Bayesian Analysis using Stata

Bill Rising

StataCorp LP

2015 Oceania Stata Users Group Meeting
Canberra, ACT
25 September 2015

Goals

- Learn a little about Bayesian analysis
- Learn the core of how Bayesian analysis are implemented in Stata 14

Uncertainty as Probability

- In the frequentist world, probabilities are long-run proportions of repeated identical experiments
  - In some ways, this means we never know any probabilities of any events
- In the Bayesian world, probabilities are an expression of uncertainty
  - The advantage of the Bayesian viewpoint is that it allows talking about probabilities for events which cannot be repeated
    - What is the chance of a major earthquake in Alaska this year?
    - What is the chance that Australia takes the 2015 Rugby World Cup?
  - The disadvantage is that these probabilities become subjective

Bayesian Analysis

- Uncertainty about parameters is expressed via a prior distribution $p(\boldsymbol{\theta})$
  - The prior distribution is necessarily subjective
  - If there is little knowledge about possible values, vague or non-informative priors get used
- The dataset $\mathbf{y}$ is used to update these priors into posterior distributions via Bayes rule

$$p(\boldsymbol{\theta}|\mathbf{y}) = \frac{p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{y})}$$

  - $p(\mathbf{y}|\boldsymbol{\theta})$ is the likelihood
  - $p(\mathbf{y})$ is the marginal density of the data

$$p(\mathbf{y}) = \int_{\boldsymbol{\theta}} p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})$$

  - This last integral has been the bugaboo

Advantages and Disadvantages of Bayesian Analysis

- Advantages
  - Theoretically should allow updating knowledge with past experience
  - Can speak directly about probabilities instead of applying long-run proportions to a single event
    - Think of confidence intervals: have long-run chance of catching the parameter value, but know nothing about the current estimate
  - Can choose among multiple competing hypotheses
- Disadvantages
  - Could be worried about subjectivity

Why Has Bayesian Analysis Become More Popular

- Computational speed allows rapid but good approximations of the marginal density of the data
  - Before computational horsepower could be used, only a small set of models could be estimated
- All the magic comes from Markov Chain Monte Carlo (MCMC) methods
  - These sample points from the not-fully-specified density in such a way that if left running forever, the density of simulation points would equal the target density

Implementation in Stata 14

- In Stata 14, the estimation portion of Baysian analysis is implented by the `bayesmh` command
  - `mh` for Metropolis-Hastings
- We will see how this works, both via point-and-click and syntactically
- We will look at some diagnostics and other post-estimation tools

A Simple Story

- We'll work with a very simple dataset measuring counts
- Here is our simulated story:
  - We've collected data from 70 people in Canberra about the number of parking tickets they've gotten in the last year
  - We would like to get some concept of the rate the people get the tickets
  - We will do this based on the rumor that Canberra is particularly finicky about parking
- We'll simulate a dataset as though the true number of parking fines per year per person is 1.3

```
. do makepois
```

- Let's see the mean count for this simulation

```
. sum y
```

Starting a Bayesian Analysis: the Prior

- We would now like to do a Bayesian investigation of the rate λ of getting fined
  - Suppose we are truly interested whether the rate of fines is over one per year per person
- To start out, we need to specify a prior distribution
- How would this possibly be done?
  - We could try to use a vague prior which has very little information in it
  - We could try to elicit the opinions of experts
- We'll start with a vague prior

Choosing a Vague Prior

- Vague priors are only vaguely defined: they ought to cover all remotely plausible values without favoring any values
- We will choose a flat prior, meaning that all possible ticketing rates have the same probability
  - Because this means that we need a probability density proportional to 1 over the interval 0 to ∞, this is an improper prior
  - Improper priors should typically be avoided, but this will help the exposition here
- So, for us, $p(\lambda) \propto 1$ for $0 < \lambda < \infty$
  - Clearly, like continuous-time white noise, this is impossible but helpful

Specifying our Model: the Interface

- We will start by using the point-and-click interface
- There are two ways to access this
  - Either select **Statistics** > **Bayesian analysis** > **Estimation**
  - Or type `db bayesmh` in the command window
- We will choose what we would like to do now, and then come back to the full range of possible models

Choosing the Likelihood Model

- We would like a univariate linear model
- Clicking the drop-down menu for the *Dependent variable* and choose y
- We have no independent variables
- Choose *Poisson regression* as the ***Likelihood model***
- We can leave the *Exposure variable* blank
- Tick the *Do not exponentiate linear predictor*
    - This will cause our out output to report rates instead of the natural log of rates

Specifying the Prior

- Click on the **Create...** button for the ***Priors of model parameters***
- From the *Parameters specification* dropdown, choose `{y:_cons}`
  - This is because we are modeling only the constant term without any covariates
- We will choose the *Flat prior* item
- Click **OK** to dismiss the subdialog

