# Bayesian analysis using Stata

Yulia Marchenko

Executive Director of Statistics
StataCorp LP

2016 German Stata Users Group meeting

# Brief overview of Bayesian analysis

Bayesian analysis is a statistical paradigm that answers research questions about unknown parameters using probability statements.

- What is the probability that a person accused of a crime is guilty?
- What is the probability that treatment A is more cost effective than treatment B for a specific health care provider?
- What is the probability that the odds ratio is between 0.3 and 0.5?
- What is the probability that three out of five quiz questions will be answered correctly by students?
- And more.

You may be interested in Bayesian analysis if

- you have some prior information available from previous studies that you would like to incorporate in your analysis. For example, in a study of preterm birthweights, it would be sensible to incorporate the prior information that the probability of a mean birthweight above 15 pounds is negligible. Or,
- your research problem may require you to answer a question: What is the probability that my parameter of interest belongs to a specific range? For example, what is the probability that an odds ratio is between 0.2 and 0.5? Or,
- you want to assign a probability to your research hypothesis. For example, what is the probability that a person accused of a crime is guilty?
- And more.

- Observed data sample $y$ is fixed and model parameters $\boldsymbol{\theta}$ are random.
- $y$ is viewed as a result of a one-time experiment.
- A parameter is summarized by an entire distribution of values instead of one fixed value as in classical frequentist analysis.

- There is some prior (before seeing the data!) knowledge about $\boldsymbol{\theta}$ formulated as a **prior distribution** $p(\boldsymbol{\theta})$.
- After data $y$ are observed, the information about $\boldsymbol{\theta}$ is updated based on the **likelihood** $f(y|\boldsymbol{\theta})$.
- Information is updated by using the Bayes rule to form a **posterior distribution** $p(\boldsymbol{\theta}|y)$:

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

where $p(y)$ is the **marginal distribution** of the data $y$.

- Estimating a posterior distribution $p(\boldsymbol{\theta}|y)$ is at the heart of Bayesian analysis.
- Various summaries of this distribution are used for inference.
- Point estimates: posterior means, modes, medians, percentiles.
- Interval estimates: **credible intervals** (CrI)—(fixed) ranges to which a parameter is known to belong with a pre-specified probability.
- Monte-Carlo standard error (MCSE)—represents precision about posterior mean estimates.

- Hypothesis testing—assign probability to any hypothesis of interest.
- Model comparison: model posterior probabilities, Bayes factors.

Bayesian inference:

- is universal—it is based on the Bayes rule which applies equally to all models;
- incorporates prior information;
- provides the entire posterior distribution of model parameters;
- is exact, in the sense that it is based on the actual posterior distribution rather than on asymptotic normality in contrast with many frequentist estimation procedures; and
- provides straightforward and more intuitive interpretation of the results in terms of probabilities.

- Potential subjectivity in specifying prior information—noninformative priors or sensitivity analysis to various choices of informative priors.
- Computationally demanding—involves intractable integrals that can only be computed using intensive numerical methods such as Markov chain Monte Carlo (MCMC).

## Research problem

- Study of the prevalence of a rare infectious disease in a small city (Hoff 2009).
- A sample of 20 subjects is checked for infection.
- Parameter $\theta$ is the proportion of infected individuals in the city.
- Outcome $y$ is the # of infected individuals in the sample.

## Model

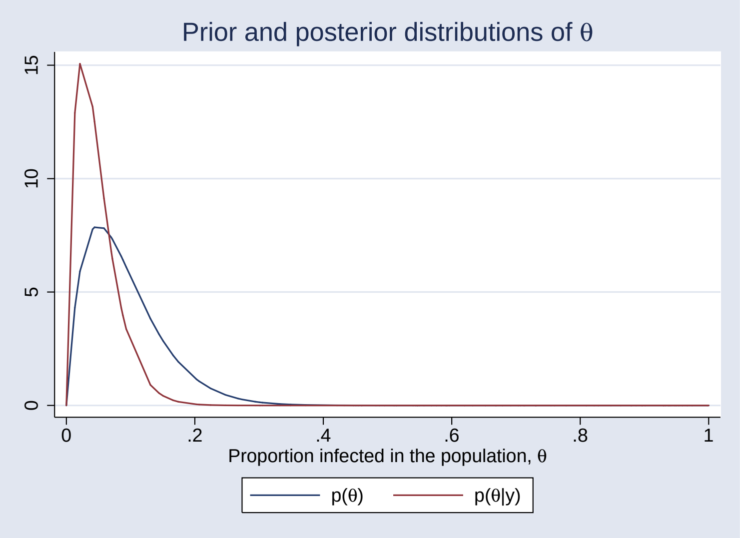
- Likelihood, $f(y|\theta)$: Binomial.
- Prior, $p(\theta)$: Infection rate ranged between 0.05 and 0.20, with an average prevalence of 0.10, in other similar cities.
- Bayesian model:

$$
\begin{aligned}
y|\theta &\sim \mathrm{Binomial}(20, \theta) \\
\theta &\sim \mathrm{Beta}(2, 20)
\end{aligned}
$$

- Posterior: $\theta|y \sim \mathrm{Beta}(2 + y, 20 + 20 - y)$.

## Observed data

- We sample individuals and observe none who have an infection, $y = 0$.
- Posterior: $\theta|y \sim \mathrm{Beta}(2, 40)$.
- Prior mean: $E(\theta) = 2/(2{+}20) = 0.09$.
- Posterior mean: $E(\theta|y) = 2/(2{+}40) = 0.0476$.
- Posterior probability: $P(\theta < 0.10) = 0.926$.


Prior and posterior distributions of θ
15
10
5
0
0
.2
.4
.6
.8
1
Proportion infected in the population, θ
p(θ)
p(θ|y)

- Fit beta-binomial model using `bayesmh`.
- Variable `y` has one observation equal to 0:

```
. set obs 1
number of observations (_N) was 0, now 1
. generate byte y = 0
```

- MCMC method: adaptive Metropolis-Hastings (MH).

```
. set seed 14

. bayesmh y, likelihood(dbinomial({theta},20)) prior({theta}, beta(2,20))

Burn-in ...
Simulation ...

Model summary
------------------------------------------------------------------------------
Likelihood:
  y ~ binomial({theta},20)

Prior:
  {theta} ~ beta(2,20)
------------------------------------------------------------------------------

Bayesian binomial model                          MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
                                                 Number of obs    =          1
                                                 Acceptance rate  =      .4399
Log marginal likelihood = -1.1636733             Efficiency       =      .1625

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. Dev.     MCSE     Median  [95% Cred. Interval]
-------------+----------------------------------------------------------------
       theta |  .0467621    .031854    .00079   .0397556   .0056963   .1282234
------------------------------------------------------------------------------
```

- The estimated posterior mean for $\theta$, 0.047, is close to the theoretical value of 0.0476.

- Compute posterior probability:

```
. bayestest interval {theta}, upper(0.1)
Interval tests     MCMC sample size =    10,000
      prob1 : {theta} < 0.1
```

| | Mean | Std. Dev. | MCSE |
|---|---|---|---|
| prob1 | .9314 | 0.25279 | .0058726 |

- The probability estimate of 0.93 is close to the theoretical value of 0.926.

# Bayesian analysis in Stata

Stata's Bayesian suite consists of the following commands.

| *Command* | *Description* |
|---|---|
| **Estimation** | |
| `bayesmh` | Bayesian regression using MH |
| `bayesmh` *evaluators* | User-written Bayesian models using MH |
| | |
| **Postestimation** | |
| `bayesgraph` | Graphical convergence diagnostics |
| | |
| `bayesstats ess` | Effective sample sizes and more |
| `bayesstats summary` | Summary statistics |
| `bayesstats ic` | Information criteria and Bayes factors |
| | |
| `bayestest model` | Model posterior probabilities |
| `bayestest interval` | Interval hypothesis testing |

- 14 built-in likelihoods: normal, logit, ologit, Poisson, ...
- 18 built-in priors: normal, gamma, Wishart, Zellner's $g$, ...
- Continuous, binary, ordinal, and count outcomes.
- Univariate, multivariate, and multiple-equation models.
- Linear, nonlinear, and canonical generalized linear and nonlinear models.
- Continuous univariate, multivariate, and discrete priors.
- User-defined models: likelihood and priors.

