Erik Swanson
Cori Saviano
Li Zha
Final Project

1 Introduction

In analyzing time series data, we are posed with the question of how past events influences the current situation. In order to determine this, we usually impose a model on our data that involves a number of lag terms. The coefficients on these lag terms give us information about the effects of past events, and should help us predict future values for whatever dependent variable we are studying. However, in adding additional lag terms, we are gaining information, but at the same time sacrificing simplicity.

When the true model is not known (which it rarely is) we run the risk of over-specifying or under-specifying the model by adding too many or too few lags. The Bayesian information criterion and the Akaike information criterion can help in regularization of our model. These criterion attempt to balance the information gained by adding additional lags against the increase in the complexity of the model that these lags cause.

In order to determine how effective these methods of regularization are, we must know the true specification of the model. Since this would be a difficult, if not impossible, task with real economic data, the problem lends itself well to Monte Carlo simulation. In this simulation we will be randomly generating data to create a wide variety of time-series specifications, and then analyzing the efficacy of the AIC and SBIC in predicting the true model under these varying circumstances.

2 AIC versus SBIC

When we make choices about the order p of an autoregression, we have to balance the marginal benefit of including more lags against the marginal cost of increased uncertainty of estimation. If we do not include enough lags, we run the risk of omitting potentially significant information contained in more distant lags. On the other hand, with too many lags we estimate more coefficients than needed. This practice might in turn lead to estimation errors.

One approach to choosing the appropriate p is to perform hypothesis tests on the final lag. For example, we start from estimating an AR(4) model and test whether the coefficient on the fourth lag is significant at the appropriate significance levels, say 5%. If it is not significant, we drop it and estimate AR(3) model, and so on. One drawback of this t-statistics approach is that such tests would incorrectly reject the null hypothesis of zero coefficient 5% of the time. To put in another way, when the true number of lags is four, the t-statisitic approach would estimate p to be six 5% of the time.

An alternative approach to get around this problem is to estimate p by using "information criterion," such as **Akaike information criterion (AIC)** and **Bayes information criterion (BIC)**, also known as **Schwartz information criterion (SIC)**. These information criteria are designed explicitly for **model selection**. Model selection criteria generally involve information criteria function calculations for each of the models. We pick the model for which the function is maximized or minimized. Stata calculates the AIC and BIC results based on their standard definitions, which include the constant term from the log likelihood function. Stata uses the following form of log likelihood function (LL) for VAR(p), as shown by Hamilton (1994, 295-296).

$$LL = (\frac{T}{2})\{\ln(|\hat{\Sigma}^{-1}|) - K\ln(2\pi) - K\}$$

where T is the number of observations, K is the number of equations, and $\hat{\Sigma}$ is the maximum likelihood estimate of $E[u_t u_t']$, where u_t is the K $\times$ 1 vector of disturbances. Because

$$\ln(|\hat{\Sigma}^{-1}|) = -\ln(|\hat{\Sigma}|)$$

we can rewrite the log likelihood as

$$LL = -(\frac{T}{2})\{\ln(|\hat{\Sigma}|) + K\ln(2\pi) + K\}$$

.

Stata uses the following definitions of AIC and BIC function based on the log likelihood function defined above.

$$AIC(p) = -2(\frac{LL}{T}) + 2\frac{t_p}{T}$$

$$BIC(p) = -2(\frac{LL}{T}) + \frac{\ln(T)}{T}t_p$$

where LL stands for the log likelihood for a VAR(p) (shown above), T is the number of observations, and p is the number of lags.

Note that STATA's suggestion for the ideal lag length is the minimization of the AIC and BIC functions.

In most cases, we prefer the model that has fewest parameter to estimate, provided that each one of the candidate models is correctly specified. This is called the most **parsimonious** model of the set. The AIC does not always suggests the most parsimonious model, because the AIC function is largely based on the log likelihood function. Davidson and McKinnon (2004, 676) indicates that whenever two or more models are nested, the AIC may fail to choose the most parsimonious one, if that these models are correctly specified. In another case, if all the models are nonnested, and only one is well specified, the AIC chooses the well-specified model asymptotically, because this model has the largest value of the log likelihood function.

The Bayesian information criterion (BIC) avoids the problem discussed above by replacing 2 in the AIC function with the $\ln(T)$ term. As $T \to \infty$, the addition of another lag would increase the BIC value by a larger margin. Hence, asymptotically, BIC would pick the more parsimonious model than AIC might suggest.

Stock and Watson (2007) recommends that we choose the model suggested by AIC rather than BIC. They argue that including more parameters is better than omitting significant parameters.

In this project, we seek to examine how well the AIC and BIC predict the lag lengths correctly in two cases. First, the lag coefficients are abruptly truncated. Second, the lag coefficients decline linearly. We used simulation experiments, often referred to as **Monte Carlo experiments** extensively in our tests. Monte Carlo experiments is the topic in the next section.

3 Monte Carlo Studies

Monte Carlo studies is the simulation of the behavior of the estimators under controlled conditions that may deviate from the standard assumptions. Monte Carlo experiments allow us to vary some parameters that in actual cases are fixed and test how changes in these parameters affect the test statistics, for example, the AIC or BIC results.