Making Our Computations Reproducible

- We should set a random seed for this MCMC
  - This will make sure that we can show our result in the future
- Click on the ***Simulation*** tab
- We'll put 7434 as the random seed
  - This is an arbitrary non-negative integer

Computing the Posterior

- We are already done specifying this simple model, so click the **Submit** button
- The command gets issued

```
. bayesmh y, likelihood(poisson, noglmtransform) ///
      prior({y:_cons}, flat) rseed(7434)
```

- Stata churns through the MCMC computations to find the posterior distribution
- Stata reports the results

General Notes about the Output

- At the top, you see `Burn In ...` followed by `Simulation ...` as notifications
  - These would be for seeing progress in very computationally intensive models
- We see the two elements we need to specify for any Bayesian analysis: the Likelihood model and the Prior distribution
- There is information about how the MCMC sampling was done
- There is information about summary statistics of the posterior distribution
  - Recall that we are not specifically trying to estimate mean values; we are finding a posterior distrbution

Output Specifics: MCMC

- By default, Stata uses a burn-in of 2,500 iterations
  - This is used to tune the adaptive model and to give time for the simulation to reach the main part of the posterior distribution
- By default, Stata runs the MCMC chain for 10,000 iterations
- The acceptance rate is the rate that new picks from the distribution are accepted
- The efficiency is relative to independent samples from the posterior distribution

Output Specifics: Regression Table

- The mean of our posterior distribution for the arrival rate is 1.27
- The standard deviation of the posterior distribution is 0.136
- The MCSE of 0.0028 is the standard error of estimation of the mean due to our using MCMC to find the posterior distribution
  - How much the posterior mean would vary from run to run if we used different random seeds
- The median is the median of the posterior distribution
- The probability that the arrival rate is between 1.021 and and 1.548 is 95%
  - Note this is not a trapping probability for unknown future samples

Starting with Postestimation

- We can see what postestimation commands are available by typing

  ```
  . db postest
  ```

- Now click on the disclosure control next to *Bayesian analysis*
- Select the *Graphical summaries and convergence diagnostics* item
- Click on the **Launch** button

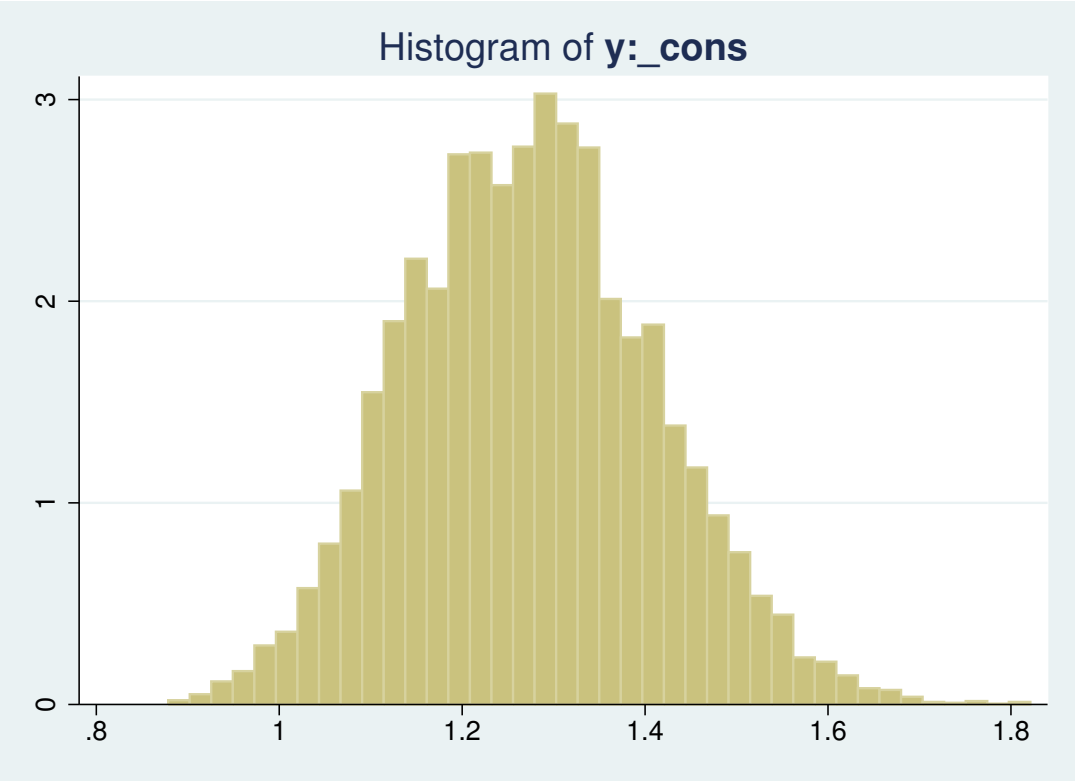
Investigating the Posterior

- We can draw a picture of the posterior distribution in a couple of ways
- To make a histogram, select the *Histograms* graph type
- To make life simple select the *Graphs for all model parameters* radio button
- Click on the **Submit** button

