1 Introduction

1.1 Goals

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

1.2 Brief Glimpse into Bayesian Analysis

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

\[
p(\theta | y) = \frac{p(y | \theta) p(\theta)}{p(y)}
\]

- \( p(y | \theta) \) is the likelihood
- \( p(y) \) is the marginal density of the data

\[
p(y) = \int_{\theta} p(y | \theta) p(\theta)
\]

⋆ 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 Bayesian analysis is implemented by the \texttt{bayesmh} command
  - \texttt{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
2 Bayesian Analysis in Stata 14

2.1 Starting Simple

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

```stata
. do makepois
. set more off
. clear
. * pick a seed for reproducibility
. set seed 1800
. * set the number of observations
. set obs 70
number of observations (_N) was 0, now 70
. * create the observations
. gen y = rpoisson(1.3)
. label var y "Parking tickets in Canberra"
. label data "The IKEA of datasets: one variable of counts"

. save pois
file pois.dta saved

end of do-file
```
- Let’s see the mean count for this simulation

```stata
. sum y

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>70</td>
<td>1.257143</td>
<td>1.099219</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>
```

Starting a Bayesian Analysis: the Prior

- We would now like to do a Bayesian investigation of the rate $\lambda$ 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 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)

  Burn-in ...
  Simulation ...

  Model summary
  -----------------------------------------------
  Likelihood:
  y ~ poisson({y:_cons})
  Prior:
  {y:_cons} ~ 1 (flat)
  -----------------------------------------------

  Bayesian Poisson regression
  Random-walk Metropolis-Hastings sampling

  MCMC iterations = 12,500
  Burn-in = 2,500
  MCMC sample size = 10,000
  Number of obs = 70
  Acceptance rate = .4271
  Log marginal likelihood = -102.22367
  Efficiency = .2315

  | Equal-tailed
  |          y          | Mean   | Std. Dev. | MCSE  | Median | [95% Cred. Interval] |
  |            |            |         |        |       |                   |
  | _cons    | 1.274535  | .1358713 | .002824 | 1.274437 | 1.020925 - 1.548038 |

- 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 distribution
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

\[
\text{bayestest interval } (y: \_cons, \text{ lower}(1))
\]
Interval tests \hspace{1cm} MCMC sample size = 10,000
\[
\text{prob1} : \{y: \_cons\} > 1
\]

<table>
<thead>
<tr>
<th>Mean</th>
<th>Std. Dev.</th>
<th>MCSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>.9837</td>
<td>0.12663</td>
<td>.0024803</td>
</tr>
</tbody>
</table>

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

2.2 Looking More Carefully
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

\[
\text{bayesgraph diagnostics } \_\text{all}
\]
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 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
  
  \[ \text{twoway function } y = \text{gammaden}(3,0.1,0,x), \text{ range(0 1.5)} \]

Specifying a New Prior

- Type \text{db bayesmh} to get our dialog box back
- Select the \text{Prior 1} prior
- Click on the \text{Edit} button
- Choose \text{Gamma distribution}
- Enter 3 as the \text{Shape} and 0.1 as the \text{Scale}
- Click on the \text{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

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

Burn-in ...
Simulation ...

Model summary
-----------------------------------------------------------------------------------
Likelihood:  
y ~ poisson({y:_cons})
Prior:  
{y:_cons} ~ gamma(3,0.1)
-----------------------------------------------------------------------------------

Bayesian Poisson regression MCMC iterations = 12,500
Random-walk Metropolis-Hastings sampling Burn-in = 2,500
Number of obs = 70
Acceptance rate = .4313
Log marginal likelihood = -107.68681 Efficiency = .2345

<table>
<thead>
<tr>
<th>Equal-tailed</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
</tr>
<tr>
<td>----</td>
</tr>
<tr>
<td>_cons</td>
</tr>
</tbody>
</table>

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

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

Interval tests MCMC sample size = 10,000

```
<table>
<thead>
<tr>
<th>prob1 : {y:_cons} &gt; 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>.8814</td>
</tr>
</tbody>
</table>
```

2.3 Changing the Problem

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
  
  \[ \text{drop if } y == 0 \]

  (23 observations deleted)

  - 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 \( \lambda \) is

  \[
p(y) = \frac{\lambda^y e^{-\lambda}}{y!}; \quad y = 0, 1, 2, \ldots
  \]

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

  \[
p(y) = \frac{\lambda^y e^{-\lambda}}{y!(1 - e^{-\lambda})}; \quad y = 1, 2, \ldots
  \]

- From this, our log likelihood becomes

  \[
y \ln(\lambda) - \lambda - \ln(y!) - \ln(1 - e^{-\lambda})
  \]

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 \( \lambda \)

- This gives the straightforward but unwieldy expression

  \[
y*\ln(\{y:_cons\})-\{y:_cons\}-%gamma(y+1)-\ln(1-exp(-\{y:_cons\}))
  \]

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

  . ** truncated poisson estimation
  . bayesmh y, prior({_cons}, flat) ///
  > rseed(3772) saving(trunc_pois) ///
  > likelihood(llf(y*ln({_cons})-{_cons}-lngamma(y+1)-ln(1-exp(-{_cons}))))
  > )

  Burn-in ...
  note: invalid initial state
  Simulation ...

