial distributions to the immigrant legislation data. I use a Monte Carlo simulation (MC), in which the counts of pro-/neutral (or anti-) immigrant legislation are randomly generated from the fitted conditional Poisson (or Negative Binomial) distribution, to obtain a good approximation to the true standard errors. I then compare such common standard error techniques as the observed information matrix and jackknife to the MC approximation. I calculate the relative error of each method in reference to the MC approximation, which provides an assessment of the validity of each of these methods. This work aims to inform researchers of the strengths and weaknesses of the most commonly applied standard error techniques and increases the rigor of hypothesis testing in the social sciences.

Conceptualization

Quantitative social scientists rely on statistical programs to analyze their data. They assume their model fit is appropriate to the data, especially the theoretical distribution of choice. However, researchers spend less time justifying the standard error specification. This step is critical as the calculation of the standard error dictates whether independent variables are considered significant. A misspecification of the standard error can lead to an incorrect interpretation of the independent variables, or parameters, of the model.

An example of a parallel threat to validity helps illustrate the issue. Consider the omitted variable bias in which significance is incorrectly assigned to an independent variable. For instance, an increase in the number of deaths from swimming is incorrectly attributed to an increase in the number of children eating ice cream. The omitted mechanism is time of year – in the summer months in which public pools are opened, children are out of school, and hotter temperatures abound, there is greater ice cream consumption and more opportunity to swim. Without knowledge of this omitted variable, researchers could incorrectly infer that consuming ice cream increases the risk of drowning.

Similarly, a misspecification of the standard errors can lead to incorrect inferences. If the true magnitude of the standard error is under-specified, then researchers will misinterpret results by concluding unimportant parameters to be significantly influential. Because researchers derive further research agendas and policy implications directly from the significance of their results, a misspecification of the standard error has large real world ramifications.

To reduce this threat to validity, researchers may relax their standard error assumptions through such techniques as robust standard errors. For a further discussion of robust standard errors, see Eicker (1967), White (1980a), and Angrist & Pischke (2009). However, too often researchers do not question the appropriateness of the standard error techniques employed in their research.

This research aims to inform social scientists in this capacity. More specifically, it examines the validity of typically applied standard error techniques in Poisson and Negative Binomial regression in a case study framework. The most commonly used standard error specification is the observed information matrix (oim), which is the default technique in Stata. The oim is a sample-based version of Fisher Information derived from the negative of the second derivative of the log likelihood function (Efron & Hinkley 1978). Other common techniques include robust standard errors, a means to relax standard error specifications to handle model misspecifications (Angrist & Pischke 2009), the outer product of the gradient, clustering (e.g. by state or by year for immigrant legislation), a nonparametric bootstrap in which the model is run a number of predetermined times by resampling observations with replacement from the actual data, and jackknife. In the jackknife specification one observation at a time is removed from the model, the model is run with n-1 observations, and results are averaged across the n models.

In order to assess the validity of these standard error techniques, I create a 'true' standard error. I sample from the fitted conditional Poisson (or Negative Binomial) model to create a Monte Carlo simulation, which yields an estimate of the true standard error when the model specification is taken as 'ground-truth' for the population of

interest.[1] Using a relative error calculation that compares commonly used standard error techniques to the MC 'true' standard error, I assess to what extent the typical standard error techniques under or overestimate the 'true' standard error. While I focus on a specific example, the conclusions of this research are unlikely to generalize to other datasets. Still, the approach taken may easily be adapted to other situations and other model formulations in which researchers are concerned with which standard error method to use.[2]

In the next section, I describe the data, the theoretical distributions that are fit to the data, and modeling strategies. I then discuss the calculation of the various common standard error specifications. Finally, I discuss the steps taken to create the MC simulation and the relative error calculation.

Research Design

Data

Dependent Variables

To examine the validity of commonly applied standard error techniques I utilize a pre-existing source of state-based immigrant-relevant legislation hosted by the National Conference of State Legislatures (NCSL). The NCSL annually compiles a list of immigrant-relevant legislation passed by each state since 2005. It is publicly available (see: http://www.ncsl.org/research/immigration/state-laws-related-to-immigration-and-

[1] The model specifications are based upon the specification of the pooled and panel Poisson and Negative binomial models in Stata. For details, see the poisson and nbreg entries in Stata's *Base Reference Manual, Release 12* and the xtpoisson and xtnbreg entries in the *Longitudinal-Data/Panel-Data Reference Manual, Release 12*.

[2] As a starting point, I provide some of the relevant Stata code in Appendices A and B.

immigrants.aspx).[3] Each law passed between 2005 and 2011 is rated as either anti-immigrant or pro-/neutral. The categories are mutually exclusive and exhaustive. The counts of each type of law for each state per year are compiled, which yields a measure of the amount of anti-immigrant legislation and pro-/neutral immigrant legislation passed in each state in each year.

While the data includes information for all fifty states, I focus on the forty-eight contiguous states. With seven years of information, the dataset consists of 336 observations. The mean number of pro-/neutral immigration laws passed per state per year is 1.44, with a standard deviation of 1.96, and maximum of 12. The mean, standard deviation, and maximum for anti-immigrant legislation are 1.13, 1.57, and 11, respectively. Because legislation is count data, and legislation is not required to be passed in each year, the minimum for both pro-/neutral and anti-immigrant legislation is 0.

Independent Variables

There are sixteen independent variables, or parameters, in the models. First, I include measures per year of the immigrant population in a state. These data come from the American Community Survey. The American Community Survey contains information on a state's number and proportion immigrant. I also include state-level unemployment, which comes from the Bureau of Labor Statistics.

[3] The current Immigration Enactments Database on the National Conference of State Legislatures website only contains enacted laws from 2008 to 2013. The organization has chosen to take down results for 2005, 2006, and 2007. For these years, please contact the NCSL directly.

Political party affiliation of the state house, senate, and governor come from *The Book of the States*, an annual periodical published by The Council of State Governments. This book contains information on the number of Republican, Democrat, Independent and other political party affiliation members of each state's legislature. While almost every legislature is bicameral, with both a House and Senate, Nebraska is unicameral, with a sole legislative body. Because the *Book of the States* includes counts of the Party affiliation of members in each part of the legislature, I am able to determine if Democrats or Republicans have a majority in the House, the Senate, and, cumulatively, the entire legislature, or if neither party is dominant. The *Book of the States* also includes the political party affiliation of the Governor, the Executive branch of the government.

I include dummy variables for whether the state shares a border with Mexico (a Southern border state), the state shares a border with Canada (a Northern border state), or if slavery was previously legalized within the state. I include binary variables to capture region of country (Northeast, South, Midwest, West) with Northeast the omitted category. Finally, I include year fixed effects to account for unobserved yearly shocks in the incidence of immigrant legislation, such as the ebb and flow of national interest in immigration reform.

Modeling Strategies

For both Pro-/Neutral and Anti-Immigrant legislation I examine Poisson and Negative Binomial modeling, treating the data, in turn, as both pooled and panel. These three sets of two (Pro-Neutral and Anti-; Poisson and Negative Binomial; pooled and panel) yield eight different models in total – four for Pro-/Neutral Immigrant legislation

and four for Anti-Immigrant legislation. I assess the validity of the common standard error specifications available for each of these eight models.[4]

Pooled Poisson

The Poisson distribution is used to model incidents of an event. For instance, the number of gold medals won by the Netherlands speed skating team in the 2014 Sochi Olympics. It is more appropriate than ordinary least squares regression when analyzing count data because ordinary least squares assumes the data is unbounded, $(-\infty, +\infty)$ (Gardner, Mulvey, & Shaw 1995; Osgood 2000). Count data, instead consists of whole numbers greater or equal to zero, $[0, \infty)$, which violates the unbounded assumption in OLS regression.

The Poisson distribution is defined by a single parameter, λ, which is both the mean and the variance of the distribution. The distribution, $P(Y = y) = \frac{e^{-\lambda}\lambda^{n}}{n!}$, measures the number of events, n, that occur over a period of time.

Pooled Negative Binomial

The requirement in the Poisson distribution that λ represents both the mean and the variance is a strict assumption often violated in real world data. If the variance is substantially larger than the mean in the observed data even after conditioning on parameters, a.k.a. 'overdispersion,' the negative binomial distribution can be used as an alternative means to model the data.

[4] I attempt to conduct the relative error calculations for each of these models. However, as discussed later in the paper, not all assessments were possible due to data limitations.

The Negative Binomial distribution has two parameters, p and r, where p is the probability that the incidence occurs and r is the dispersion parameter, accounting for the variance exceeding the mean. Formally, the negative binomial distribution has probability mass function of the form

$$P[Y = y] = \binom{y-1}{y-r}(1-p)^r p^{y-r},$$

equivalently

$$P[Y = y] = \frac{\Gamma(y+r)}{y!\Gamma(r)}(1-p)^r p^y.$$

The mean of the negative binomial distribution in terms of its parameters may be written as $\frac{(1-p)r}{p}$. I can denote this quantity by λ.

The variance may be written as $\sigma^2 = \lambda + \frac{\lambda^2}{r}$. As r approaches infinity the variance converges to λ. In this situation, λ has the same meaning for both the Negative Binomial and Poisson distributions. Therefore, the negative binomial distribution is a more general form of the Poisson distribution that allows for randomness in the rate of events. Subsequently, to say that a random variable, Y, has a negative binomial distribution, I will use the notation Y ~NB (λ, α), where λ is the mean parameter and α is the overdispersion parameter.

Panel Poisson

Pooled data assumes that the observations are independent and identically distributed (i.i.d.). This is incorrect for the immigrant legislative database because there is a relationship across years within states. Of the 336 observations, there are seven years

of data for each of the forty-eight contiguous United States. As an example, the virulently anti-immigrant sentiment exhibited in Alabama's 2011 law HB 56 is likely to be consistent across years, making the state more likely to pass anti- and less likely to pass pro-/neutral immigrant legislation than would be expected if each state-year observation were independent. Moreover, a state that passes an immigrant law in a given year has less need to pass a similar law the next year because laws often do not expire. A particularly productive immigrant legislative year should lead to relative lulls in subsequent years.

Due to the correlation in laws across years, i.e. the nested structure of state-year within state, the data is more appropriately modeled as panel data (rather than pooled). I therefore fit a panel Poisson model.[5] Formally, $P(Y_t = y_t|\lambda_t) = \frac{e^{-\lambda_t}\lambda_t{}^{y_t}}{y_t!}$.