Histogram of the Posterior

- Here is the histogram version of the posterior distribution

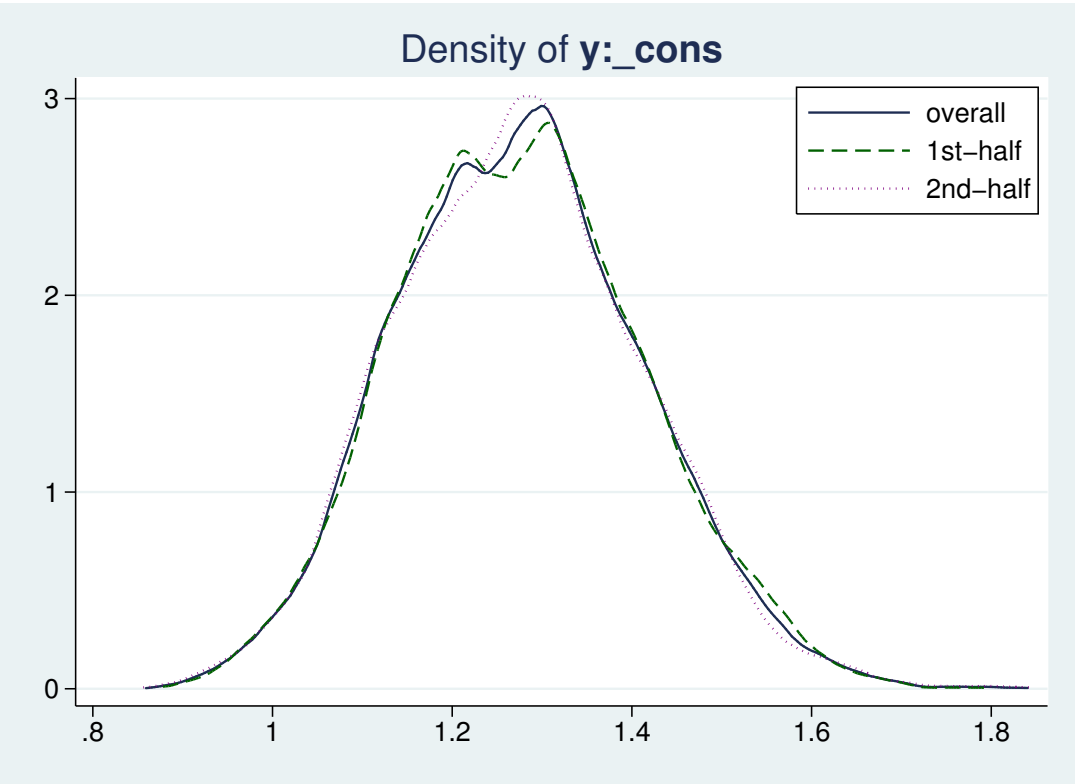
```
. bayesgraph histogram _all
```



Density Plot of the Posterior

- To get a density plot, select the *Density plots* graph type
- Click on the **Submit** button

```
. bayesgraph kdensity _all
```



Finding the Probability the Rate is Larger than 1

- Navigate back to the *Postestimation Selector* dialog box
- Select the *Interval hypothesis testing* menu item
- Choose `{y:_cons}` parameter from the *Test model parameter* list
- Enter 1 as the *Lower* bound and leave `.` as the *Upper* bound
- Click the **Submit** button

```
. bayestest interval ({y:_cons}, lower(1))
```

- We can read off the probability as 0.98
  - This is a true probability
  - It is a subjective probability based on our flat prior