The first step in Monte Carlo studies is to generate data for simulations, We can use actual variables from real data sets or generate data from **random-number generator (RNG)**. Note that because most RNGs require a specification of the first term of the sequence, the **seed**, the data generating process is not purely random. RNGs generate later terms in the sequence based on calculations. Many RNGs generate numbers that appear to be drawings from the uniform $U(0, 1)$ distribution. Another popular choice is to draw from the normal distribution with mean and standard deviation given, such as $N(0, 1)$.

When we implement the Monte Carlo experiment, we specify the number of repeated samples and the test statistics of interest (AIC and BIC) to be calculated in each simulation. We must also take great care to accumulate the estimates of the test statistics in a new data set.

Once we obtained the simulation results, we need to examine the properties of the estimates. For example, we calculate the mean and standard deviation, or use quantiles to assess critical values (for example, the predetermined lag length in our study.)

4 Data Generation Process

As stated above, the first step of a Monte Carlo simulation is the data generation process. In generating our data, we must consider what selection of the possible models we want to analyze. We choose to analyze the following changes.

- Abrubtly declining versus gradually declining lag coefficients

- The number of lags
- Autocorrelation of the X values
- Number of observations

Here we will describe our method for generating the data. The data is generated through the use of a Stata .do file, which can be found in Appendix A. The end result of our data generating process is a number of models which share the form:

$$Y_t = \beta_{00} + \beta_0 X_t + \beta_1 X_{t-1} + \beta_2 X_{t-2} + \ldots + \beta_p X_{t-p} + u_t$$

This is an time series regression, but it is not auto regressive. The X values in this model will be generated independently of the Y values. As can be seen in the equation above, our Y values are dependent on the distribution of the error term, the distribution of the X values, and the number and magnitude of the lag coefficients.

The first step in the data generating process is to generate a number of X_t values that we will later apply our lag coefficients to. Since one of the changes we want to analyze is the autocorrelation of X values, they are a first order autoregressive with the following form:

$$X_t = \alpha Z_{t-1} + Z_t$$

Where Z_t is a normally distributed random variable with mean 5 and standard deviation 5. In our models we specify 10 different values for α, ranging from 0 to 0.9. We will eventually pair each of these sets of X values with each of our specifications for the lag coefficients. First however, we must create our specification for the variation in number and magnitude of lag coefficients.

Our models are created with numbers of lags including each of 1 to 10 lag coefficients. for both abruptly truncated and gradually declining coefficients. The method behind our generation of these coefficients is that we want the effects of X to be as close to stationarity as possible. This will ensure that we are creating the most significant coefficients that we can, with the effect that we are not giving unreasonably small coefficients for AIC and SBIC to pick up.

To that end, if we are to have constant coefficients on the number of lags, the coefficient must be less than one over the number of lags for the process to be stationary. In our process, we use the following formula to calculate the coefficients for our constant lag value, where p is the number of lags and β is the coefficient on all of these lags:

$$\beta = \frac{1}{p} - .001$$

If we are to have decreasing lags, an interesting specification for the lag coefficients would be one in which the coefficients are linearly decreasing. The following equation gives us linearly decreasing coefficients for each n of p lags, and is stationary in its first difference. That is, it has a single unit root.

$$\beta_n = \frac{p-(n-1)}{\frac{1}{2}p^2 + \frac{1}{2}p}$$

However, we can achieve stationarity without differences if we subtract a small amount from each coefficient. So, in our data generation process we use the following equation to generate our lag coefficients:

$$\beta_n = \frac{p-(n-1)}{\frac{1}{2}p^2 + \frac{1}{2}p} - .001$$

Then, with these lag coefficients, and our X values, we construct our models for Y_t. For simplicity, we assume that the values of β_{00} and β_0, as seen in the equation for Y_t above, are equal to zero. This leaves us with