Panel Negative Binomial

The benefits to using the Negative Binomial distribution are the same whether the data is modeled as pooled or panel. In order to account for the potential of overdispersion, I fit a panel data Negative Binomial model.

[5] A special type of the Poisson panel model is the Poisson Process. Fitting a Poisson Process with rate λ >0 to the data is an appropriate technique if: (1) no events have occurred at time t=0; (2) the independent increment assumption is valid. This assumes that the number of events in a given period is not correlated with the number of events in previous or later periods. The immigrant legislative data does not fit this assumption; (3) the stationary increment assumption is not violated. Stationary increments assume that the number of events in a given period of time only depends on the length of time and not the position of the period within a timeline. This assumption also does not hold with the immigrant legislation data; and (4) the probability of the number of events occurring in a given time period is equal to the Poisson distribution with parameter λ, time t, and events n, i.e. $P[N(s+t)-N(s)=n] = \frac{e^{-\lambda t}(\lambda t)^n}{n!}$, where the number of events are counted in increments of t, rather than pooled together.

Modeling of Observed Data

I model state-based immigrant-relevant legislation in the 2005 to 2011 time period as four model types: (1) Pooled cross-sectional count data through the Poisson distribution; (2) Pooled cross-sectional count data through the Negative Binomial distribution; (3) Poisson panel data with random effects; and (4) Negative Binomial panel data with random effects. Pro-/neutral and anti-immigrant state-year legislation are the two dependent variables. By using two separate dependent variables, I lower the threat to validity that my findings are an artifact of the particular data.

To illustrate the modeling strategy, I describe the model technique for the panel Poisson model (Model 3). The number of laws passed for state i in time period t is Y_{it}. I_{it} is the percent immigrant in a state-year, P_{it} the political party of the state-year, U_{it} the state unemployment rate in a given year, R_i the region of country, S_i a vector of time-constant variables including if the state is a Northern border state or a Southern border state, and V_i a vector of indicator variables for year. In the expression below c_i is the random effects term. Model 4 includes the same parameters but applies the data to the Negative Binomial distribution instead. Models 1 and 2 specify the same parameters, but do not include a time element.

$(Y_{it}|\lambda_{it}, I_{it}, P_{it}, U_{it}, R_i, S_i, V_i, c_i)$ has a Poisson distribution with parameter λ_{it} where $\log \lambda_{it} =$

$$B_0 + B_1 I_{it} + B_2 P_{it} + B_3 U_{it} + B_4 R_i + B_5 S_i + B_6 V_i + c_i \quad (3)$$

Analytic Strategies

I compare commonly used standard error specifications to the Monte Carlo simulation's standard error approximations. The Monte Carlo simulation generates synthetic data from each of the four model specifications (M1 through M4) to obtain good approximations to the true standard errors. I discuss the steps taken to create synthetic data for each of the model specifications.

Pooled Poisson

For brevity, I bundle the variables (i.e. parameters), represented by I_i, P_i, U_i, R_i, S_i and V_i, into the vector X_i. In the pooled Poisson model, I then assume $Y_i|X_i \sim Poisson\ (\lambda_i)$ where log λ_i is a linear function of the covariates. That is, conditioning on the independent variables, the number of pro-/neutral (or anti-) immigrant related laws passed by a state in each year from 2005 to 2011 are distributed as a Poisson random variable.

Given the observed data, I use Stata's predict command with the count option to predict the number of laws passed for each state in each year. Each predicted count represents the expected value of the mean and variance of the Poisson distribution for that state-year for the fitted model. In other words, I obtain the conditional expectation of the Poisson distribution for all 336 state-year observations, λ_θ.

To generate the synthetic data from these models and arrive at the Monte Carlo approximation to the true standard errors, I run the following set of steps (Steps 1-3) one thousand times.

1. First, I use a Poisson random data generator with λ_θ as the parameter. This yields randomly generated counts of immigrant legislation, $\widetilde{Y_i}$, for all 336 observations.

2. Second, I fit the randomly generated counts into a Poisson regression model, where $\widetilde{Y_i}|X_i$ has a Poisson distribution with parameter λ_i where $\log \lambda_i = \tilde{B}_0 + \tilde{B}_1 I_i + \tilde{B}_2 P_i + \tilde{B}_3 U_i + \tilde{B}_4 R_i + \tilde{B}_5 S_i + \tilde{B}_6 V_i$ (5)

The independent variables are defined in the same way as Model 3 above.

3. Third, Model 5 yields coefficients, $\tilde{B}_p$, for each parameter. I save these values for each of the thousand iterations.

I then implement a subsequent set of steps to analyze the validity of commonly implemented standard error techniques. These steps are described below.

4. I combine the $\tilde{B}_p$ into a new dataset that has one thousand observations. Each of the observations contains the vector of $\tilde{B}$ from one of the thousand iterations of the Monte Carlo simulation.

5. I then calculate the standard error between these values and the observed values from the original regression equation (Model 1) that contains the observed immigrant legislation counts. This yields the Monte Carlo standard error approximation, $SE_{\theta,p}$, for

each independent variable. I call this the 'true' standard error. Formally, for parameter p and iteration i,

$$SE_{\theta,p} = \sqrt{\left(\left(\sum_{i=1}^{1000}\left(\left(\widetilde{\beta_{pi}} - \beta_p\right)^2\right)\right)/1000\right)} \quad (6)$$

6. In order to compare the MC 'true' standard error to the standard error specifications most commonly utilized, I next obtain standard error estimates for each of the available standard error techniques pre-programmed in Stata. In pooled Poisson modeling there are seven available options: the observed information matrix (oim), robust standard errors, cluster (I cluster separately by state and by year), the outer product of the gradient (opg), a nonparametric bootstrap, and a jackknife procedure. Each of these standard error specifications is obtained by extracting the requisite information post-estimation from a pooled Poisson regression equation that specifies the relevant standard error. These models all resemble Model 1, except that the standard error specification varies. Once these standard error specifications are obtained, I am then able to judge the individual performance of each parameter as compared to their true standard error through the calculation of a relative error measure.

However, I am interested in the overall performance of the standard error, not the performance of the standard error for each parameter. I therefore create a summary measure that captures the relative error in the entire model for the commonly utilized standard error procedures as compared to the MC standard error. To do this, I average the relative error of each parameter. I do this for all seven of Stata's preprogrammed

specifications. Formally, the relative error (RE) for Stata standard error specification j with parameter p is

$$RE_j = \frac{\sum_{p=1}^{16}\left(\frac{SE_{j,p}-SE_{\theta,p}}{SE_{\theta,p}}\right)}{16} \quad (7)$$

This yields a measure that may be used to evaluate how well each pre-programmed standard error specification performs in pooled Poisson modeling as compared to the true standard error. I perform the entire sequence of steps (Steps 1-6) both for pro-/neutral and anti-immigrant legislation.

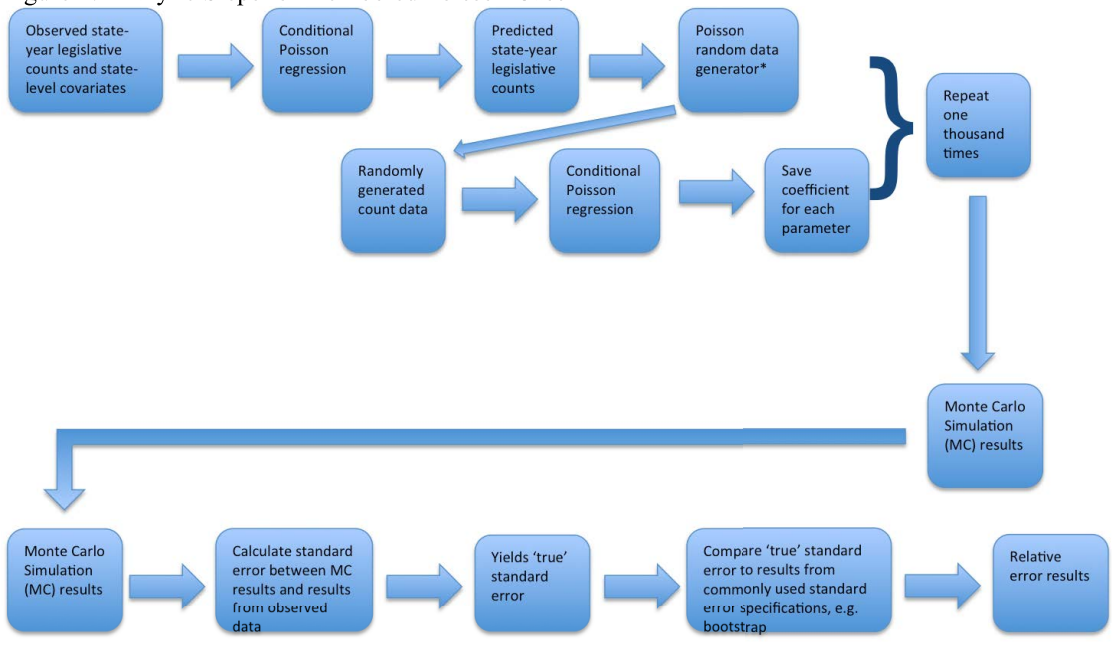
For each of the two types of immigrant legislation I also evaluate the performance of the standard error specifications for varying sample sizes. While the sample consists of 336 observations, I decrease the sample size in half to 168 observations and increase it two-fold (672), three-fold (1008), and five-fold (1680) to examine whether the best performing common standard error specification changes with the reduction or increase of the sample size. When the sample size is reduced, I do so by randomly dropping 168 observations.[6] For the analyses that require a sample size increase, I increase the sample by generating random counts of pro-neutral (or anti-) immigrant legislation with the same procedures used in Steps (1-3). Therefore, the independent variables' values remain the same for each state-year, but the counts of immigrant legislation vary. For example, in the 672 observation data, there are two instances of New Mexico in 2005, 2006, and through to 2011. The unemployment rate for both 2005 observations of New Mexico is the same, but the predicted number of immigrant legislation (pro-/neutral or anti-) differs.

[6] I randomly selected a seed order in Stata for this.

In the pooled Negative Binomial, panel Poisson, and panel Negative Binomial cases, I also decrease and increase the sample size. I use the appropriate steps for each of these modeling techniques (discussed below) to randomly generate data for the additional observations. Finally, for the analyses for increased sample sizes, not only do the sample sizes need to be increased for the MC simulation, but also for the regressions with the commonly applied standard error specifications. This is necessary in order to compare a MC standard error with 1,680 observations, for instance, to the common standard error specifications also with 1,680 observations. For each of the four model types, I use the same techniques to generate random data for these regressions as for the MC simulation.