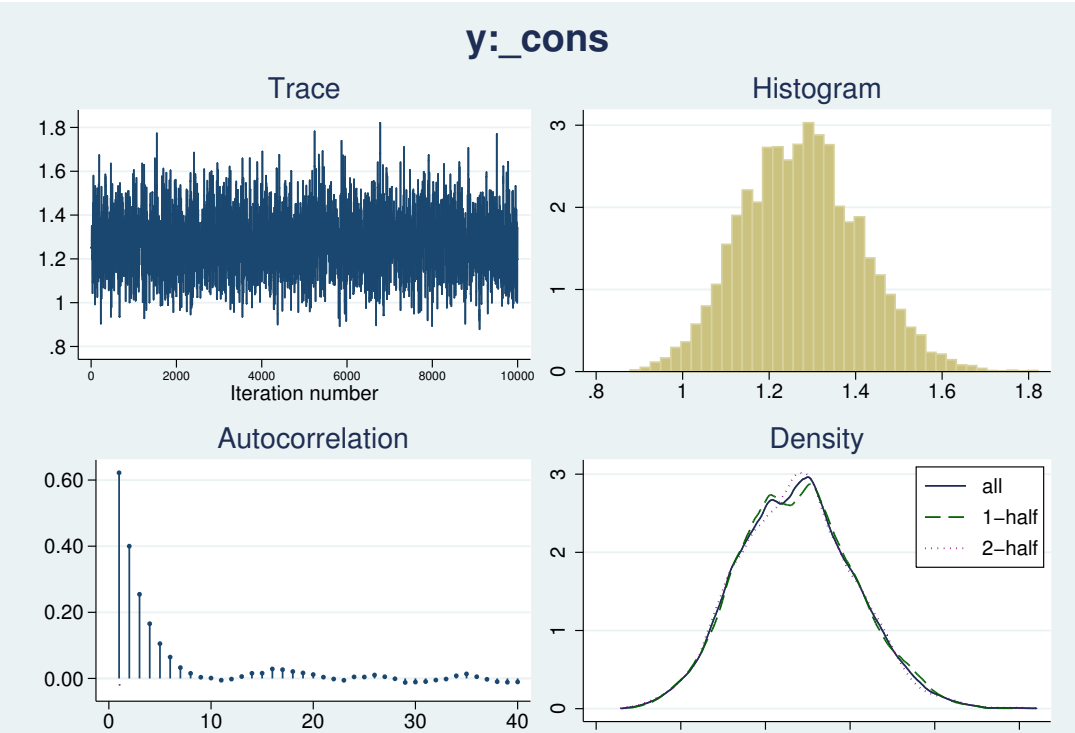
How MCMC Can Break

- There are multiple ways that MCMC can give bad answers
  - It can mix poorly, meaning either that
    - New candidate points for the simulation get rejected too often
    - The jumps are too small to cover the distribution
  - It can have bad initial values
    - These should be irrelevant because of the long burn-in sequence
    - But... if there is poor mixing this might not be the case
    - This leads to what is called 'drift'

MCMC Diagnostics

- There is a simple tool for looking at the standard diagnostics all at once
- Select *Multiple diagnostics in compact form* in the *bayesgraph* dialog, and press **Submit**

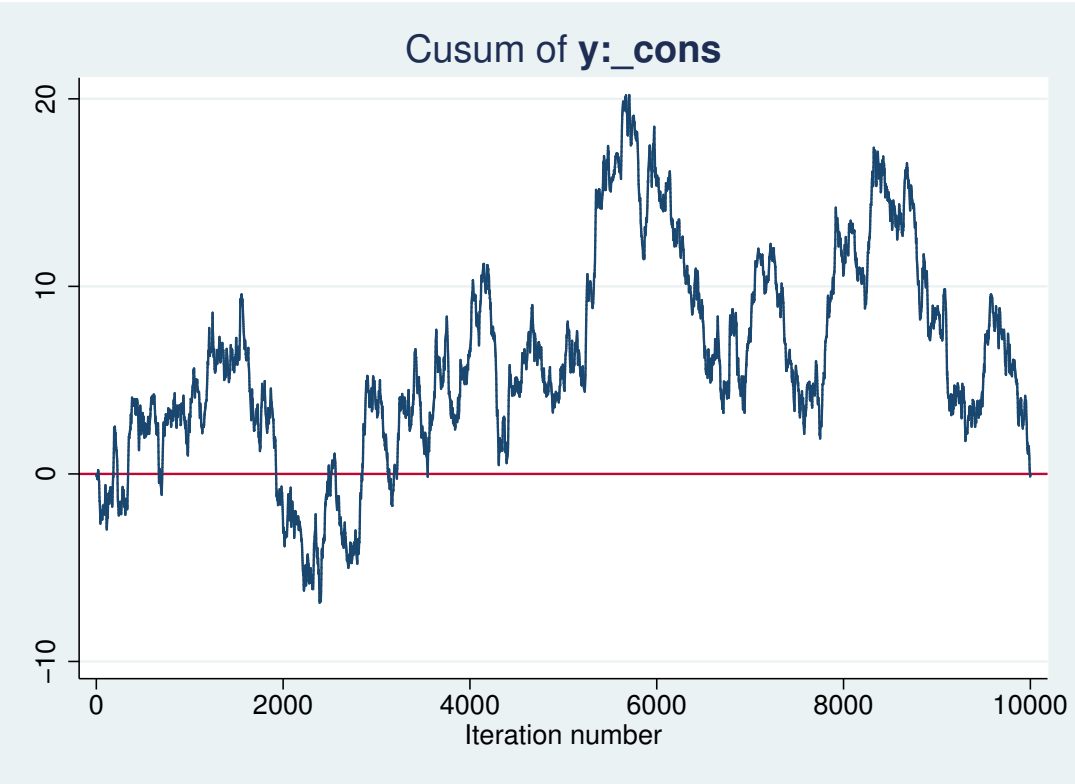
```
. bayesgraph diagnostics _all
```



Looking for Drift

- The cusum (short for cumulative sum) plot is used to look for small step size and drift
- Select *Cumulative sum plots* and press **Submit**

```
. bayesgraph cusum _all
```