$$Y_t = \beta_1 X_{t-1} + \beta_2 X_{t-2} + \ldots + \beta_p X_{t-p} + u_t$$

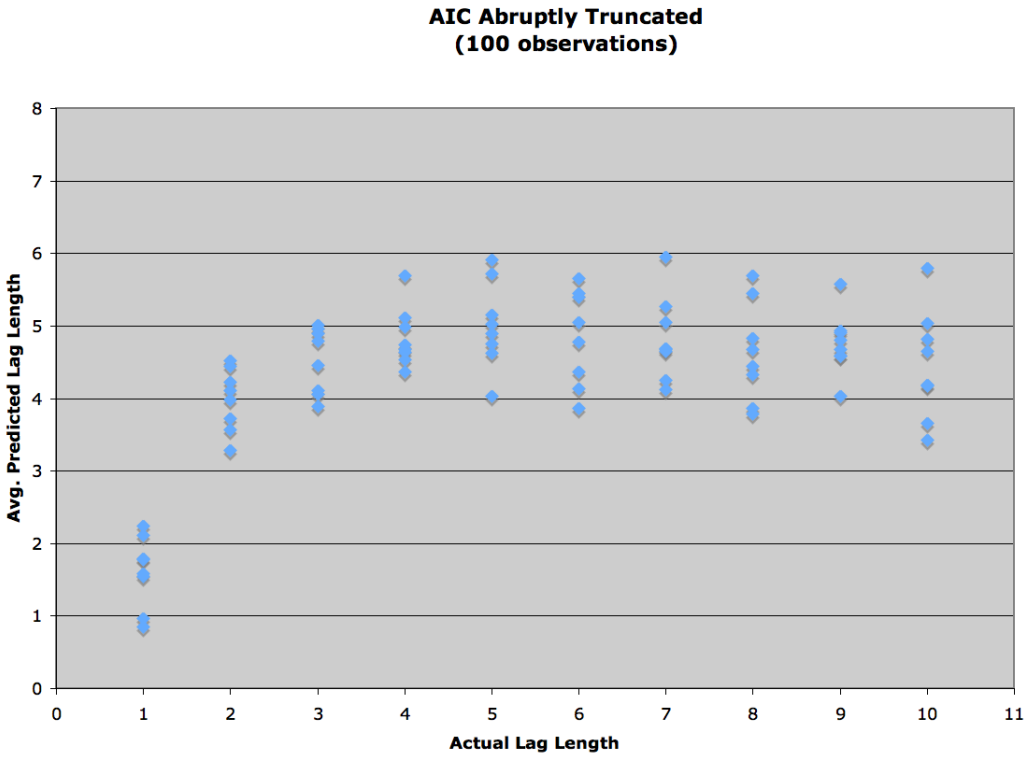
.

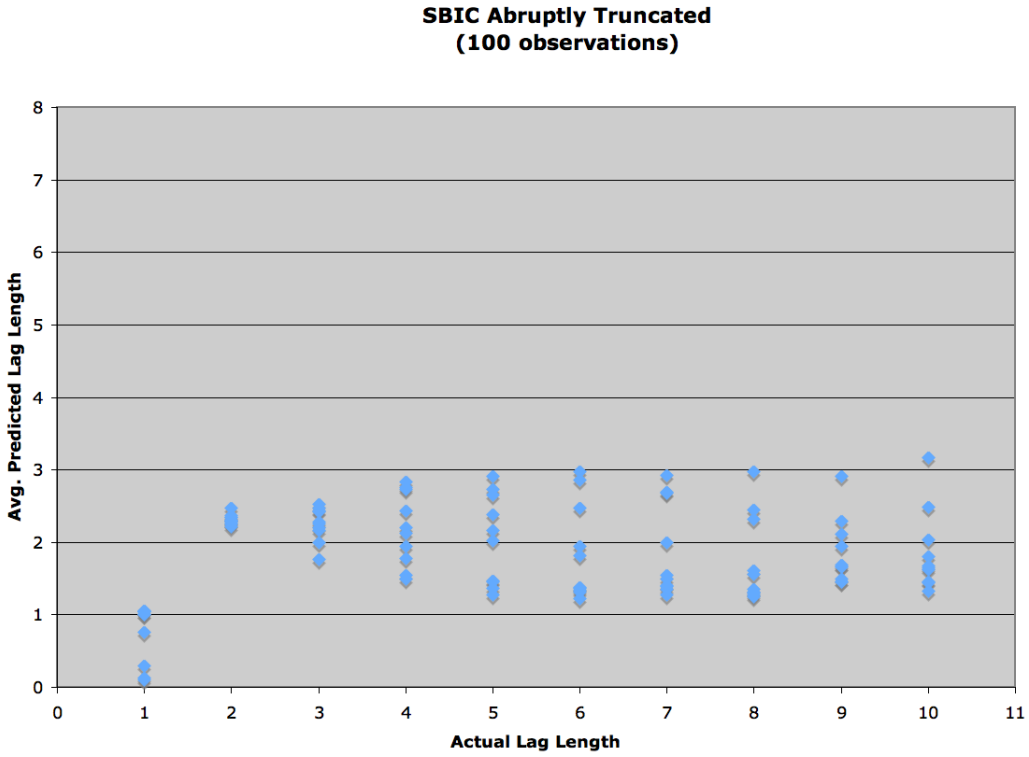
Where u_t is a random normal distribution with mean zero and standard deviation one. Since we have ten X_t series, ten lag specifications with abruptly truncated coefficients, and ten lag specifications with gradually decreasing coefficients, a single run of our simulation generates 200 Y_t series.

The next step in a Monte Carlo simulation is the definition of the test to be performed, and the number of times it is performed. For each of our 200 Y_t series we have Stata perform an AIC and SBIC test, and store the number of predicted lags associated with each parameterization of Y_t. This gives us 400 final data points for a single run of the simulation. We then use Stata's simulate command to run this simulation 100 times for both Y_t with $t = 100$, and $t = 1000$. The following section will present our results and analysis of the changes defined at the start of this section.

5 Data and Analysis

We begin our analysis with the simulation with 100 observations for each Y_t. This is a relatively small sample size, but not unreasonably small for some time series analyses. In this smaller sample size, we might predict that, since AIC tends to be less parsimonious than SBIC, that AIC would predict a larger number of lags on average.