  Model summary
  ---------------------------------------------------------------------
  Likelihood: y ~ logdensity(y*ln({_cons})-{_cons}-lngamma(y+1)-ln(1-exp(-{_cons})))
  Prior: {_cons} ~ 1 (flat)
  ---------------------------------------------------------------------

  Bayesian regression Random-walk Metropolis-Hastings sampling
  MCMC iterations = 12,500
  Burn-in = 2,500
  MCMC sample size = 10,000
  Number of obs = 47
  Acceptance rate = .4315
  Log marginal likelihood = -56.819423 Efficiency = .2378

file trunc_pois.dta saved

  . ** storing the model for later
  . est store trunc_pois

  . end of do-file

  ◦ 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 \( \lambda \) 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))
    
    Interval tests MCMC sample size = 10,000
    
    prob1 : \{y:_cons\} > 1
    
    | Mean    | Std. Dev. | MCSE   |
    |----------|-----------|--------|
    | prob1    | .9894     | 0.10241 | .0017583 |
    
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

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

  • This distribution has mean \( \mu \) 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:

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

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

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

  . do trunc_gpois
  
  . ** truncated gen'l poisson estimation
  . ** specified nocons, so that the two parameters \{mu\} and \{alpha\}
  
  . ** could both be specified by name
  
  . bayesmh y, nocons prior({mu}, uniform(0,100)) prior({alpha}, flat) ///
  > rseed(40213) saving(trunc_gpois) ///
  > likelihood(llf(-lngamma(y+1) + y*(ln({mu}) - ln(1 +{alpha}*{mu})) ///
  > + (y-1)*ln(1 + {alpha}*y) - ({mu}*(1 + {alpha}*y))/(1 +{alpha}*{mu}) ///
  > - ln(1 - exp(-{mu}/(1+{alpha}*{mu})))))

Burn-in ...

note: invalid initial state
Simulation ...
Likelihood:
\[ y \sim \logdensity(\text{expr1}) \]

Priors:
\[
\{\mu}\sim \text{uniform}(0,100) \\
\{\alpha}\sim 1 \text{ (flat)}
\]

Expression:
\[
\text{expr1} : -\text{lngamma}(y+1)+y*(\ln(\{\mu}\) - \ln(1 +\{\alpha}\*\{\mu}\))+(y-1)*\ln(1 - \exp(-\{\mu}\)/(1+\{\alpha}\*\{\mu}\)))
\]

Bayesian regression
Random-walk Metropolis-Hastings sampling
MCMC iterations = 12,500
Burn-in = 2,500
MCMC sample size = 10,000
Number of obs = 47
Acceptance rate = .2366
Efficiency: min = .05308
avg = .08066
max = .1082

Log marginal likelihood = -59.917961

<table>
<thead>
<tr>
<th>Equal-tailed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean Std. Dev. MCSE Median [95% Cred. Interval]</td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>{\mu}</td>
</tr>
<tr>
<td>{\alpha}</td>
</tr>
</tbody>
</table>

file trunc_gpois.dta saved

. ** storing the model for later
. est store trunc_gpois

end of do-file

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

  Efficiency summaries MCMC sample size = 10,000

<table>
<thead>
<tr>
<th>ESS Corr. time Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>{\mu}</td>
</tr>
<tr>
<td>{\alpha}</td>
</tr>
</tbody>
</table>

- We should investigate this

Plotting Simulations

- We can make a scatterplot matrix of the simulation values
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 $\alpha$ depends on $\mu$.
- We might also try specifying legal initial values.

```stata
. bayesmh y, nocons prior({mu}, uniform(0,100)) prior({alpha}, flat) ///
   rseed(40213) saving(trunc_gpois2) ///
   likelihood(llf(-lngamma(y+1) + y*(ln({mu}) - ln(1 +{alpha}*{mu})) ///
      + (y-1)*ln(1 + {alpha}*y) - ({mu}*(1 + {alpha}*y))/(1 +{alpha}*{mu}) ///
      - ln(1 - exp(-{mu}/(1+{alpha}*{mu})))) ///
     initial({mu} 1 {alpha} 0)

Burn-in ...
Simulation ...

Model summary
------------------------------------------------------------------------------
Likelihood: y ~ logdensity(<expr1>)

Priors:
{mu} ~ uniform(0,100)
{alpha} ~ 1 (flat)

Expression:
expr1 : -lngamma(y+1)+y*(ln({mu}) - ln(1 +{alpha}*{mu}))*y-({mu}*(1 +{alpha}*y))/(1 +{alpha}*{mu})-ln(1 - exp(-{mu}/(1+{alpha}*{mu})))
```