Figure 1 details this entire sequence of steps. I next discuss the MC simulation procedure for the other three model types. I then discuss the results, limitations, and conclusions.

Figure 1. Analytic Steps for the Pooled Poisson Case



*Lambda defined as the observed counts

Pooled Negative Binomial

I fit the Negative Binomial distribution to the data. There are also seven standard error specifications applicable for the legislative data in Stata's negative binomial regression – oim, robust, opg, nonparametric bootstrap, jackknife, cluster state, and cluster year. I run one thousand iterations of the synthetic data and then the subsequent steps necessary to obtain the relative error results for each of the seven standard error specifications. In this way the steps taken to obtain the results are strikingly similar to the pooled Poisson model specification outlined above.

What differs is how the parameter(s) of the theoretical distribution are obtained. That is, the parameter(s) of the distribution from which the random counts of legislative immigration are generated in the Monte Carlo simulation.

The negative binomial has two parameters, but these parameters may be represented in different forms. For instance, the mean (μ) and the variance (σ^2) can define the negative binomial distribution, but so too can the parameters r (here the inverse of the overdispersion parameter) and p (here the probability of failure for a given trial), as discussed above, where $\mu = \frac{(1-p)r}{p}$ and $\sigma^2 = \frac{(1-p)r}{p^2}$. Solving in terms of r and p, $r = \frac{\mu^2}{\sigma^2-\mu}$ and $p = \frac{\mu}{\sigma^2}$. Solving back for the variance, $\sigma^2 = \mu + \frac{u^2}{r}$.

The mean is derived directly from the model, just as in the pooled Poisson case. Here the conditional distribution of the state-year incidence of pro-neutral (or anti-) immigration laws is defined as $Y_i|X_i \sim NB\left(e^{\sum_j \beta_j X_{ij}}, \alpha\right)$. The predicted counts for each state-year therefore represent μ.

The pooled negative binomial model reports an estimate of the overdispersion parameter, here denoted as α. Because r may also be defined as the inverse of this parameter, r is also equal to $1/\alpha$.

Using the predicted values of μ and deriving r from $\frac{1}{\alpha}$, I obtain the variance of the negative binomial distribution. With the mean and variance, I then derive p, as per the equation above.

With values of r and p, I have the parameters necessary to generate new negative binomial random variables for each of the thousand iterations of the Monte Carlo simulation.[7] As discussed, all subsequent steps to derive the relative error results are the same as those in the pooled Poisson modeling (Steps 3-6 in the previous section).

Panel Poisson

Due to the nested structure of the data (state-year within state), I also model the immigrant legislation data as panel rather than pooled. Formally, $Y_{it}|X_{it} \sim Pois\ (e^{\sum_j \beta_j X_{jit} + \gamma_i})$, where e^{γ_i} represents a random state effect. It is assumed to be independent and identically distributed (i.i.d.) and derived from a gamma distribution with a mean of one and a variance of alpha, where alpha is estimated directly from the data.[8]

The steps necessary to derive the relative error results change slightly when estimating the data as panel Poisson rather than pooled Poisson.

[7] Using the rnbinomial command. While there are multiple ways to define the parameters of the negative binomial distribution, and therefore multiple ways to sample from it, Stata specifies the negative binomial according to the specifications discussed above. Proof is available upon request.
[8] The xtpoisson command reports an estimate of alpha in the output.

1. Similar to the pooled models, I derive predicted counts of the number of pro-/neutral (or anti-) immigrant legislation passed in a given state in a given year.[9]

2. Unlike in the pooled specification, a random state effect, e^{γ_i}, must be incorporated into the random generation of Poisson data. The random state effect is assumed to be gamma distributed with mean 1.[10] The gamma distribution is defined by two parameters, the shape (k) and scale (l) parameters. The mean and variance of the gamma distribution are defined in terms of the shape and scale parameters by $\mu = k * l$ and $\sigma^2 = k * l^2$. As $\mu = 1$ and $\sigma^2 = \alpha$, where alpha is estimated directly from the data, solving for k and l yields $k = 1/\alpha$ and $l = \alpha$. These two parameters are required to draw random data from the gamma distribution in Stata. I draw gamma random variables for each state to model a state random effect.

3. To incorporate the generated state random effect into the model, the Poisson parameter is multiplied by the state random effect.

4. I generate Poisson random variables using the new legislative counts derived in Step 3 as the parameter of the Poisson distribution.

5. These randomly generated counts are fit to a panel Poisson regression model,

$$Y_{it} \sim Pois\ (e^{\tilde{B}_0 + \tilde{B}_1 I_{it} + \tilde{B}_2 P_{it} + \tilde{B}_3 U_{it} + \tilde{B}_4 R_i + \tilde{B}_5 S_i + \tilde{B}_6 V_i}) \quad (8)$$

[9] I use the predict, nu0 command in Stata to obtain the predicted counts.
[10] As discussed in Stata's xt manual for Poisson panel regression. See the xtpoisson entry.

Steps 2-5 are run for each of the thousand iterations of the Monte Carlo simulation. I then follow the same procedure discussed in Steps 3-6 of the pooled Poisson case to derive the relative error results. However, in panel data (either Poisson or Negative Binomial) the only standard error specifications available are the observed information matrix, nonparametric bootstrap, and jackknife. Due to the random nature of the nonparametric bootstrap standard error specification, I create five separate sets of nonparametric bootstrap results to compare against the MC approximation. I do this both for the panel Poisson and for the Negative Binomial model types.

Panel Negative Binomial

The panel Negative Binomial modeling type is the most intricate of the four. In fact, although I utilize a Negative Binomial model for the Monte Carlo simulation, the procedure calls for the use of Poisson random variables rather than Negative Binomial random variables to generate the immigration law counts.[11] Formally,

$$Y_{it}|X_{it} \sim Pois(\gamma_{it}) \quad (9)[12]$$

$\gamma_{it} \sim \Gamma(\lambda_{it}, \delta_i)$, where log λ_{it} is the linear expression of the covariates, δ_i is the random state component, and Γ denotes the gamma distribution. (10)[13]

[11] I do this, following the discussion in Stata's xt manual. For further information, see the Methods and formulas section for the xtnbreg command.

[12] I replace all values of γ_{it} that are less than .00001 with a value of .00001. I do this because otherwise the generation of Poisson random variables yields missing values, at least through the use of Stata's rpoisson command.

[13] The gamma distribution is unable to account for values of zero in either of its parameters. Because λ_{it}, the observed number of laws passed, does contain zero values, the random draws from the gamma distribution yield missing data. To deal with this issue, I substitute the state-year observations with a value of 0 with a randomly generated value from a uniform distribution that ranges from .4 to .6.

$\frac{1}{1+\delta_i} \sim Beta(r, s)$, where r and s are estimated directly from the data. (11)

Working backwards from equation 11 to equation 9, I obtain randomly generated estimates of Negative Binomial data. These values are used in a Negative Binomial panel data regression equation with random effects to obtain parameters for each independent variable.

This entire sequence of steps is implemented one thousand times, creating the Monte Carlo creation of synthetic data. I then follow Steps 3-6 from the pooled Poisson section to derive the relative error results.

Results

Pooled Results

Table 1 reports the pooled pro-/neutral immigrant legislation relative error results for the observed information matrix, robust standard error, outer product of the gradient, state and year clustering, nonparametric bootstrap, and jackknife. Table 2 reports results of the pooled anti-immigrant legislation analyses.

In the original data (336 observations), the observed information matrix outperforms the other standard error specifications in pooled Poisson and pooled Negative Binomial models, for both Pro-/Neutral and Anti-Immigrant Legislation. For instance, the 2nd column in the left panel of Table 1 describes results from the pooled Poisson modeling of pro-/neutral immigration legislation for the normal sized data. The relative error of the observed information matrix is .021, which is smaller than the next

closest relative error of .196 (the outer product of the gradient). Indeed, the observed information matrix performs best in three of these four model types as well when the sample size is reduced in half. The exception is for the negative binomial model of anti-immigrant legislation (1/2 size sample, right panel of Table 2), where all techniques (other than clustering by state or year) perform better than the OIM.

When the sample size is increased to two times, three times, or five times the actual size in the pooled Poisson models (left panels of Tables 1 and 2), the jackknife performs either best (four times) or second best (two times) for both pro-/neutral and anti-immigration laws. As an example, the relative error of the jackknife for the twofold sample (672 observations) in the left panel of Table 1 is .204, smaller than the oim's .226. The observed information matrix, in fact, is consistently outperformed by a number of the other techniques, including robust standard errors.

Table 1. Pooled pro-/neutral immigrant legislation relative error results

	Pro- and Neutral Immigration Laws									
	Poisson					Negative Binomial				
Standard Error Specification	1/2 size	Normal	2x	3x	5x	1/2 size	Normal	2x	3x	5x
Observed Information Matrix	0.044	0.021	0.226	0.281	0.311	0.125	0.031	0.150	0.141	0.191
Robust	0.298	0.308	0.208	0.270	0.275	0.179	0.071	0.171	0.168	0.225
Outer Product of the Gradient	0.240	0.196	0.298	0.301	0.344	0.179	0.101	0.171	0.152	0.172
Nonparametric Bootstrap	0.417	0.427	0.196	0.304	0.279	0.234	0.106	0.213	0.199	0.209
Jackknife	0.467	0.413	0.204	0.265	0.272	0.199	0.112	0.151	0.154	0.210
Cluster State	0.462	0.543	0.226	0.338	0.391	0.267	0.183	0.207	0.196	0.232
Cluster Year	0.466	0.499	0.542	0.536	0.478	0.427	0.422	0.548	0.457	0.509
N	168	336	672	1008	1680	168	336	672	1008	1680

Table 2. Pooled anti-immigrant legislation relative error results

	Anti-Immigration Laws									
	Poisson					Negative Binomial				
Standard Error Specification	1/2 size	Normal	2x	3x	5x	1/2 size	Normal	2x	3x	5x
Observed Information Matrix	0.167	0.043	0.276	0.403	0.302	0.285	0.038	0.145	0.150	0.227
Robust	0.190	0.159	0.197	0.340	0.284	0.281	0.051	0.153	0.151	0.201
Outer Product of the Gradient	0.212	0.144	0.322	0.447	0.310	0.223	0.089	0.181	0.137	0.238
Nonparametric Bootstrap	1.317	0.170	0.211	0.379	0.321	0.218	0.186	0.146	0.163	0.244
Jackknife	0.260	0.262	0.185	0.322	0.276	0.197	0.067	0.133	0.129	0.190
Cluster State	0.386	0.405	0.266	0.320	0.311	0.317	0.219	0.189	0.215	0.276
Cluster Year	0.451	0.560	0.563	0.585	0.574	0.493	0.490	0.392	0.438	0.497
N	168	336	672	1008	1680	168	336	672	1008	1680

There is not a consistent pattern across the pooled Negative Binomial results when the sample size is increased (right panels of Tables 1 and 2). For Pro-/neutral immigrant legislation, the observed information matrix continues to perform best (Table 1, right panel). For example, the relative error for the oim in the triple-sized sample is .141 for pro-/neutral immigrant laws. The next closest is the outer product of the gradient, with a relative error of 152. That said, the outer product of the gradient, performs best when the sample size increases to five times the actual sample size. In contrast, for the anti-immigration data, the jackknife performs best across all three increased sample sizes.