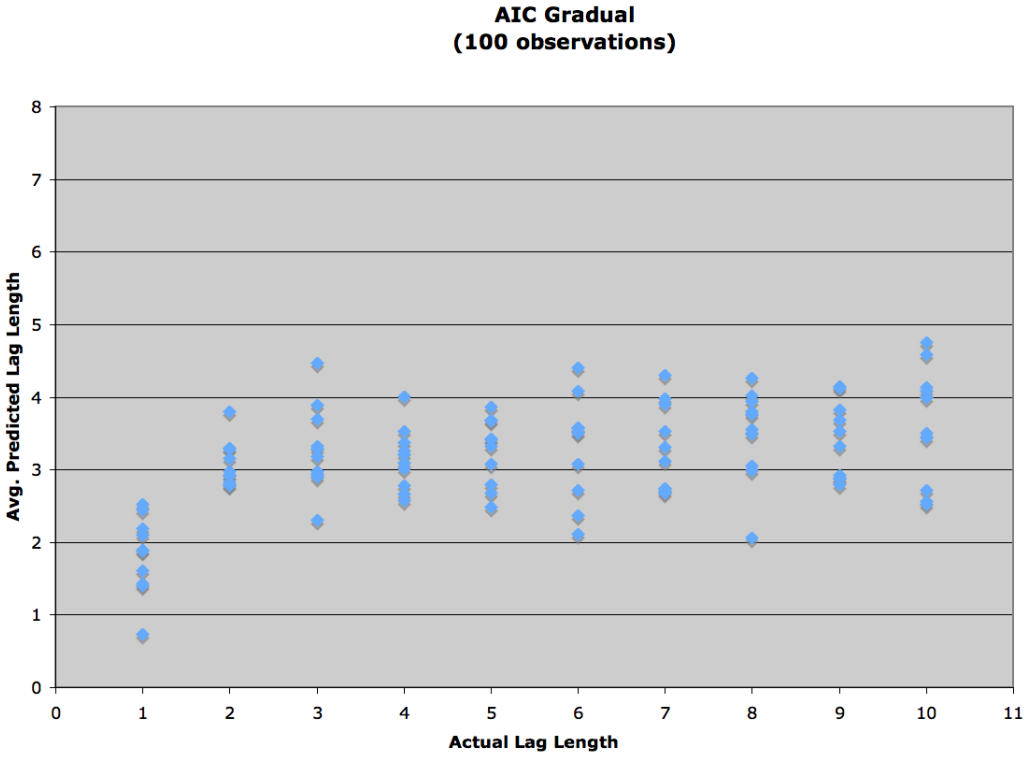


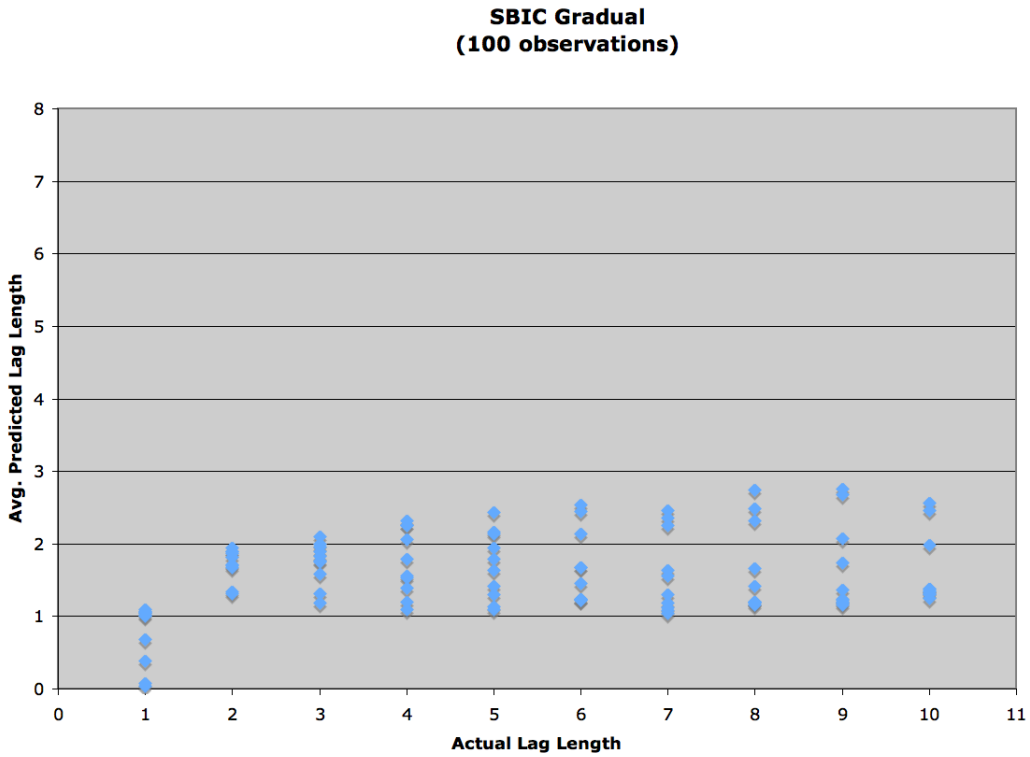


The graphs above plot the average predicted number of lags for the 100 runs of the

simulation (across all X's) against the actual lags in the abruptly truncated model. Since these graphs are on the same scale, it is easy to see that the AIC predicts larger lag lengths than the SBIC. Still, both under-specify related to the true model after 6 and 2 lags for AIC and SBIC respectively.