Bayesian Analysis using Stata © StataCorp LP
Bayesian regression
Random-walk Metropolis-Hastings sampling
MCMC iterations = 12,500
Burn-in = 2,500
MCMC sample size = 10,000
Number of obs = 47
Acceptance rate = .2634
Efficiency: min = .09876
avg = .1121
Log marginal likelihood = -60.126325
max = .1255

| Equal-tailed |
|-------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| Mean        | Std. Dev.       | MCSE            | Median          | [95% Cred. Interval] |
-------------|-----------------|-----------------|-----------------|-----------------|-----------------|
mu     | 1.685297       | .1746314        | .005557         | 1.694533         | 1.324338         | 2.001587         |
alpha  | -.1648068      | .0583825        | .001648         | -.1739788        | -.2474469        | -.0151704        |

file trunc_gpois2.dta saved
.
. ** storing the model for later
. est store trunc_gpois
.
end of do-file

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
  . ** truncated gen’l poisson estimation
  . ** specified nocons, so that the two parameters {mu} and {alpha}
  . ** could both be specified by name
  . bayesmh y, nocons prior({mu}, uniform(0,100)) prior({alpha}, flat) ///
  > rseed(40213) saving(trunc_gpois3) ///
  > likelihood(llf(-lngamma(y+1) + y*(ln({mu}) - ln(1 +{alpha}*{mu})) - (y-1)*ln(1 + {alpha}*y) - ({mu}*(1 + {alpha}*y))/(1 +{alpha}*{mu}) - ln(1 - exp(-{mu}/(1+{alpha}*{mu}))))) ///
  > initial({mu} 1 {alpha} 0) ///
  > mcmcsize(25000)

Burn-in ... Simulation ...

Model summary
---------------------------------------------------------------------------------
Likelihood:
  y ~ logdensity(<expr1>)

Priors:
  {mu} ~ uniform(0,100)
  {alpha} ~ 1 (flat)
Expression:
\[ \text{expr1} : \text{expr1} = -\ln \Gamma(y+1) + y(y \ln(\mu) - \ln(1 + \alpha \mu)) + (y-1) \ln(1 + \alpha y) - \frac{\mu}{1 + \alpha y} \ln(1 - \exp(-\frac{\mu}{1 + \alpha y})) \]

---

Bayesian regression
Random-walk Metropolis-Hastings sampling
MCMC iterations = 27,500
Burn-in = 2,500
MCMC sample size = 25,000
Number of obs = 47
Acceptance rate = 0.2641
Efficiency: min = 0.1003
avg = 0.1026
max = 0.1049

Log marginal likelihood = -60.079039

<table>
<thead>
<tr>
<th>Equal-tailed</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>MCSE</th>
<th>Median</th>
<th>[95% Cred. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>1.685861</td>
<td>0.1765273</td>
<td>0.003525</td>
<td>1.695722</td>
<td>1.304009 1.999089</td>
</tr>
<tr>
<td>alpha</td>
<td>-0.1642611</td>
<td>0.0613568</td>
<td>0.001198</td>
<td>-0.1746719</td>
<td>-0.2463381 -0.0211519</td>
</tr>
</tbody>
</table>

file trunc_gpois3.dta saved

** storing the model for later
. est store trunc_gpois

end of do-file

---

Comparing Competing Models

- We can now see which of the two models we prefer
- This is done using the \texttt{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
  . \texttt{bayestest model trunc*}

Bayesian model tests

|          | log(ML) | P(M) | P(M|y) |
|----------|---------|------|-------|
| trunc_pois |  \text{-56.8194} | 0.5000 | 0.9630 |
| trunc_gpois |  \text{-60.0790} | 0.5000 | 0.0370 |

Note: Marginal likelihood (ML) is computed using Laplace-Metropolis approximation.