To summarize, the observed information matrix performs best in the original sample size and reduced sample size across both model types and both sets of data. When the sample size increases, the jackknife technique performs particularly well, especially in pooled Poisson data.

Panel Results

The Poisson and Negative Binomial panel data analyses compare the observed information matrix, nonparametric bootstrap, and jackknife to the Monte Carlo approximation for pro-/neutral immigrant legislation and anti-immigrant legislation. Due to the random nature of the nonparametric bootstrap, these results are averaged across five independent iterations. Results are reported in Table 3.

While I attempted to perform the analyses described above for panel Poisson and panel Negative Models with Pro-/Neutral and Anti-immigration legislation, I experienced a great amount of difficulty with the Negative Binomial panel modeling. In fact, I could

not perform the Monte Carlo simulation under the Negative Binomial specification when I cut the sample size in half for either Pro-/Neutral or Anti-immigrant legislation. Nor did the procedure run with the regular sample size for anti-immigrant legislation. More specifically, at a relatively early point within the one thousand iterations of the Monte Carlo simulation the model would fail to converge. Attempts to skip a single bad run and proceed with further iterations were unsuccessful - a random number of iterations later the procedure also failed to converge. Multiple attempts all yielded the same results.

At the same time, the relative error comparisons are contingent on comparing the commonly used standard error specifications to the MC approximation. The nonparametric bootstrap, a common technique easily applied in Stata, did not converge for the regular sample size, two times the sample size, or three times the sample size for the pro-/neutral data for the Negative Binomial model. It did not converge for the anti-immigrant legislation for two times, three times, or five times the regular sample size, either. In fact, the only panel Negative Binomial model in which I was able to compare the commonly used standard error techniques to the MC approximation was for five times the regular sample size in the pro-/neutral data. I therefore do not report the results for the panel Negative Binomial model and focus solely on the results for the panel Poisson model. Also note that the MC simulation did not run for the one half sample size for either the pro-/neutral or anti-immigrant legislation datasets in the panel Poisson models.

Table 3. Panel Poisson immigrant legislation relative error results

	Pro- and Neutral Immigration Laws					Anti-Immigration Laws				
		Poisson					Poisson			
Standard Error Specification	**1/2 size**	**Normal**	**2x**	**3x**	**5x**	**Normal**	**1/2 size**	**2x**	**3x**	**5x**
True Standard Error										
Observed Information Matrix	N/A	0.173	0.185	0.251	0.292	N/A	0.211	0.167	0.253	0.305
Nonparametric Bootstrap (1)	N/A	0.301	0.200	0.249	0.308	N/A	0.200	0.146	0.279	0.364
Nonparametric Bootstrap (2)	N/A	0.201	0.174	0.235	0.313	N/A	0.251	0.195	0.262	0.311
Nonparametric Bootstrap (3)	N/A	0.243	0.232	0.288	0.269	N/A	0.243	0.227	0.265	0.327
Nonparametric Bootstrap (4)	N/A	0.245	0.241	0.277	0.318	N/A	0.266	0.177	0.264	0.349
Nonparametric Bootstrap (5)	N/A	0.273	0.200	0.283	0.292	N/A	0.219	0.219	0.292	0.334
Bootstrap Average	N/A	0.253	0.210	0.267	0.300	N/A	0.236	0.193	0.272	0.337
Jackknife	N/A	0.207	0.201	0.250	0.287	N/A	0.217	0.179	0.245	0.317
N	168	336	672	1008	1680	168	336	672	1008	1680

In both data types the observed information matrix performs better than the average of the five nonparametric bootstraps or the jackknife for the regular data size. It also performs better when the sample size is doubled. For instance, the relative error for the oim in the right panel of Table 3 for the double-size data is .167, which is better than the relative error of the bootstrap average at .193 or the jackknife relative error of .179.

The jackknife procedure, on the other hand, performs best three of the four times when the sample size is increased to three or five times the regular size. The relative error of the jackknife for the five-fold sample for pro/neutral immigration laws is .287 (left panel of Table 3) whereas the oim is .292. When the sample size is tripled the relative errors are .250 and .251 for the jackknife and oim specifications, respectively. Although these differences are not large, the outperformance of the jackknife parallels the results in the pooled specification.

Finally, there is a great amount of variability in the relative error of the nonparametric bootstrap procedure. This is to be expected as the nonparametric bootstrap by definition contains a random component.

Discussion

In this research I evaluate the accuracy of commonly used standard error techniques. While researchers often describe the logic behind the derivation of their model fit, the validity of the standard error specification is less commonly questioned. I analyze real world count data of pro-/neutral and anti-immigrant legislation laws to assess the validity of standard error techniques. I do so by comparing these techniques to a Monte Carlo simulation of synthetic data.

I find that the observed information matrix, which is the default standard error technique in Stata, does an excellent job approximating the true standard error, especially with small sample sizes. With only 336 observations and sixteen covariates, the data is stretched thin, especially for panel data specifications with only 48 level-2 observations. At this size and a ½ sample size the observed information matrix performs best. The jackknife method performs particularly well also, especially in large sample sizes. In fact, the jackknife outperforms the oim when the sample is doubled, tripled, or multiplied fivefold.

Conclusions

Methodological Contributions

This research contributes in two key ways. First, it describes a method in which the validity of standard errors may be tested. By describing the steps necessary to create a Monte Carlo simulation for pooled and panel Poisson and Negative Binomial modeling, and then outlining the relative error assessments, I provide researchers a means in which to evaluate the validity of the standard errors in their own research.

Second, the findings suggest that the jackknife performs especially well. Both in pooled data, with a plethora of techniques to choose from, and in panel data, where the options are more limited, the jackknife procedure stands out as a good approximation to the 'true' standard error. Researchers interested in testing alternative standard error specifications should begin with the jackknife.

Implications

The goal of this research is to inform social scientists as to the validity of commonly used standard error techniques. The observed information matrix performs particularly well, even in small sample sizes. This finding should give researchers additional confidence in their model fit of count data.

At the same time, the variability associated with the nonparametric bootstrap technique (as described in Table 3), should caution researchers from concluding results based upon a single run of any model specified with nonparametric bootstrap standard errors. Instead, if the nonparametric bootstrap technique is of theoretical interest for the research project, results should be averaged across a number of iterations.

Finally, the jackknife method should be considered a viable alternative to other standard error specifications if researchers are interested in conducting sensitivity checks.

Limitations

The size of the dataset is of potential concern. Although I increase the sample size to evaluate the results under larger samples in which the data is less strained, I only vary the counts of immigration laws. I do not randomly vary the covariates. For instance,

I do not randomly select a state's foreign-born percentage. This may restrict the variability of the increased samples. At the same time, several of the covariates overlap. In future work the number of covariates should be reduced to eliminate multicollinearity concerns. This may help the nonconvergence issue that prevented the analysis of the panel Negative Binomial data and panel Poisson model of the ½ sample.

Although I conduct analyses on two separate datasets - pro/neutral and anti-immigrant legislation, these datasets are not entirely independent as states that pass one law type should be less likely to pass the other. To further evaluate the validity of standard error techniques in count data, independent datasets of larger size should be utilized.