Simple Diagnostic Conclusion

- Everything looks fine because there is no sign of bad mixing or drift

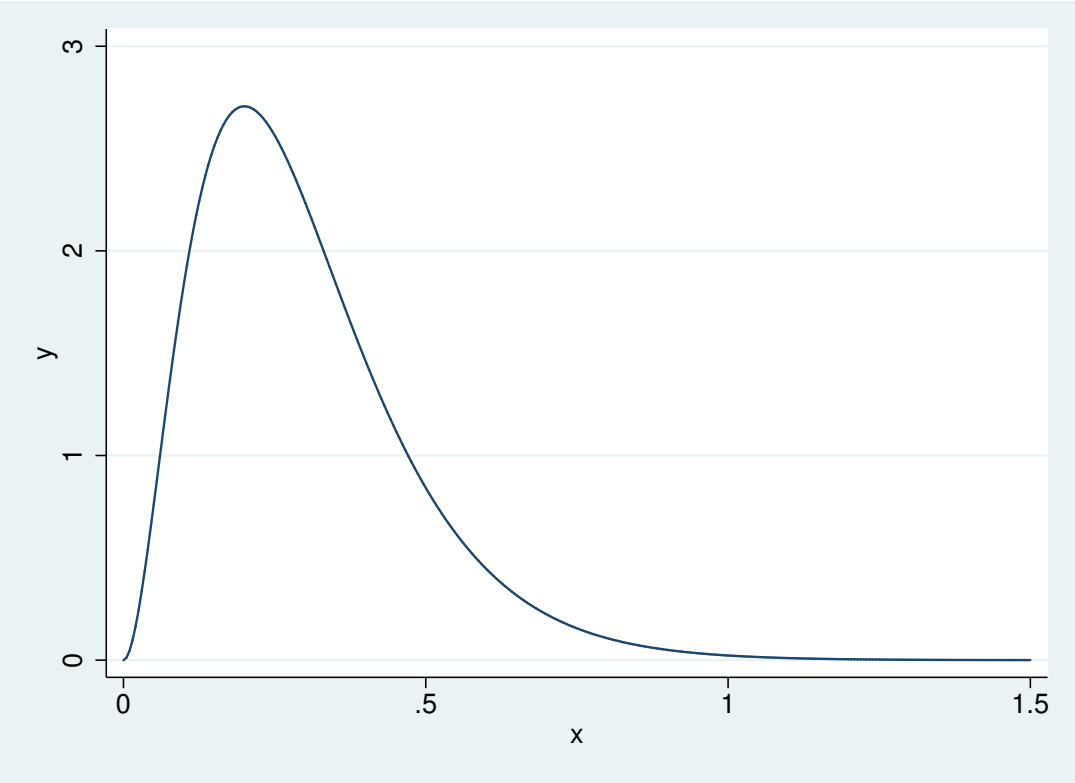
Playing with Different Priors

- Suppose we talk to people in Sydney, Melbourne, Adalaide, and Brisbane
- They all agree that the the rate of fines should be about 1 every 3 years, with little chance of averaging more than 1 fine per year
  - Thus, they are completely incorrect about Canberra
- Based on this, a good prior would be a Gamma(3, 0.1)

Aside: Graph of the Prior

- Here is a graph of the Gamma(3, 0.1) distribution

```
. twoway function y = gammaden(3,0.1,0,x), range(0 1.5)
```



Specifying a New Prior

- Type `db bayesmh` to get our dialog box back
- Select the *Prior 1* prior
- Click on the **Edit** button
- Choose *Gamma distribution*
- Enter 3 as the *Shape* and 0.1 as the *Scale*
- Click on the `OK` button to dismiss the subdialog

Changing the Seed

- Go to the ***Simulation*** tab
- Change the random seed to some other number, say 9983
- Click on the `Submit` button to run the analysis

```
. bayesmh y, likelihood(poisson, noglmtransform) ///
      prior({y:_cons}, gamma(3,0.1)) rseed(9983)
```

What Happened?

- We can see that the mean of the posterior distribution is smaller
  - We should, however, be encouraged that the mean is only somewhat smaller despite the very-different prior
- If we now compute our probability that the rate is larger than 1, though: 0.88

```
. bayestest interval ({y:_cons}, lower(1))
```

Specifying Our Own Likelihood

- What if we wanted a likelihood which is not one of the 10 built-in likelihoods?
- We can specify this by using the `likelihood()` option with the `llf()` suboption
- We just need an example to show this...