This under-specification is even more pronounced when we have gradually decreasing lag coefficients. As can be seen in the graphs below, the AIC under-specifies the model after 4 true lags.





These graphs can give us an intuitive answer to the effect of gradually decreasing lags on the predictions of the AIC and SBIC. It appears that gradually decreasing lags make the information gained from an additional lag term not worth the added complexity. To represent this numerically, however, we can run an OLS regression of the predicted lags on the true number of lags. If AIC and SBIC could perfectly predict the true number of lags, we would expect coefficients of one on these the true number of lags.

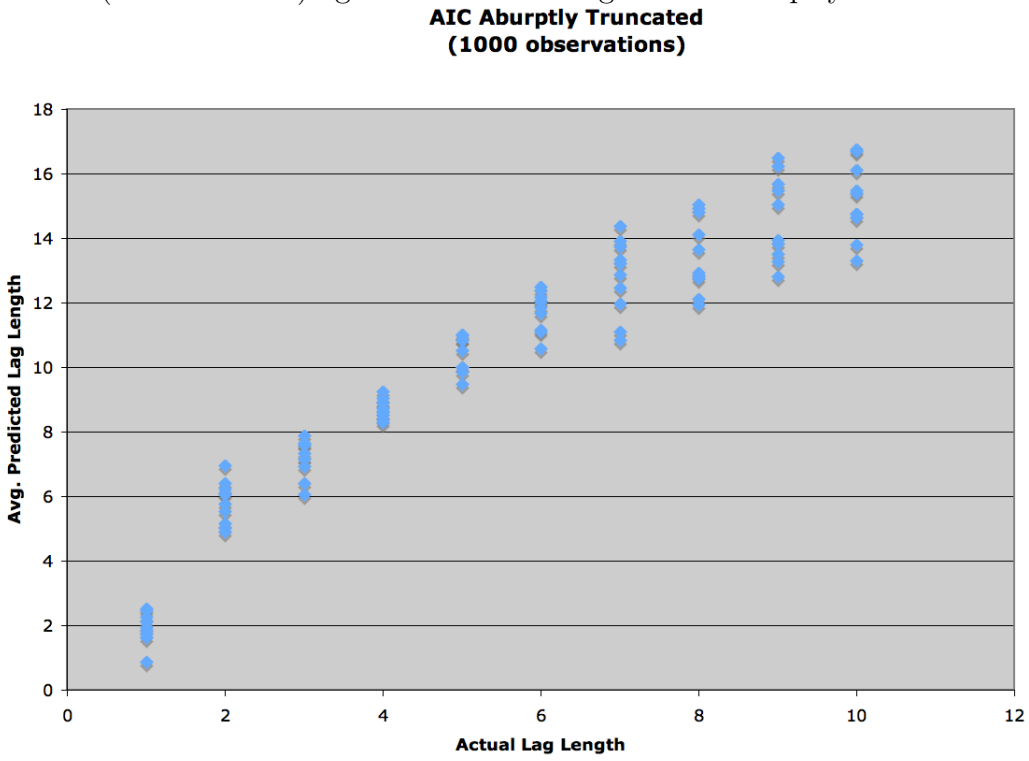
	(1)	(2)	(3)	(4)
VARIABLES	AICTrunc100	AICGrad100	SBICTrunc100	SBICGrad100
NumLags	0.236***	0.122***	0.0200	0.0408**
	(0.0416)	(0.0255)	(0.0251)	(0.0203)
Constant	3.193***	2.502***	1.766***	1.366***
	(0.267)	(0.159)	(0.169)	(0.124)
Observations	100	100	100	100
R-squared	0.324	0.230	0.008	0.047

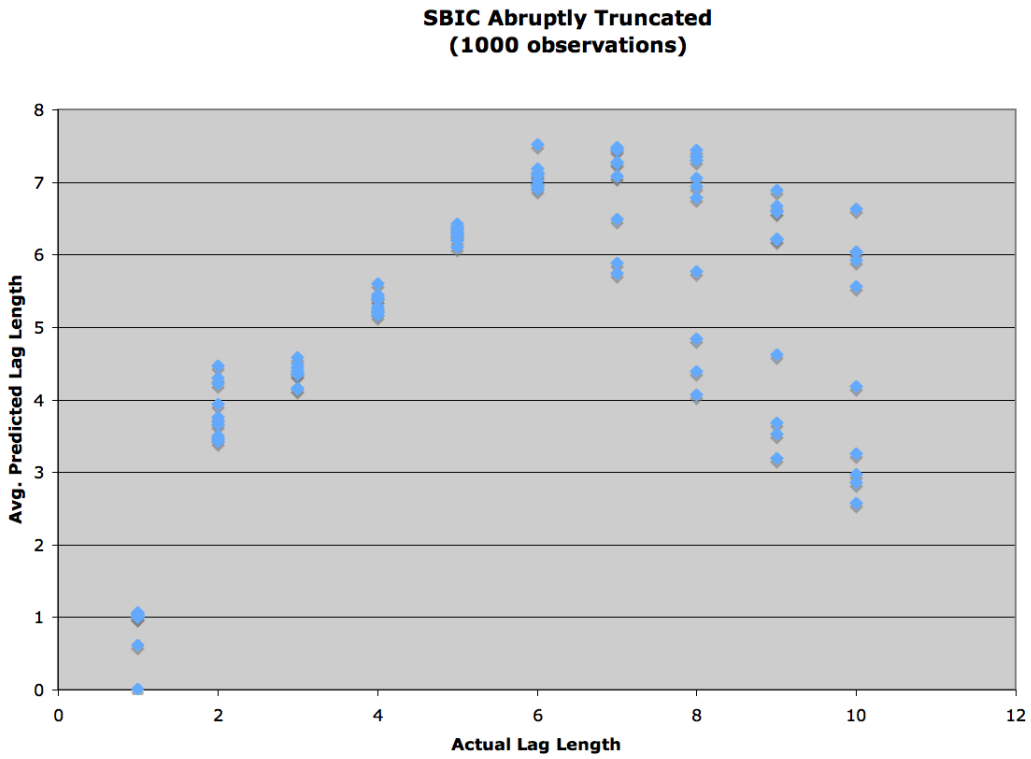
Robust standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1