- 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*

```
Bayesian information criteria

<table>
<thead>
<tr>
<th>DIC</th>
<th>log(ML)</th>
<th>log(BF)</th>
</tr>
</thead>
<tbody>
<tr>
<td>trunc_pois</td>
<td>114.3289</td>
<td>-56.81942</td>
</tr>
<tr>
<td>trunc_gpois</td>
<td>109.3351</td>
<td>-60.07904</td>
</tr>
</tbody>
</table>
```

Note: Marginal likelihood (ML) is computed using Laplace-Metropolis approximation.

- 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

2.4 Something A Little More Complex

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
  (auto data in metric units)
  - 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 \( \text{lp100km} \) on \( \text{weight} \), \( \text{displacement} \) and \( \text{foreign} \)
- 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
  . * using centering
  . bayesmh \text{lp100km} \text{weight} \text{displacement} \text{foreign}, ///
  > likelihood(normal({var})) ///
  > prior({weight}, normal(0,1000)) ///
  > prior({displacement}, normal(0,1000)) ///
  > prior({foreign}, normal(0,1000)) ///
  > prior({_cons}, normal(0,1000)) ///
  > prior({var}, igamma(0.001,0.001)) ///
  > rseed(142857)

  Burn-in ... 
  Simulation ... 

Model summary

Likelihood: 
\text{lp100km} \sim \text{normal}(\text{xb\_lp100km},\text{var})

Prior(s):
\{\text{lp100km:weight displacement foreign }_\text{cons}\} \sim \text{normal}(0,1000) \hspace{1cm} (1)
\text{var} \sim \text{igamma}(0.001,0.001)

(1) Parameters are elements of the linear form \text{xb\_lp100km}.

Bayesian normal regression
Random-walk Metropolis-Hastings sampling

<table>
<thead>
<tr>
<th>MCMC iterations</th>
<th>12,500</th>
</tr>
</thead>
<tbody>
<tr>
<td>Burn-in</td>
<td>2,500</td>
</tr>
<tr>
<td>MCMC sample size</td>
<td>10,000</td>
</tr>
<tr>
<td>Number of obs</td>
<td>74</td>
</tr>
<tr>
<td>Acceptance rate</td>
<td>.3087</td>
</tr>
<tr>
<td>Efficiency:</td>
<td>min = .03667</td>
</tr>
<tr>
<td></td>
<td>avg = .04561</td>
</tr>
<tr>
<td>Log marginal likelihood</td>
<td>-164.5299</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Equal-tailed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>\text{lp100km}</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
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
using centering

```
bayesmh lp100km wt1300 displacement3 foreign, ///
    likelihood(normal(var)) ///
    prior(wt1300), normal(0,1000) ///
    prior(displacement3), normal(0,1000) ///
    prior(foreign), normal(0,1000) ///
    prior(_cons), normal(0,1000) ///
    prior(var), igamma(0.001,0.001) ///
    rseed(142857)
```

Burn-in ... Simulation ...

Model summary

```
Likelihood:
    lp100km ~ normal(xb_lp100km,{var})

Priors:
    {lp100km:wt1300 displacement3 foreign _cons} ~ normal(0,1000) (1)
    {var} ~ igamma(0.001,0.001)
```

(1) Parameters are elements of the linear form xb_lp100km.

Bayesian normal regression MCMC iterations = 12,500
Random-walk Metropolis-Hastings sampling Burn-in = 2,500
MCMC sample size = 10,000
Number of obs = 74
Acceptance rate = .2936
Efficiency: min = .0276
            avg = .05888
Log marginal likelihood = -157.72151 max = .1017

<table>
<thead>
<tr>
<th>Equal-tailed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean  Std. Dev.  MCSE Median  [95% Cred. Interval]</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
</tbody>
</table>