Changing the Problem

- Suppose now that our sample came just from those who had had a ticket in the last year

```
. drop if y == 0
```

  - We've lost quite a bit of our sample
- We cannot use the same likelihood model as we had before
- Instead, we have a truncated Poisson, where the probability of 0 fines has become 0
- Truncated Poisson distributions are not a part of Stata's suite, so we need to do some math

Writing Our New Likelihood Model

- Here is the Poisson distribution with parameter λ is

$$p(y) = \frac{\lambda^y \mathrm{e}^{-\lambda}}{y!}; \qquad y = 0, 1, 2 \ldots$$

- If y cannot be zero, we just need to rescale to get the total probability to be 1:

$$p(y) = \frac{\lambda^y \mathrm{e}^{-\lambda}}{y!(1 - \mathrm{e}^{-\lambda})}; \qquad y = 1, 2 \ldots$$

- From this, our log likelihood becomes

$$y \ln(\lambda) - \lambda - \ln(y!) - \ln(1 - \mathrm{e}^{-\lambda})$$

Substitutable Expressions

- The way we tell Stata to use the log-likelihood function is by using a *substitutable expression*
- We just need to replace
  - Symbols with the variables that represent them
  - Coefficient names to replace parameters

Specifying Our New Likelihood Model

- In our case
  - `y` (the variable) replaces y the count symbol
  - `{y:_cons}` replaces λ
- This gives the straightforward but unwieldy expression

```
y*ln({y:_cons})-{y:_cons}-lngamma(y+1)-ln(1-exp(-{y:_cons}))
```

Working from Do-files

- Now the commands are becoming complicated enough that typing as we go will be unhelpful
- Let's open up a project file for this talk

```
. projman bayes
```

Finally: Analyzing the Truncated Gamma

- We can run our analysis with this do-file

  ```
  . do trunc_pois
  ```

  - The `saving()` option has been added because we will need it if we would like to compare this model to another later
  - We stored the model for later comparisons
- The mean from our posterior distribution now overshoots the true mean
  - This could happen because there were too many 0-valued observations in the original dataset

Truncated Gamma Notes

- Notice the `note: invalid initial state` warning under `Burn in ...`:
  - This happened here because Stata started λ at 0, which is not a valid rate
  - This should only worry us if the efficiencies are low or if the chain did not converge
- Just as before, we can look at the diagnostics (not shown)
- Here is the probability that the rate of fines is over 1

```
. bayestest interval ({y:_cons}, lower(1))
```

A Competing Likelihood

- Suppose we suspect that there could be overdispersion or underdispersion for our model
- We could try specifying a likelihood which accommodates this: the generalized Poisson distribution
- Here is one parameterization

$$p(y) = \frac{1}{y!}\left(\frac{\mu}{1+\alpha\mu}\right)^{y}(1-\alpha y)^{y-1}\exp\left\{-\frac{\mu(1+\alpha y)}{1+\alpha\mu}\right\}; y = 0, 1, 2, ...$$

- This distribution has mean μ and variance $\mu(1+\alpha\mu)^2$
  - Thus, if $\alpha > 0$ there is overdispersion; if $\alpha < 0$ there is underdispersion

Estimating This Competing Likelihood

- We can once again specify our own log likelihood:

$$llf(y) = -\ln(y!) + y\left(\ln(\mu) - \ln(1+\alpha\mu)\right) + (y-1)\ln(1+\alpha y) - \frac{\mu(1+\alpha y)}{1+\alpha\mu} - \ln\left(1 - \exp(-\frac{\mu}{1+\alpha\mu})\right)$$

  - The last term comes from rescaling because the distribution is truncated

- Luckily, this mess has been put in a do-file

```
. do trunc_gpois
```

Uh oh! Bad Efficiency

- If we look at the efficiencies, we can see that one of the parameters probably has high autocorrelations
- First, let's see which parameter had which efficiency by looking at effective sample sizes