Appendix A. Monte Carlo Simulation

I include an example of the Stata code for generating the Monte Carlo simulation. The example is taken from the panel Poisson case of pro/neutral immigrant legislation.

```
*====================================================================

cd "~/Documents/Johns Hopkins/AMS/AMS Master's Paper/Stata/bootstrap/"

log using bootstrap_xt_poisson_positivelaws.log, replace

clear
clear matrix
*set mem 300m

set pagesize 300
set more off, permanently



*====================================================================
use bootstrap-prep

sort stateid year
xtset stateid year
xtsum stateid year plawcount plaws party slavestate sbstate nbstate
region percentfb  unemp

*preserve
d
codebook id
keep id stateid year state plaws alaws tlaws ulaws tres pres ares
plawcount alawcount tlawcount ///
party dem repub mixedother region northeast south midwest west
slavestate sbstate nbstate percentfb unemp ///
governor house senate year* ///
mt mp ma ///
pp pa pt ///
nbp nbp_r nbp_p ///
nba nba_r nbp_p ///
nbt nbt_r nbt_p ///
xtpp xtpa xtpt ///
xtnbp xtnbp_r xtnbp_p ///
xtnba xtnba_r xtnba_p ///
xtnbt xtnbt_r xtnbt_p


global Year year2 year3 year4 year5 year6 year7
d $Year
poisson plaws slavestate sbstate nbstate repub mixedother south midwest
west percentfb unemp $Year
est store p
nbreg plaws slavestate sbstate nbstate repub mixedother south midwest
west percentfb unemp $Year
```

```
est store nb
xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, irr re vce(oim)
est store xtp
xtnbreg plaws slavestate sbstate nbstate repub mixedother south midwest
west percentfb unemp $Year, irr re vce(oim)
est store xtnb

est table p nb xtp xtnb
est stats p nb xtp xtnb



*=====================================================================
*XT Poisson parameter estimation for pro/neutral legislation
*=====================================================================
*Begin with xtpp
*but need to incorporate random effects component into the POISSON
panel data parametric bootstrap

*Steps:

*1. Fit model to actual data
*   predict Y-hat's. i.e. the predicted counts (e.g. xtpp, xtpa)

*2. Capture alpha from the data


*For each of the 1,000 iterations (steps 3-6):

*3. Generate epsilon for each state
*   where epsilon ~ Gamma w/ mean 1 and variance = alpha = 1/theta
*   theta is calculated from the data

*4. Create lambda's with multiplicative effect
*   epsilon * Y-hat = lambda


*5. With these I can generate Yboot, i.e. rpoisson(lambda)
*   this gives 336 y's

*6. Fit model to get Beta-boot



*=====================================================================
*STEPS for incorporating random effects component into the POISSON
panel data parametric bootstrap by adding in random gamma draws
*=====================================================================

/*
Yit | Xit ~ Poisson (Lambda-it)
where Lambda-it = exp (Xit*B)
in the random effects model, there is a random component that
is derived from the gamma distributon
```

```
Ei ~ Gamma (1, 1/theta)
see the technical documentation on xtpoisson
*/

*What does a poisson distribution look like?
d, s
sum xtpp
preserve
keep xtpp plaws
set obs 336
gen double pois=rpoisson(xtpp)
sum pois xtpp plaws
restore


/*
      Run the entire sequence 1,000 times.
*/



*1. How do I generate the predicted counts?
d xtpp
sum xtpp
xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, irr re vce(oim)
predict xtpp_alt, nu0
corr xtpp xtpp_alt
drop xtpp_alt
      *This is how I derived xtpp
      *There's no need to start with the predicted counts



*2. Working backwards, Generate alpha, the variance of the gamma
distribution
xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, irr re vce(oim)
      *This equation also gives me 'alpha'
      *In stata, the standard random-effects model, v, is assumed to be
i.i.d.
      *such that exp(v) is gamma with mean one and variance alpha.
      *Alpha is estimated from the data
      *This gives us alpha
gen alpha=e(alpha)
tab alpha



*I want random effects, epsilon, for each state
*where epsilon ~ gamma (1, 1/theta) and 1/theta = alpha


*Knowing the mean(1) and variance(alpha), estimate the gamma
distribution parameters
      *They are: alphastar and thetastar
```

```
      *Note: alpha and alphastar are not the same.
      /*
      mean = alphastar * thetastar
      variance = alphastar * (thetastar)^2

      In stata mean = 1 and variance=alpha
      Thus,
      we get thetastar=alpha
      we get alphastar= 1/alpha
      */

gen thetastar=alpha
gen alphastar=1/alpha
      *Now I have the parameters for the gamma distribution
sum alphastar thetastar

*A Check:
      *Simple Model
xtpoisson plaws
gen testalpha=e(alpha)
gen testtheta=1
gen testalphastar=1/testalpha
gen testthetastar=testalpha

*for the mean:
display testalphastar*testthetastar
      *=1, which is what stata assumes

*for the var:
est
display testalphastar*(testthetastar^2)
      *=.46705086
      *This is the same as the alpha output from the xtpoisson
regression
      *It is correct as the alpha output represents the variance of the
gamma distribution



*Steps 3-6 must be for all 1,000 iterations because of the random draw
in step 3


forval i=1/1000 {

*3. Generate Epsilon
*     Epsilon-i ~ Gamma (1, 1/theta)


sort stateid
*keep if stateid!=stateid[_n-1]
gen double epsilon`i' = rgamma(alphastar, thetastar) if
stateid!=stateid[_n-1]
sum epsilon`i'
codebook epsilon`i'
qui:list stateid epsilon`i'
```

```
replace epsilon`i'=epsilon`i'[_n-1] if stateid==stateid[_n-1] &
epsilon`i'==.
sum epsilon`i'
codebook epsilon`i'
*tab epsilon`i'
qui:list stateid epsilon`i'


*A test to see if this is correct:
gen k=7
gen th=.5
*set seed 17325
gen double test=rgamma(k, th)
qui: tab test
sum test
*Is this correct?
      *mean reported is 3.588 [will change unless set seed #]
      *The mean from the gamma is equal to k*th
display k*th
      *= 3.5
      *3.588 is close to true value of 3.5!
      *The standard deviation reported is 1.268 [will change unless set
seed #]
      *The variance from the gamma is equal to k * th^2
display sqrt(k*th*th)
      *=1.323
      *1.268 is close to true value of 1.323!
drop k th test



*4. Generate rescaled lambda's, rescaled by the state-level random
effect
*     Lambda-it = exp (Xit*B)

gen lambda`i' = epsilon`i' * xtpp
sum lambda`i'
codebook lambda`i'


*5. Generate rpoisson using lambda
*     Yit | Xit ~ Poisson (Lambda-it)

gen double rxtpp`i' = rpoisson(lambda`i')
      sum rxtpp`i'
      corr xtpp rxtpp`i'
      sum plaws xtpp rxtpp`i'
      corr plaws pp xtpp rxtpp`i'


*6 Bootstrapping

*Note: The poisson distribution only has one parameter - lambda.
*Here it is xtpp_gamma
      *xtpp_gamma is the parameter after I introduce random effects by
state for the parametric bootstrap.
      *Without this the parameter is simply xtpp
```

```
corr plaws plawcount
d

*A test:
generate double rxtpp = rpoisson(lambda`i')
xtpoisson rxtpp slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, vce(oim)
corr rxtpp xtpp pp plaws
xtpoisson rxtpp slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, vce(oim)
drop rxtpp


*preserve
xtpoisson rxtpp`i' slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, vce(oim)
*estimates store rpooledboot`i'
matrix list e(b)
display _b[_cons]
display _b[repub]
display _b[mixedother]
gen constantboot=_b[_cons]
gen repubboot= _b[repub]
gen mixedotherboot=_b[mixedother]
gen slavestateboot=_b[slavestate]
gen sbstateboot=_b[sbstate]
gen nbstateboot=_b[nbstate]
gen southboot=_b[south]
gen midwestboot=_b[midwest]
gen westboot=_b[west]
gen percentfbboot=_b[percentfb]
gen unempboot=_b[unemp]
gen year2boot=_b[year2]
gen year3boot=_b[year3]
gen year4boot=_b[year4]
gen year5boot=_b[year5]
gen year6boot=_b[year6]
gen year7boot=_b[year7]

d *`i'
sum *`i'
drop rxtpp`i'
codebook id
preserve
keep if id==1
keep id constantboot repubboot mixedotherboot slavestateboot
sbstateboot nbstateboot southboot ///
midwestboot westboot percentfbboot unempboot year2boot year3boot
year4boot year5boot ///
year6boot year7boot
d *boot
sum *boot
save boot`i', replace
*This way only have 1 observation per dataset
restore
drop constantboot repubboot mixedotherboot slavestateboot sbstateboot
```

```
nbstateboot southboot ///
      midwestboot westboot percentfbboot unempboot year2boot year3boot
year4boot year5boot ///
      year6boot year7boot ///
      epsilon`i' lambda`i'
      *drops the created variables so that they are not stored for the
next iteration of the loop - don't need them since already saved above
}
*restore

d
sum
d, s
save bootstrap_pre-append, replace



*==================
*appending data

clear
use boot1
d
sum
codebook id
save bootstrap-prelim, replace

d
forval j=2/1000 {
clear
clear matrix
clear mata
use bootstrap-prelim
append using boot`j'

d constant* repub* mixedother*
sum constant* repub* mixedother*
keep *boot id
codebook id
compress
d
save bootstrap-prelim, replace
}
*restore
*A check:
      tab1 constantboot percentfbboot year7boot
      preserve
      forval i=1/1000 {
      use boot`i'
      sum constantboot percentfbboot year7boot
      }
      restore

sum
sum id
rename id idold
gen id=[_n]
```

```
list id idold constant* repub* mixed* slave* sb* nb*
drop idold
sum *boot
d, s

save bootstrap_xt_poisson_positivelaws.dta, replace


*=====================================================================
*For comparison to observed XT Poisson positive/neutral data
*=====================================================================
clear
cd "~/Documents/Johns Hopkins/AMS/AMS Master's Paper/Stata/bootstrap/"
use bootstrap-prep


poisson plawcount slavestate sbstate nbstate repub mixedother south
midwest west percentfb  unemp $Year
est store m1
poisson plawcount slavestate sbstate nbstate repub mixedother south
midwest west percentfb  unemp $Year, vce(oim)
est store m2
poisson plawcount slavestate sbstate nbstate repub mixedother south
midwest west percentfb  unemp $Year, vce(robust)
est store m3
poisson plawcount slavestate sbstate nbstate repub mixedother south
midwest west percentfb  unemp $Year, vce(cluster state)
est store m4
poisson plawcount slavestate sbstate nbstate repub mixedother south
midwest west percentfb  unemp $Year, vce(cluster year)
est store m5
poisson plawcount slavestate sbstate nbstate repub mixedother south
midwest west percentfb  unemp $Year, vce(opg)
est store m6
poisson plawcount slavestate sbstate nbstate repub mixedother south
midwest west percentfb  unemp $Year, vce(bootstrap)
est store m7
poisson plawcount slavestate sbstate nbstate repub mixedother south
midwest west percentfb  unemp $Year, vce(jackknife)
est store m8
est table m1 m2 m3 m4 m5 m6 m7 m8

est table m1 m2 m3 m4 m5 m6 m7 m8, se stats(N r2 r2_a)


ereturn display
estimates dir

xtpoisson plawcount slavestate sbstate nbstate repub mixedother south
midwest west percentfb  unemp $Year, re irr vce(oim)
xtpoisson plawcount slavestate sbstate nbstate repub mixedother south
midwest west percentfb  unemp $Year, irr vce(bootstrap)
xtpoisson plawcount slavestate sbstate nbstate repub mixedother south
midwest west percentfb  unemp $Year, irr vce(jackknife)


*=================================
```

```
d
sum


log close
translate bootstrap_xt_poisson_positivelaws.log "~/Documents/Johns
Hopkins/AMS/AMS Master's
Paper/Stata/bootstrap/bootstrap_xt_poisson_positivelaws.smcl",
linesize(79) translator(smcl2log) replace
```