. end of do-file

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 distrbution explicitly
- 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

```stata
.do reggibbs

* using centering
.bayesmh lp100km wt1300 displacement3 foreign, ///
> likelihood(normal({var})) ///
> prior({wt1300}, normal(0,1000)) ///
> prior({displacement3}, normal(0,1000)) ///
> prior({foreign}, normal(0,1000)) ///
> prior({_cons}, normal(0,1000)) ///
> prior({var}, igamma(.001,.001)) ///
> block({lp100km:wt1300}, gibbs) ///
> block({lp100km:displacement3}, gibbs) ///
> block({lp100km:foreign}, gibbs) ///
> block({lp100km:_cons}, gibbs) ///
> block({var}, gibbs) ///
> rseed(142857)

Burn-in ...
Simulation ... 

Model summary

------------------------------------------------------------------------------
<table>
<thead>
<tr>
<th>Likelihood:</th>
<th>Prior:</th>
</tr>
</thead>
<tbody>
<tr>
<td>lp100km ~ normal(xb_lpk100km,{var})</td>
<td>{lp100km:wt1300 displacement3 foreign _cons} ~ normal(0,1000) (1)</td>
</tr>
<tr>
<td>{var} ~ igamma(.001,.001)</td>
<td></td>
</tr>
</tbody>
</table>
------------------------------------------------------------------------------

(1) Parameters are elements of the linear form xb_lp100km.

Bayesian normal regression

Gibbs sampling

MCMC iterations = 12,500
Burn-in = 2,500
MCMC sample size = 10,000
Number of obs = 74
Acceptance rate = 1
Efficiency: min = .07728
avg = .2942

Log marginal likelihood = -157.63634

<table>
<thead>
<tr>
<th>Equal-tailed</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>MCSE</th>
<th>Median</th>
<th>[95% Cred. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>------------------</td>
<td>------</td>
<td>-----------</td>
<td>------</td>
<td>--------</td>
<td>---------------------</td>
</tr>
<tr>
<td>lp100km</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>wt1300</td>
<td>7.500904</td>
<td>1.137687</td>
<td>.040773</td>
<td>7.502938</td>
<td>5.242152</td>
</tr>
<tr>
<td>displacement3</td>
<td>.2416701</td>
<td>.2732162</td>
<td>.009828</td>
<td>.238976</td>
<td>-.2873393</td>
</tr>
<tr>
<td>foreign</td>
<td>1.479528</td>
<td>.4995871</td>
<td>.009963</td>
<td>1.473448</td>
<td>.494879</td>
</tr>
<tr>
<td>_cons</td>
<td>10.78787</td>
<td>.2489216</td>
<td>.004643</td>
<td>10.78923</td>
<td>10.2879</td>
</tr>
<tr>
<td>var</td>
<td>2.231845</td>
<td>.3881057</td>
<td>.004403</td>
<td>2.189157</td>
<td>1.596965</td>
</tr>
</tbody>
</table>

end of do-file

- 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

```stata
    . do reggibbs2

    . * using centering
    . bayesmh lp100km wt1300 displacement3 foreign, ///
       likelihood(normal({var})) ///
       prior({wt1300}, normal(0,1000)) ///
       prior({displacement3}, normal(0,1000)) ///
       prior({foreign}, normal(0,1000)) ///
       prior({_cons}, normal(0,1000)) ///
       prior({var}, igamma(.001,.001)) ///
       block({lp100km:wt1300}, gibbs) ///
       block({lp100km:displacement3}, gibbs) ///
       block({lp100km:foreign}, gibbs) ///
       block({lp100km:_cons}, gibbs) ///
       block({var}, gibbs) ///
       mcmcsize(50000) ///
       rseed(142857)

Burn-in ... Simulation ...

Model summary

------------------------------------------------------------------------------------------------------------------
Likelihood:  
lp100km ~ normal(xb_lp100km,{var})

Priors:  
{lp100km:wt1300 displacement3 foreign _cons} ~ normal(0,1000) (1)
{var} ~ igamma(.001,.001)

------------------------------------------------------------------------------------------------------------------

(1) Parameters are elements of the linear form xb_lp100km.

Bayesian normal regression
Gibbs sampling

MCMC iterations = 52,500
Burn-in = 2,500
MCMC sample size = 50,000
Number of obs = 74
Acceptance rate = 1
Efficiency: min = .102
              avg = .3223

Log marginal likelihood = -157.64735

------------------------------------------------------------------------------------------------------------------

                      | Equal-tailed
                      |      Mean  Std. Dev.  MCSE  Median  [95% Cred. Interval]
------------------------------------------------------------------------------------------------------------------
lp100km
wt1300 |  7.504571  1.111813  .015286  7.500416  5.313615  9.693149
displaceme3 | .2415545  .2662422  .003729  .2390393  -.2807387  .7687533
foreign |  1.484437  .484902  .004194  1.484105  .5278635  2.43858
_cons |  10.78488  .2444281  .001997  10.78407  10.30452  11.2654
var |  2.228945  .3880054  .001897  2.185259  1.592776  3.106856

------------------------------------------------------------------------------------------------------------------

end of do-file
3 Conclusion

3.1 Conclusion

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 Bayesian 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 \texttt{ml} command

Conclusion

• We’ve just touched on what can be done
• I hope this has been somewhat informative
Index

B
Bayesian
  analysis, 1–24
  diagnostics, 8–11, 15–18
  estimation, see bayesmh
  postestimation, 6–8
bayesmh, 4–6, 10–13, 19–24
    specifying a custom likelihood, 11–13

C
comparing models, 18, 19

G
Gibbs sampling, 22–24

P
prior distributions, 3, 4

S
simulation, 3