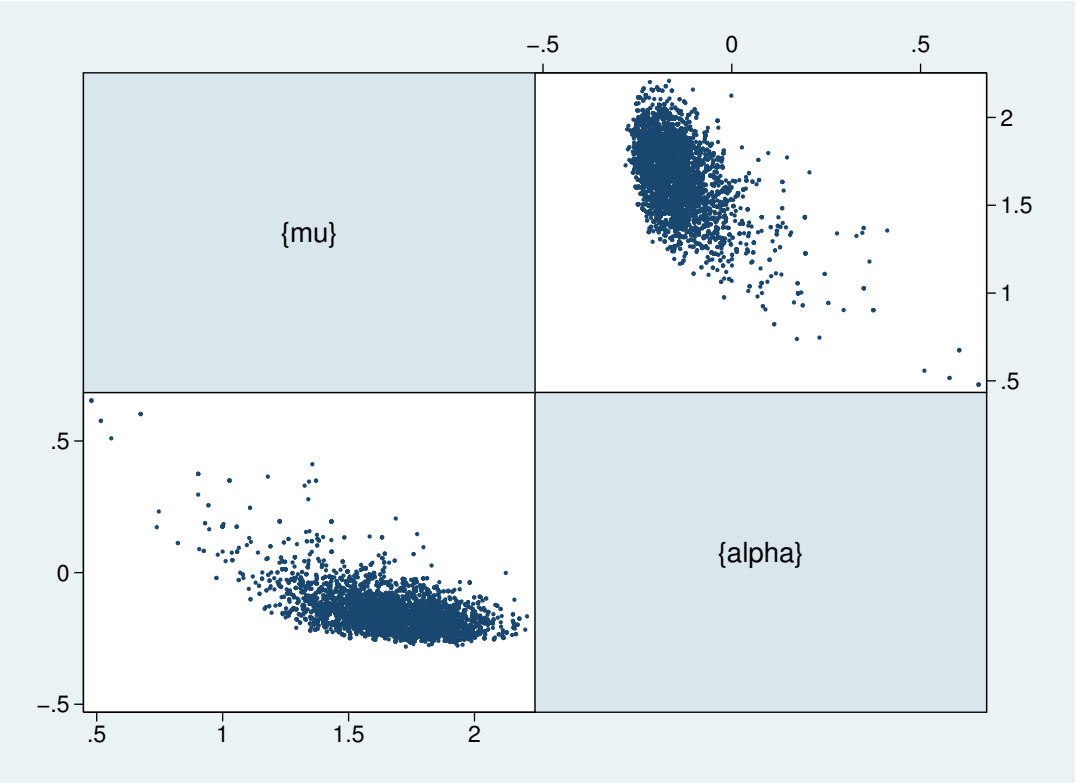
```
. bayesstats ess _all
```

- We should investigate this

Plotting Simulations

- We can make a scatterplot matrix of the simulation values

```
. bayesgraph matrix {mu} {alpha}
```



Correlated Simulations

- Correlated MCMC simulation values slow down the MCMC chain, as do possibly illegal values
- One solution we could try here would be to transform the parameters to make their range extend over the whole real line
  - This is hard here, because the range of α depends on μ
- We might also try specifying legal initial values

```
. do trunc_gpois2
```

- This seemed to help
  - Try experimenting with other starting values if you like

Extending the Chain

- If we would like to get an effective sample size which is close to what we had for the truncated poisson model, we need to extend the chain
- The `mcmcsize(25000)` option does this

```
. do trunc_gpois3
```

Comparing Competing Models

- We can now see which of the two models we prefer
- This is done using the `bayestest model` command
- Being Bayesians, we assign prior probabilities to each of the models, and then compute their posterior probabilities given our data
- We have no reason to think one model is better than the other so we'll use the default of equally likely

```
. bayestest model trunc*
```

- We now think that there is a 96% chance that simple truncated poisson is true

Aside: Bayesian Hypothesis Testing

- One wonderful part of the Bayesian world is that more than two models may be compared
- One must take care that hypotheses are plausible
  - No point values for continuous variables, for example, unless they are 0 values for something that might not exist
- Sometimes it makes sense to have prior distributions which are not evenly distributed
  - There can be a decision-theoretic reason for this, for example different costs associated with falsely conclusions
- This is far more flexible than the typical us-versus-them hypothesis testing

Information Criteria

- We can also compare models using the deviance information criterion (DIC) and Bayes factors

```
. bayesstats ic trunc*
```

- The smaller DIC for the `trunc_gpois` model says that it should do a better job producing a similar dataset
- The log(BF) column gives the log of odds that the `trunc_gpois` model is true
  - Here: `ln(0.0370/0.9630)`
- The Bayes factor will always give the same subjective result as assuming equal prior probabilities for models

Linear Regression

- All we've been doing is looking at a dataset of counts

```
. save pois_plus, replace
```

- Now let's try playing with linear regressions
- Open up the `autometric` dataset

```
. use autometric
```

  - Made for all countries except the US, Liberia, and Myanmar

Modeling Energy Usage

- We'd like to measure energy usage of these cars
- Perhaps: regressing `lp100km` on `weight`, `displacement` and `foreign`
- Let's go back to the dialog box for teaching purposes
  - Reset the dialog box by clicking the big **R** button