While we do not have coefficients of one, we do have positive and statistically significant coefficients for both of the AIC predictions, and for the SBIC prediction for the gradually declining lag coefficients. It is worth noting that, at least for the AIC, the coefficient on the

number of lags for the truncated lag coefficients is larger than for gradually declining lags. It is also worth noting that all of these coefficients would likely be closer to one if less lags were included in the independent variable.

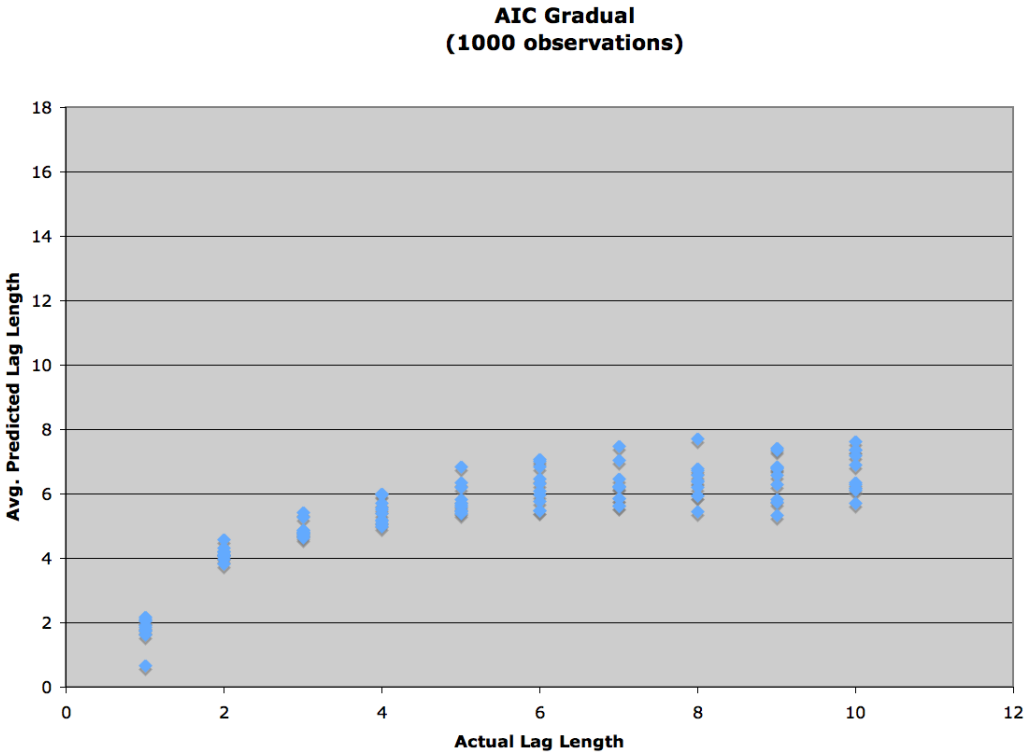
We can now move on to similar analysis for 1000 observations in each Y_t. We would expect with more observations to see larger number of lags predicted by both AIC and SBIC. The following graphs are, again, the average predicted number of lags for the 100 runs of the simulation (across all X's) against the actual lags in the abruptly truncated model.