Appendix B. Coding of the Analysis

I include an example of the Stata code for generating the relative error results of the various standard error specifications. The example is taken from the panel Poisson case of pro/neutral immigrant legislation.

```
*=====================================================================
* Goal:
      *Compare the standard error from the parametric bootstrap to the
various standard error options that are
      *pre-programmed in stata [i.e vce()]
            *Recall that the parametric bootstrap takes 1,000 samples
of coefficients w/ dep. var. from random draw of poisson or NB
distribution
            *For poisson lambda is taken as the predicted counts per
state per year from the original data
            *For NB defining r and p is more labor intensive
      *Assuming the parametric bootstrap is the true value for the SE,
which of the pre-programmed options performs closest?

      *Note I am comparing the MSE for each parameter [parameter here
is defined as the coef. for each covariate]
      *there are 16 of these.  As the mean does not change, all the
action in the MSE is the variance, therefore I compare SE's.

*=====================================================================

cd "~/Documents/Johns Hopkins/AMS/AMS Master's
Paper/Stata/Results/xt_poisson_positive/"

log using results_xt_poisson_positivelaws.log, replace

clear
clear matrix
set mem 300m

set pagesize 300
set more off, permanently


*=====================================================================
*=====================================================================
cd "~/Documents/Johns Hopkins/AMS/AMS Master's Paper/Stata/bootstrap/"
use bootstrap-prep
d, s

append using bootstrap_xt_poisson_positivelaws
cd "~/Documents/Johns Hopkins/AMS/AMS Master's
Paper/Stata/Results/xt_poisson_positive/"
d, s
d
sum
drop nbregrate nbregp* nbregc* mpstate*
```

```
*====================================================================
*For comparison to observed xt poisson positive laws data
*====================================================================
xtset id year
xtsum plaws slavestate sbstate nbstate repub mixedother south midwest
west percentfb unemp $Year

global Year year2 year3 year4 year5 year6 year7
d $Year

xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re
est store default_long1x
est save default_long1x, replace
xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re vce(oim)
est store oim_long1x
est save oim_long1x, replace



xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re vce(bootstrap)
est store bootstrap_long1x_1
est save bootstrap_long1x_1, replace
xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re vce(bootstrap)
est store bootstrap_long1x_2
est save bootstrap_long1x_2, replace
xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re vce(bootstrap)
est store bootstrap_long1x_3
est save bootstrap_long1x_3, replace
xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re vce(bootstrap)
est store bootstrap_long1x_4
est save bootstrap_long1x_4, replace
xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re vce(bootstrap)
est store bootstrap_long1x_5
est save bootstrap_long1x_5, replace

xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re vce(jackknife)
est store jackknife_long1x
est save jackknife_long1x, replace
est table default_long1x oim_long1x /*robust clstate clyear opg*/
bootstrap_long1x_1 bootstrap_long1x_2 bootstrap_long1x_3
bootstrap_long1x_4 bootstrap_long1x_5 jackknife_long1x


est table default_long1x oim_long1x /*robust clstate clyear opg*/
bootstrap_long1x_1 bootstrap_long1x_2 bootstrap_long1x_3
bootstrap_long1x_4 bootstrap_long1x_5 jackknife_long1x, se stats(N r2
r2_a)
```

```
est table default_long1x oim_long1x /*robust clstate clyear opg*/
bootstrap_long1x_1 bootstrap_long1x_2 bootstrap_long1x_3
bootstrap_long1x_4 bootstrap_long1x_5 jackknife_long1x, star stats(N r2
r2_a)
      *coefficients are the same, the SE's are not
est table default_long1x oim_long1x /*robust clstate clyear opg*/
bootstrap_long1x_1 bootstrap_long1x_2 bootstrap_long1x_3
bootstrap_long1x_4 bootstrap_long1x_5 jackknife_long1x, p stats(N r2
r2_a)


ereturn display
estimates dir


*=====================================================================



*Intentionally Blank



*=====================================================================
*Summary Statistics for bootstrapped data


global X0 slavestate sbstate nbstate repub mixedother south midwest
west percentfb unemp year2 year3 year4 year5 year6 year7

foreach i of varlist $X0 {
egen `i'boot_m=mean(`i'boot)
egen `i'boot_sd=sd(`i'boot)
gen `i'boot_v=(`i'boot_sd)^2
egen `i'boot_std=std(`i'boot)
}
d *_m *_sd *_v *_std
sum *_m *_sd *_v *_std
      *Each standardized variable has a mean of 0 and a variance of 1

*Note that each value for _m, _sd_and _v are the same for EVERY
observation because they represent
      *parameters of the distribution.
*but _std varies because the observations are standardized.  They still
vary.
*Therefore, I can't inculde _std in the matrix.


egen slavestateboot_m_alt=mean(slavestateboot)
egen slavestateboot_sd_alt=sd(slavestateboot)
gen slavestateboot_v_alt=slavestateboot_sd^2
egen slavestateboot_std_alt=std(slavestateboot)
sum slavestateboot_m slavestateboot_m_alt
sum slavestateboot_sd slavestateboot_sd_alt
sum slavestateboot_v slavestateboot_v_alt
sum slavestateboot_std slavestateboot_std_alt
      *The same!
```

```
drop *_alt


d *boot*
sum *boot*



*==========
*Matrix Creation for parametric bootstrapped data as compared to means
and standard errors of regressions w/ various SE specifications
*==========
*Trivial Case:
matrix dumbo = [1, 2 \3, 4]
matrix list dumbo
matrix drop dumbo

*Grab Original Data?
      *No! - Because are interested in estimating the coefficients, not
the values of the observed data
sum slavestate
      *original data
sum slavestateboot
      *results from parametric bootstrap



d $X0
*1. mean 2. variance 3. sd
foreach i of varlist $X0 {
d `i'boot*
sum `i'boot*
mkmat `i'boot_m `i'boot_v `i'boot_sd if id==336, matrix(`i'_boot)
matrix `i'_boot=`i'_boot'
matrix list `i'_boot
}


global X0 slavestate sbstate nbstate repub mixedother south midwest
west percentfb unemp year2 year3 year4 year5 year6 year7
      *global X0 slavestate

global X1 oim_long1x /*robust opg*/ bootstrap_long1x_1
bootstrap_long1x_2 bootstrap_long1x_3 bootstrap_long1x_4
bootstrap_long1x_5 jackknife_long1x /*state year*/
global X1a oim_long1x /*robust opg*/ bootstrap_long1x_1
bootstrap_long1x_2 bootstrap_long1x_3 bootstrap_long1x_4
bootstrap_long1x_5 jackknife_long1x
*global X1b state year
      *global X1 oim


*drop m_* sd_* v_*
*A test:
      *1. mean 2. variance 3. sd
foreach i in oim {
xtpoisson plaws slavestate sbstate nbstate repub mixedother south
```

```
midwest west percentfb unemp $Year, re vce(`i')
est store `i'_long1x
est save `i'_long1x, replace
*matrix list e(b)
matrix B=e(b)
matrix list B
*matrix list e(V)
matrix V=e(V)
matrix Var=vecdiag(V)
matrix list Var
      foreach j of numlist 1/1 {
      gen m_`j'_`i'=B[1,`j']
      *display m_`j'_`i'
      gen v_`j'_`i'=Var[1,`j']
      *display v_`j'_`i'
      *display sqrt(v_`j'_`i')
      gen sd_`j'_`i'=sqrt(Var[1,`j'])
      *d *_`j'_`i'
      *sum *_`j'_`i'
      mkmat m_`j'_`i' v_`j'_`i' sd_`j'_`i' if id==336, matrix(`i'_`j')
      matrix `i'_`j'=`i'_`j''
      matrix list `i'_`j'
      }
}
xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re vce(oim)
drop m_* sd_* v_*



*For vce specifications of oim robust opg bootstrap and jackknife
      *1. mean 2. variance 3. sd
foreach i in $X1a {
est use `i'

*matrix list e(b)
matrix B=e(b)
matrix list B

*matrix list e(V)
matrix V=e(V)
matrix Var=vecdiag(V)
matrix list Var

      foreach j of numlist 1/16 {

      gen m_`j'_`i'=B[1,`j']
      *display m_`j'_`i'

      gen v_`j'_`i'=Var[1,`j']
      *display v_`j'_`i'
      *display sqrt(v_`j'_`i')
      gen sd_`j'_`i'=sqrt(Var[1,`j'])

      *d *_`j'_`i'
      *sum *_`j'_`i'
```

```
      mkmat m_`j'_`i' v_`j'_`i' sd_`j'_`i' if id==336, matrix(`i'_`j')
      matrix `i'_`j'=`i'_`j''
      matrix list `i'_`j'
      }
}


matrix dir


foreach i in $X1 {
matrix rename `i'_1 slavestate_`i'
matrix rename `i'_2 sbstate_`i'
matrix rename `i'_3 nbstate_`i'
matrix rename `i'_4 repub_`i'
matrix rename `i'_5 mixedother_`i'
matrix rename `i'_6 south_`i'
matrix rename `i'_7 midwest_`i'
matrix rename `i'_8 west_`i'
matrix rename `i'_9 percentfb_`i'
matrix rename `i'_10 unemp_`i'
matrix rename `i'_11 year2_`i'
matrix rename `i'_12 year3_`i'
matrix rename `i'_13 year4_`i'
matrix rename `i'_14 year5_`i'
matrix rename `i'_15 year6_`i'
matrix rename `i'_16 year7_`i'
}

matrix dir


foreach i in $X1 {
foreach j in m v sd {

rename `j'_1_`i' `j'_slavestate_`i'
rename `j'_2_`i' `j'_sbstate_`i'
rename `j'_3_`i' `j'_nbstate_`i'
rename `j'_4_`i' `j'_repub_`i'
rename `j'_5_`i' `j'_mixedother_`i'
rename `j'_6_`i' `j'_south_`i'
rename `j'_7_`i' `j'_midwest_`i'
rename `j'_8_`i' `j'_west_`i'
rename `j'_9_`i' `j'_percentfb_`i'
rename `j'_10_`i' `j'_unemp_`i'
rename `j'_11_`i' `j'_year2_`i'
rename `j'_12_`i' `j'_year3_`i'
rename `j'_13_`i' `j'_year4_`i'
rename `j'_14_`i' `j'_year5_`i'
rename `j'_15_`i' `j'_year6_`i'
rename `j'_16_`i' `j'_year7_`i'
}
}
d
```

```
*=====
*Combine
*A test:
matrix llcoolj=slavestate_boot, slavestate_oim_long1x
matrix list llcoolj
matrix drop llcoolj


d $X0
foreach i in $X0 {
matrix `i'=`i'_boot, `i'_oim_long1x, /*`i'_robust, `i'_state, `i'_year,
`i'_opg,*/ `i'_bootstrap_long1x_1, ///
`i'_bootstrap_long1x_2, `i'_bootstrap_long1x_3, `i'_bootstrap_long1x_4,
`i'_bootstrap_long1x_5, `i'_jackknife_long1x
*sum `i'boot
matrix list `i'_boot
*estimates
matrix list `i'_oim_long1x
*matrix list slavestate
matrix rowname `i' = mean variance sd
matrix colname `i' = samplingboot oim /*robust clstate clyear opg*/
bootstrap_1 bootstrap_2 bootstrap_3 bootstrap_4 bootstrap_5 jackknife
matrix list `i'
}
      *The variance and sd is based off of 1,000 for the samplingboot,
the others are from 336.
      *However, I want to compare the SE based off of using the
regression coef. as true theta, not just the SD of the parametric
bootstrap
foreach i in $X0 {
matrix list `i'
}

*Compare the matrices to the bootstrap results and the saved regression
results to make sure this is correct
sum *boot_m
sum *boot_v
sum *boot_sd

est use oim_long1x
estimates


estimates use bootstrap_long1x_1
estimates
estimates use jackknife_long1x
estimates
      *I checked each of these regression standard errors as compared
to the matrices I created
      *Looks exactly right.  Note that the point estimates don't change
across the regression models, only the standard errors do.


*==============
*Comparing Standard Errors
*==============
      *Instead of examining the SD and var of the bootstrap, use the
```

```
formula:

      *SE = square root ( sum of i=1 to n (each coef. for each of the
1,000 bootstraps - coef. from normal model)^2 )

sum *boot

xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re vce(oim)
*xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re vce(robust)
xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re vce(bootstrap)
      *notice that the coefficients don't change
            *= I can use the coefficients from any of these to
represent the 'true' coefficients

*Proof:
foreach i in $X0 {
sum m_`i'_*
}
      *Describes the coef. for each different se specification from
observed data
            *Notice that the coefficients are the same

matrix betas = e(b)
matrix list betas


*These are the existing matrices comparing the mean, variance, and sd
for the boot and oim
foreach i in $X0 {
matrix list `i'
}


*I want to create a new matrice that has the SE of the parametric
bootstrap and compares it to the sd of the other