Filling in the Dialog Box

- This will take a little effort, but specify
  - `{var}` as the variance for the likelihood
  - Normals with large variances for the coefficients
  - Jeffries prior for the prior of `{var}`
  - A random seed of 142857
- Click on `OK` to submit and close

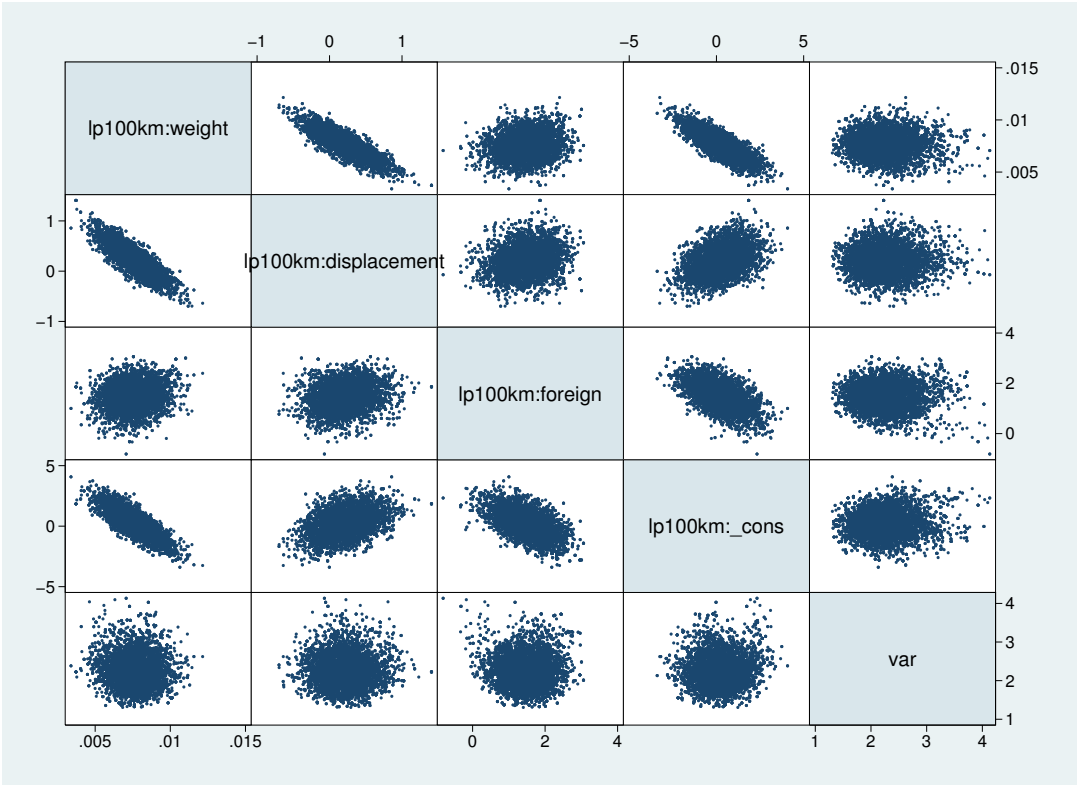
```
. do reg
```

- The model converges, but not at all efficiently

Looking at the Problem

- Draw a graph matrix to see the problems

```
. bayesgraph matrix _all
```



Partial Fix Number 1

- If we mean center the weight and the displacement, we'll get rid of some of the correlation between their simulated values and those of the intercept

```
. sum weight displacement
```

- While we're at it, let's make weight no so big

```
. gen wt1300 = (weight-1300)/1000
. gen displacement3 = displacement - 3
```

- Now let's see what happened

```
. do regcent
```

Partial Fix Number 2

- We've chosen very special prior distributions for our model
  - Normal priors for a normal regression are semi conjugate
  - This means that they produce normal posterior distributions
    - This means we know the posterior distrobution explicity
- So... we can use Gibbs sampling here
  - This is a special case of Metropolis-Hastings which exploits knowledge fo the closed form
- As a side effect, we will estimate each of the predictors separately
  - The default is to estimate them all at once

Result of Gibbs Sampling

- Here is our Gibbs sampler

```
. do reggibbs
```

- This has helped a bunch with everything except the correlated predictors
- So: collinearity is a problem here, too!
- Our only solution is to run the chain much longer

```
. do reggibbs2
```

What We Have Seen

- Use of part of the GUI for Bayesian analysis in Stata
- Specification of a non-standard likelihood
- Specification of priors
- Basic Bayesian estimation
- Basic Baysian model comparison
- Gibbs samplers
- Centering

What We Have Not Seen

- Complex models
  - There are many many examples in the manuals
- Writing our own evaluators
  - If you have a likelihood function which is not the sum of the likelihoods for each of the observations, you can write a specially-formed evaluator program
    - This is similar in kind to the `ml` command

Conclusion

- We've just touched on what can be done
- I hope this has been somewhat informative