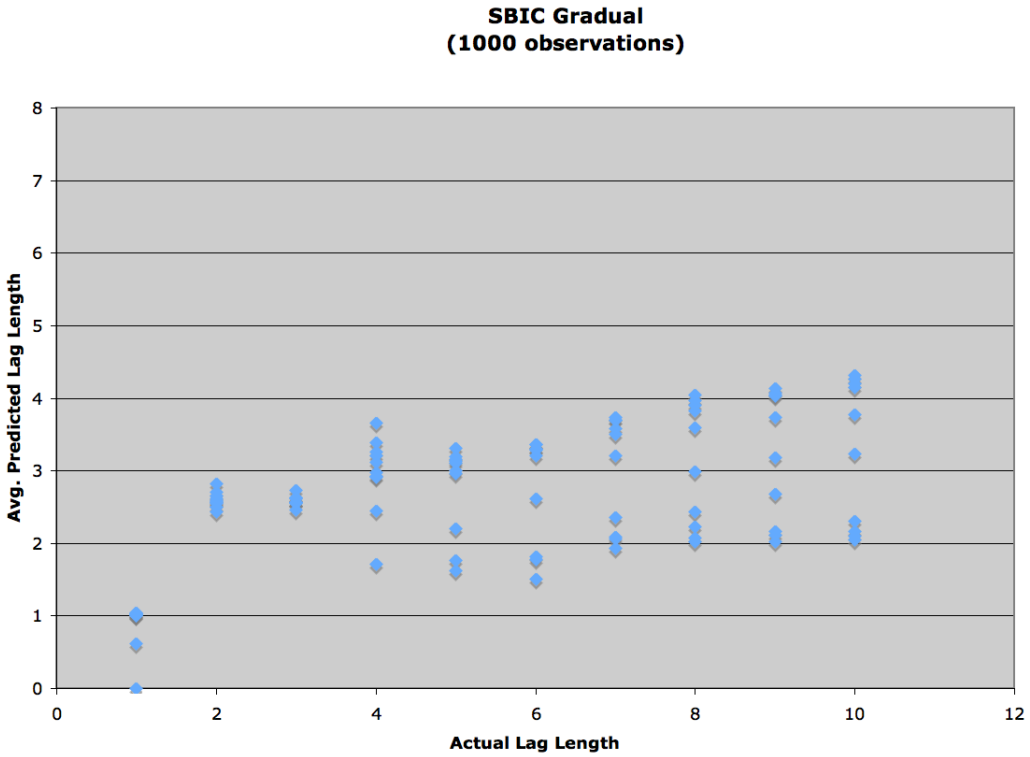




As can be seen in the first graph, the AIC over-specifies the model for all values of the true lags, but has a fairly linear trend. The SBIC model, however, exhibits some strange behavior, as it's predicted number of lags seems to curve downward after five true lags. This is almost certainly a product of our specification for generating the lag coefficients as the number of lags increases. In our method for generating these coefficients, as more lags were added, the magnitude of all of the coefficients decreased fairly significantly. Before this, however, we see a fairly consistent, if slightly over-specified trend.

Next we can look at the predictions when we have 1000 observations and gradually declining lag coefficients.





As in the case where we had 100 observations, both AIC and SBIC predict fewer lags when the lag coefficients are gradually decreasing as opposed to abruptly truncated. Additionally, even with 1000 observations, the addition of more lags in a gradually decreasing specification

has a decreasing effect as we get into higher number of lags.

Again, we can represent these results numerically by running an OLS regression, the results of which are given below.

	(1)	(2)	(3)	(4)
VARIABLES	AICTrun1000	AICGrad1000	SBICTrun1000	SBICGrad1000
NumLags	1.371***	0.431***	0.358***	0.172***
	(0.0534)	(0.0374)	(0.0702)	(0.0308)
Constant	2.647***	3.060***	3.127***	1.764***
	(0.323)	(0.248)	(0.393)	(0.179)
Observations	100	100	100	100
R-squared	0.909	0.672	0.289	0.294

Robust standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1

When we have 1000 observations, all of our coefficients on the true number of lags become significant at the 1% level. Now, for both the AIC and SBIC we can see a larger coefficient on the true number of lags when the lag coefficients are abruptly truncated as opposed to gradually declining.

Next we can turn our attention to the effect of autocorrelation in the X series on the predictions of the AIC and SBIC. In order to determine this effect, we run the same OLS regression as previously with the addition of a term for the amount of autocorrelation in the X series, the results of which are below.

	(1)	(2)	(3)	(4)
VARIABLES	AICTrunk100	AICGrad100	SBICTrunk100	SBICGrad100
NumLags	0.236***	0.122***	0.0200	0.0408**
	(0.0399)	(0.0233)	(0.0230)	(0.0155)
Xvalue	0.116***	0.116***	0.134***	0.140***
	(0.0344)	(0.0200)	(0.0192)	(0.0124)
Constant	2.557***	1.861***	1.026***	0.596***
	(0.342)	(0.201)	(0.208)	(0.129)
Observations	100	100	100	100
R-squared	0.402	0.440	0.370	0.598

Robust standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1

Since the number of lags and autocorrelation of the X's in this model are unrelated, we would expect the same coefficients for number of lags as we saw before, which is in fact the case. What is interesting, however, is the positive and statistically significant effect of

autocorrelation in the X's on the predicted number of lags. The following table is the same regression, but with the data from the set of with 1000 observations per Y_t.