      *se specifications
foreach i in $X0 {
gen `i'_coef=`i'[1,2]
}
sum *_coef
      *This creates a variable that has the coefficient value for each
of the covariates


preserve

sum *boot
keep if constantboot!=.
      *So only do the calculation for the 1,000 bootstrap samples
d, s

foreach i in /*slavestate - used for a test run */ $X0 {
sum `i'boot
      *Bootstrap value - 1,000 different observations
```

```
codebook `i'boot
sum `i'_coef
      *the regression coefficient [i.e. parameter, to use Naiman's
terminology]
      *1 unique value
codebook slavestate_coef

*Using the formula:
*SE = square root (1/n ( sum of i=1 to n ((each coef. for each of the
1,000 bootstraps - coef. from normal model)^2) ) )

*Steps:
*1. squared difference
gen `i'_se = (`i'boot-`i'_coef )^2
sum `i'_se

*2. Sum these (sum of squares):
egen x=total(`i'_se)
sum `i'_se x

*3. Divide by n
replace x=x/1000

*4. Take the square root
replace x=sqrt(x)

drop `i'_se
rename x `i'_se
sum `i'_se

mkmat `i'_se if id==1000, matrix(`i'_se)
matrix list `i'_se
}

*Combine into 1 matrix for se's of parametric bootstrap
matrix paraboot_se = slavestate_se \ sbstate_se \ nbstate_se \ repub_se
\ mixedother_se \ south_se \ midwest_se \ ///
west_se \ percentfb_se \ unemp_se \ year2_se \ year3_se \ year4_se
\ year5_se \ year6_se \ year7_se
matrix rowname paraboot_se = slavestate nbstate sbstate repub
mixedother south midwest west percentfb unemp year2 ///
year3 year4 year5 year6 year7
matrix list paraboot_se


*Create comparison matrix for se's for all the model specifications
matrix list slavestate
d *oim*
d sd_*_oim*
sum sd_*_oim*

foreach i in $X0 {
mkmat `i'_se `i'boot_sd sd_`i'_oim_long1x /*sd_`i'_robust
sd_`i'_clstate sd_`i'_clyear sd_`i'_opg*/ ///
sd_`i'_bootstrap_long1x_1 sd_`i'_bootstrap_long1x_2
sd_`i'_bootstrap_long1x_3 sd_`i'_bootstrap_long1x_4 ///
sd_`i'_bootstrap_long1x_5 sd_`i'_jackknife_long1x ///
```

```
if id==1000, matrix(`i'_comparison)
matrix list `i'_comparison
}
restore

*Combine the matrices so all in 1 comparison matrix
matrix comparison = slavestate_comparison \ sbstate_comparison
\ nbstate_comparison \ repub_comparison \ mixedother_comparison
\ south_comparison \ midwest_comparison \ ///
west_comparison \ percentfb_comparison \ unemp_comparison
\ year2_comparison \ year3_comparison \ year4_comparison
\ year5_comparison \ year6_comparison \ year7_comparison
matrix colname comparison = paraboot_mse paraboot_sd oim /*robust
clstate clyear opg*/ bootstrap jackknife
matrix rowname comparison = slavestate nbstate sbstate repub mixedother
south midwest west percentfb unemp year2 ///
year3 year4 year5 year6 year7
matrix list comparison
      *looks right

*A check:
matrix list paraboot_se

foreach i in $X0 {
matrix list comparison
matrix list `i'
}

matrix list comparison
      *Notice how close the mse is from the sd
      *It's because the means are so close, e.g.
      *matrix list slavestate


*Create variables so have them
foreach i in $X0 {
matrix list `i'_comparison
gen `i'_se=`i'_comparison[1,1]
sum `i'_se
}
matrix list comparison
sum *_se



*But what do the SE's mean?
      *To understand the magnitude of the effect, I must know the means

*drop *_var
foreach i in $X0 {
gen `i'_var = (`i'_se)^2
sum `i'_se `i'_var
display `i'_se^2
sum `i'boot `i'boot_m `i'_coef `i'_se `i'_var
}
xtpoisson plaws slavestate sbstate nbstate repub mixedother south
midwest west percentfb unemp $Year, re vce(oim)
```

```
est use oim_long1x
estimates


d $X0
foreach i in /*slavestate*/ $X0 {
sum `i'_se
matrix list `i'_se
matrix list `i'_comparison

mkmat `i'boot_m `i'_var `i'_se if id==336, matrix(`i'_se_etc_paraboot)
matrix `i'_se_etc_paraboot=`i'_se_etc_paraboot'
matrix list `i'_se_etc_paraboot

matrix list `i'_boot
      *Notice that the means are the same as I am using the parametric
bootstrap means but the variance and sd or se
      *are different because one is the SD, the other is the SE
calculation using `i'_coef as the true theta value

matrix list `i'
      *This is the old matrix that uses the sd instead of se
calculation
      *It is made up of:
matrix list `i'_boot
matrix list `i'_oim_long1x

matrix list `i'_bootstrap_long1x_1
matrix list `i'_bootstrap_long1x_2
matrix list `i'_bootstrap_long1x_3
matrix list `i'_bootstrap_long1x_4
matrix list `i'_bootstrap_long1x_5
matrix list `i'_jackknife_long1x


*I want to create a new matrix
matrix `i'_final = `i'_se_etc_paraboot, `i'_oim_long1x,  /*`i'_robust,
`i'_state, `i'_year, `i'_opg,*/ ///
`i'_bootstrap_long1x_1, `i'_bootstrap_long1x_2, `i'_bootstrap_long1x_3,
`i'_bootstrap_long1x_4, ///
`i'_bootstrap_long1x_5, `i'_jackknife_long1x
matrix list `i'_final
matrix list `i'

matrix `i'_final=`i'_final'
matrix colname `i'_final = mean variance se
matrix rowname `i'_final = paraboot_se oim /*robust clstate clyear
opg*/ bootstrap_1 bootstrap_2 bootstrap_3 bootstrap_4 bootstrap_5
jackknife
matrix list `i'_final
}

foreach i in $X0 {
matrix list `i'_final
}
      *This is KEY!
sum *boot
```

```
*===
*And what if I want a matrix that lists only the se's?
      *Look at the matrix comparison
matrix list comparison
      *This is KEY



*===
*What about total SE for each method?
matrix list comparison

*preserve
*keep if id==1

global X1 oim_long1x /*robust clstate clyear opg*/ bootstrap_long1x_1
bootstrap_long1x_2 ///
bootstrap_long1x_3 bootstrap_long1x_4 bootstrap_long1x_5
jackknife_long1x

*drop *_sum
foreach i in $X1 {
sum  sd_*_`i'

matrix list comparison

gen `i'_sum= sd_slavestate_`i' + ///
sd_sbstate_`i' + ///
sd_nbstate_`i' + ///
sd_repub_`i' + ///
sd_mixedother_`i' + ///
sd_south_`i' + ///
sd_midwest_`i' + ///
sd_west_`i' + ///
sd_percentfb_`i' + ///
sd_unemp_`i' + ///
sd_year2_`i' + ///
sd_year3_`i' + ///
sd_year4_`i' + ///
sd_year5_`i' + ///
sd_year6_`i' + ///
sd_year7_`i'
sum `i'_sum

display sd_slavestate_`i' + ///
sd_sbstate_`i' + ///
sd_nbstate_`i' + ///
sd_repub_`i' + ///
sd_mixedother_`i' + ///
sd_south_`i' + ///
sd_midwest_`i' + ///
sd_west_`i' + ///
sd_percentfb_`i' + ///
sd_unemp_`i' + ///
```

```
sd_year2_`i' + ///
sd_year3_`i' + ///
sd_year4_`i' + ///
sd_year5_`i' + ///
sd_year6_`i' + ///
sd_year7_`i'
}


*What about for the parametric bootstrap SE?
foreach i in se {
sum  *_`i'
matrix list comparison
gen `i'_sum= slavestate_`i' + ///
sbstate_`i' + ///
nbstate_`i' + ///
repub_`i' + ///
mixedother_`i' + ///
south_`i' + ///
midwest_`i' + ///
west_`i' + ///
percentfb_`i' + ///
unemp_`i' + ///
year2_`i' + ///
year3_`i' + ///
year4_`i' + ///
year5_`i' + ///
year6_`i' + ///
year7_`i'
sum `i'_sum

display slavestate_`i' + ///
sbstate_`i' + ///
nbstate_`i' + ///
repub_`i' + ///
mixedother_`i' + ///
south_`i' + ///
midwest_`i' + ///
west_`i' + ///
percentfb_`i' + ///
unemp_`i' + ///
year2_`i' + ///
year3_`i' + ///
year4_`i' + ///
year5_`i' + ///
year6_`i' + ///
year7_`i'
      *looks right
}

sum *_sum
      *This is KEY


*===
*What about a standardized comparison?
*I.e. a relative error comparison of the stata standard error technique
```

```
vs the true standard error?
      *e.g. ( sigma(jack - sigma (true) ) / sigma(true)
            *then take the absolute value
            *then average over i [the 16 coefficients]

d, s
sum
d

d sd*
d sd*oim*
d sd*bootstrap*
d sd*jackknife*


foreach i in $X0 {
matrix list `i'_final
}
matrix list comparison

matrix dir
d $X0
foreach i in /*slavestate*/ $X0 {
sum `i'_se
matrix list `i'_jackknife_long1x
matrix list `i'_se
matrix list `i'_comparison
matrix list `i'_se_etc_paraboot
matrix list `i'_final
}

sum percentfb_se sd_percentfb_oim sd_percentfb*
codebook id percentfb_se sd_percentfb_oim_long1x
sd_percentfb_bootstrap_long1x_2 sd_percentfb_jackknife_long1x
matrix list percentfb_final

*creating a matrix for relative error
foreach i in $X0 {
gen x=abs( ((sd_`i'_oim_long1x - `i'_se) / `i'_se) )
gen y1=abs( ((sd_`i'_bootstrap_long1x_1 - `i'_se) / `i'_se) )
gen y2=abs( ((sd_`i'_bootstrap_long1x_2 - `i'_se) / `i'_se) )
gen y3=abs( ((sd_`i'_bootstrap_long1x_3 - `i'_se) / `i'_se) )
gen y4=abs( ((sd_`i'_bootstrap_long1x_4 - `i'_se) / `i'_se) )
gen y5=abs( ((sd_`i'_bootstrap_long1x_5 - `i'_se) / `i'_se) )
gen z=abs( ((sd_`i'_jackknife_long1x - `i'_se) / `i'_se) )
sum x y1 y2 y3 y4 y5 z
mkmat x y1 y2 y3 y4 y5 z if id==1000, matrix(`i'_relativeerror)
matrix `i'_relativeerror=`i'_relativeerror'
matrix list `i'_relativeerror
drop x y1 y2 y3 y4 y5 z
}

d $X0
matrix relativeerror = slavestate_relativeerror, sbstate_relativeerror,
nbstate_relativeerror, repub_relativeerror, mixedother_relativeerror,
///
south_relativeerror, midwest_relativeerror, west_relativeerror,
```

```
percentfb_relativeerror, unemp_relativeerror, year2_relativeerror, ///
year3_relativeerror, year4_relativeerror, year5_relativeerror,
year6_relativeerror, year7_relativeerror
matrix relativeerror=relativeerror'
matrix colname relativeerror = oim boot_1 boot_2 boot_3 boot_4 boot_5
jack
matrix rowname relativeerror = slave sb nb repub mixed south midwest
west percentfb unemp yr2 yr3 yr4 yr5 yr6 yr7
matrix list relativeerror

*Sum and then average the values
foreach i in $X0 {
gen x_`i'=abs( ((sd_`i'_oim_long1x - `i'_se) / `i'_se) )
gen y1_`i'=abs( ((sd_`i'_bootstrap_long1x_1 - `i'_se) / `i'_se) )
gen y2_`i'=abs( ((sd_`i'_bootstrap_long1x_2 - `i'_se) / `i'_se) )
gen y3_`i'=abs( ((sd_`i'_bootstrap_long1x_3 - `i'_se) / `i'_se) )
gen y4_`i'=abs( ((sd_`i'_bootstrap_long1x_4 - `i'_se) / `i'_se) )
gen y5_`i'=abs( ((sd_`i'_bootstrap_long1x_5 - `i'_se) / `i'_se) )
gen z_`i'=abs( ((sd_`i'_jackknife_long1x - `i'_se) / `i'_se) )
}

foreach i in x y1 y2 y3 y4 y5 z {
gen `i'=`i'_slavestate + `i'_sbstate + `i'_nbstate + `i'_repub +
`i'_mixedother + `i'_south + `i'_midwest + `i'_west ///
 + `i'_percentfb + `i'_unemp + `i'_year2 + `i'_year3 + `i'_year4 +
`i'_year5 + `i'_year6 + `i'_year7
replace `i'=`i'/16
tab `i'
 }
rename x oim_relativeerror
rename y1 bootstrap_1_relativeerror
rename y2 bootstrap_2_relativeerror
rename y3 bootstrap_3_relativeerror
rename y4 bootstrap_4_relativeerror
rename y5 bootstrap_5_relativeerror
rename z jackknife_relativeerror
sum *relativeerror
display 2.66/16
      *looks right
codebook *relativeerror

matrix list relativeerror
mkmat oim_relativeerror bootstrap_1_relativeerror
bootstrap_2_relativeerror bootstrap_3_relativeerror ///
bootstrap_4_relativeerror bootstrap_5_relativeerror
jackknife_relativeerror if id==1000, matrix(relativeerror_final)
matrix colname relativeerror_final = oim boot_1 boot_2 boot_3 boot_4
boot_5 jack
matrix rowname relativeerror_final = relativeerror
matrix list relativeerror_final
      *KEY

*OIM looks like it is best for xt_poisson_positive



*=====================================================================
```

```
*Data Analysis Techniques
*=====================================================================

      *1. SE comparisons
      matrix list comparison
            *I also have comparison of mean, sd, var - not as important
      foreach i in $X0 {
      matrix list `i'
      }
      *2. boxplot of variance of 1,000 samples w/ mean of predicted
                  *not very useful
      *3. hist of variance of 1,000 samples w/ mean of predicted
                  *not very useful
      *4. dotplot
            *SE's - best
            *variances
                  *also useful but stick with SE's
            *means - don't need


*=====================================================================
*Ranking of SE techniques(again)

matrix list comparison
sum *_sum



*=====================================================================
d
sum
d, s

save results_xt_poisson_positivelaws.dta, replace

log close
translate results_xt_poisson_positivelaws.log "~/Documents/Johns
Hopkins/AMS/AMS Master's
Paper/Stata/Results/xt_poisson_positive/results_xt_poisson_positivelaws
.smcl", linesize(79) translator(smcl2log) replace
```