	(1)	(2)	(3)	(4)
VARIABLES	AICTrunk1000	AICGrad1000	SBICTrunk1000	SBICGrad1000
NumLags	1.371***	0.431***	0.358***	0.172***
	(0.0509)	(0.0363)	(0.0675)	(0.0257)
Xvalue	0.152***	0.112***	0.146**	0.175***
	(0.0445)	(0.0313)	(0.0574)	(0.0217)
Constant	1.812***	2.442***	2.322***	0.801***
	(0.409)	(0.301)	(0.460)	(0.207)
Observations	100	100	100	100
R-squared	0.921	0.717	0.337	0.598

Robust standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1

Again, we see the same coefficients on the true number of lags in the model, as well as positive and statistically significant coefficients for the autocorrelation of the X series. Interestingly the coefficients on this autocorrelation is not incredibly different from those in the model with only 100 observations. This suggests that for every 10% increase in autocorrelation of the X series, we can expect an increase of around .14 predicted lags. So, a small amount of autocorrelation does not have a large effect on the predictions of the AIC and SBIC, a large amount of autocorrelation could have a noticeable effect.

6 Conclusions

In this Monte Carlo simulation we have studied a number of different models and the ability of the AIC and SBIC to predict the true number of lags in these models. The variation that had the most significant effect was the number of observations in our Y_t series. More observations led to significantly more accurate predictions of the true number of lags. However, this occurred mostly with large numbers of lags, that would be undesirable in a model with only 100 observations.

A more interesting effect is the difference between abruptly truncated lag coefficients and gradually declining lag coefficients. It was clear that the AIC and SBIC were less accurate in predicting gradually declining lag coefficients, both in terms of magnitude and statistical significance. While this could be due to the model used to generate these lag coefficients, it makes sense that since, as these lags include less and less information, the AIC and SBIC would tend to not include them.

An additional finding of this simulation was that autocorrelation in the dependent variables in a lagged time series model can effect the ability to predict the correct number of lags. While this effect was small, it was statistically significant across AIC and SBIC, abruptly

truncated and gradually declining lag coefficients, and number of observations. Additionally, it could have a noticeable effect with large degrees of autocorrelation.

Appendix A

```
capture program drop metrics
program define metrics, rclass

*** Set up files and memory:

clear
clear matrix
set mem 100m
set matsize 800

*** Set the number of observations and time series

set obs 1010
range time 1 1010
tsset time


*** Generate 10 random X's with autocorrelation from 0 to .9

local i = 0
foreach x of newlist x1-x10{
    gen `x' = rnormal(5,1)
    replace `x' = `i'*l.`x' + `x' if time > 1
    local i = `i' + .1
}


*** Define global macros for the different lag length parameterizations
***
*** These beta values (B's) specify a number of different parameterizations
*** such that we can analyze the following:
***     1) Abrupt vs gradually declining betas
***     2) number of lags


*** Abruptly truncated betas with 1 to 10 lags
*** $bset`i'_`j' gives you the coefficient of the jth lag for the ith specification
*** Abruptly truncated takes up $bset1_0 to $bset10_10
```

```
local i = 1
foreach betaSet of newlist bset1-bset10{
    local j = 1
    while `j' <= `i' {
        global `betaSet'_`j' = (1/(`i'))-.001
        local j = `j' + 1
    }
    local i = `i' + 1
}

*** Gradually declining betas with 0 to 10 lags
*** $bset`i'_`j' gives you the coefficient of the jth lag for the ith specification
*** gradually declining takes up $bset11_0 to $bset20_10

local i = 1
foreach betaSet of newlist bset11-bset20{
    local j = 1
    while `j' <= `i' {
        global `betaSet'_`j' = ((`i'-(`j'-1))/(.5*`i'^2 + .5*`i'))-.001
        local j = `j' + 1
    }
    local i = `i' + 1
}


*** Generate Y's based on the X's and a number of lags and betas

foreach xnum of varlist x1-x10 {
    forvalues v = 1/10 {
        global bset`v'
        forvalues lnum = 1/`v'{
            global bset`v' ${bset`v'} +${bset`v'_`lnum'}*l`lnum'.`xnum'
        }
        gen Y`xnum'_`v' = ${bset`v'} + rnormal(0,1) if time > `v'
    }
}
```

```
foreach xnum of varlist x1-x10 {
    forvalues v = 11/20 {
        global bset`v'
        local z = `v' -10
        forvalues lnum = 1/`z' {
            global bset`v' ${bset`v'} +${bset`v'_`lnum'}*l`lnum'.`xnum'
        }
        gen Y`xnum'_`v' = ${bset`v'} + rnormal(0,1) if time > `z'
    }
}

*** Run a varsoc on each of the Y's that were generated.
*** The matrix r(stats) contains the information about AIC and SBIC
*** The optimum number of lags is equal to the minimum in the AIC and SBIC columns
*** The forvalues loops find these values and return them in an r() value for later proc


foreach y of varlist Yx* {
    varsoc `y' if time > 10, maxlag(20)
    matrix A = r(stats)
    global aicValue = .  // Missing value (period) is the largest number in Stata
    global aicLag = 0
    global sbicValue = .
    global sbicLag = 0

    forvalues v = 0/20 {
        if $aicValue > A[(`v'+1),7] {
            global aicValue = A[(`v'+1),7]
            global aicLag = `v'
        }
    return scalar AIC`y' = $aicLag
    }

    forvalues v = 0/20 {
        if $sbicValue > A[(`v'+1),9] {
            global sbicValue = A[(`v'+1),9]
            global sbicLag = `v'
        }
    return scalar SBIC`y' = $sbicLag
    }

}

end
```