Works Cited

American Community Survey, U.S. Census Bureau. 2005-2011. "American Community Survey Data on the Foreign-Born Population." Retrieved March 17, 2012 (http://www.census.gov/population/foreign/data/acs.html).

American Community Survey, U.S. Census Bureau. 2005-2011. "Summary File." Retrieved March 17, 2012 (http://ftp2.census.gov).

Angrist, J.D. and Pischke, J.S. 2009. *Mostly Harmless Econometrics: An Empiricist's Companion*. NJ: Princeton University Press.

Bureau of Labor Statistics, U.S. Department of Labor. 2005-2011. "Employment status of the noninstitutional population in states by sex, race, Hispanic or Latino ethnicity, and detailed age: 2005 to 2011 Annual Averages." Retrieved December 10, 2012 (http://www.bls.gov/lau/#ex14?).

Bureau of Labor Statistics, U.S. Department of Labor. "Geographic Profile of Employment and Unemployment." Retrieved September 14, 2013 (http://www.bls.gov/opub/gp/laugp.htm).

The Council of State Governments. 2005 to 2011. *The Book of the States, Vols. 37-43.* Lexington, KY: The Council of State Governments.

Efron, B. and Hinkley, D.V. 1978. "Assessing the accuracy of the maximum likelihood estimator: Observed versus expected Fisher Information." *Biometrika* 65(3): 457-483.

Eicker, F. 1967. "Limit Theorems for Regressions with Unequal and Dependent Errors." In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability* Vol. I(1): 59-82.

Gardner, W., Mulvey, E., and Shaw, E. 1995. "Regression Analyses of Counts and Rates: Poisson, Overdispersed Poisson, and Negative Binomial Models." *Psychological Bulletin* 118(3): 392-404.

Osgood, D.W. 2000. "Poisson-Based Regression Analysis of Aggregate Crime Rates." *Journal of Quantitative Criminology* 16(1): 21-43.

StataCorp LP. 2011. "nbreg." In *Stata Base Reference Manual, Release 12*. TX: StataCorp LP.

StataCorp LP. 2011. "poisson." In *Stata Base Reference Manual, Release 12*. TX: StataCorp LP.

StataCorp LP. 2011. "xtnbreg." In *Stata Longitudinal-Data/Panel-Data Reference Manual, Release 12*. TX: StataCorp LP.

StataCorp LP. 2011. "xtpoisson." In *Stata Longitudinal-Data/Panel-Data Reference Manual, Release 12*. TX: StataCorp LP.

"State Laws Related to Immigration and Immigrants." National Conference of State Legislatures. 2005-2011. Retrieved February 14, 2012 (http://www.ncsl.org/issues-research/immig/state-lawsrelated- to-immigration-and-immigrants.aspx).

State of Alabama. 2011. "House Bill 56." Alabama: State Legislature. Retrieved August 15, 2013 (http://alisondb.legislature.state.al.us/acas/searchableinstruments/2011rs/bills/hb6.htm).

White, H. 1980. "A Heteroskedasticity-Consistent Covariance Matrix Estimator and a Direct Test of Heteroskedasticity." *Econometric Society* 48: 817-838.

Curriculum Vita

Robert Nathenson was born in Pittsburgh, Pennsylvania on January 5, 1983. He completed his undergraduate training in History and American Culture Studies at Washington University in St. Louis before completing a Master's degree in Sociology at the University of Oxford. He joined the Johns Hopkins Sociology Department as an Institute of Education Sciences Predoctoral Trainee in 2008. In 2010 he enrolled in the joint degree program in Applied Math & Statistics. He graduated in 2014 from Johns Hopkins University with a Master's in Engineering in Applied Math & Statistics and a Doctor of Philosophy in Sociology. He is currently a Postdoctoral Researcher and a Junior Fellow at the Leonard Davis Institute of Health Economics at the University of Pennsylvania. His research interests include education, child development, health, migration, stratification, and the application of quantitative methods.