MCMC methods:

- Adaptive MH.
- Adaptive MH with Gibbs updates—hybrid.
- Full Gibbs sampling for some models.

## Built-in models

```
bayesmh ..., likelihood() prior() ...
```

## User-defined models

*Posterior evaluator*

```
bayesmh ..., evaluator() ...
```

*Likelihood evaluator with built-in priors*

```
bayesmh ..., llevaluator() prior() ...
```

Postestimation features are the same whether you use a built-in model or program your own!

- Recall the beta-binomial model from the motivating example.

```
. set seed 14
. bayesmh y, likelihood(dbinomial({theta},20)) prior({theta}, beta(2,20))

Burn-in ...
Simulation ...
Model summary
------------------------------------------------------------------------------
Likelihood:
  y ~ binomial({theta},20)
Prior:
  {theta} ~ beta(2,20)
------------------------------------------------------------------------------

Bayesian binomial model                          MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
                                                 Number of obs    =          1
                                                 Acceptance rate  =      .4399
Log marginal likelihood = -1.1636733             Efficiency       =      .1625

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. Dev.     MCSE     Median  [95% Cred. Interval]
-------------+----------------------------------------------------------------
       theta |  .0467621    .031854    .00079   .0397556   .0056963   .1282234
------------------------------------------------------------------------------
```

- Let's talk about the specification and results in more detail.

- By default, `bayesmh` uses an adaptive random-walk MH method but you can also use Gibbs sampling or a combination of the two algorithms, a hybrid method, for some of the supported likelihood and prior combinations.
- The default burn-in is 2,500 iterations and the default MCMC sample size is 10,000. These numbers are arbitrary and will likely need to be adjusted. Use options `burnin()` and `mcmcsize()` to change the defaults.
- In our beta-binomial example, we used the defaults for the MCMC method, burn-in, and MCMC sample size.

- Let's compute an HPD CrI in our example.
- We specify option `hpd` on replay to recompute CrIs without refitting the model.

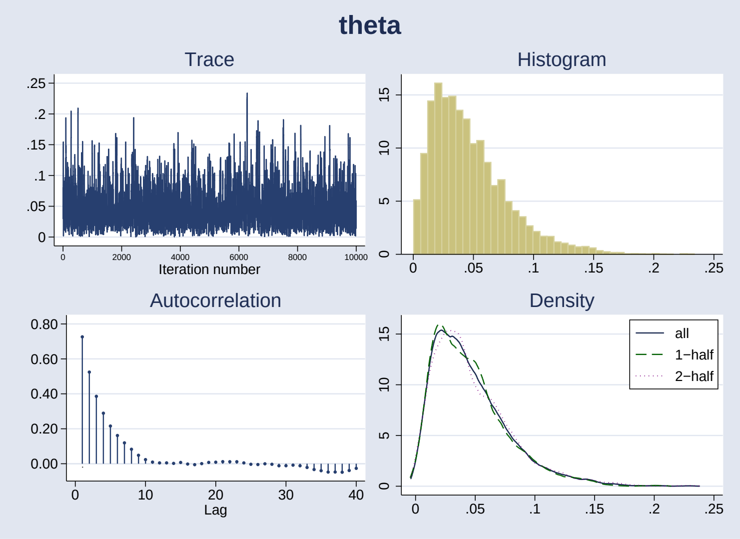
```
. bayesmh, hpd
Model summary
------------------------------------------------------------------------------
Likelihood:
  y ~ binomial({theta},20)
Prior:
  {theta} ~ beta(2,20)
------------------------------------------------------------------------------

Bayesian binomial model                          MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
                                                 Number of obs    =          1
                                                 Acceptance rate  =      .4399
Log marginal likelihood = -1.1636733             Efficiency       =      .1625

------------------------------------------------------------------------------
             |                                                HPD
             |      Mean   Std. Dev.     MCSE     Median  [95% Cred. Interval]
-------------+----------------------------------------------------------------
       theta |  .0467621    .031854    .00079   .0397556   .0009822   .1093087
------------------------------------------------------------------------------
```

- Let's store the estimation results for future comparison.
- `estimates store` stores estimation results but requires first saving `bayesmh`'s MCMC data.
- Use option `saving()` with `bayesmh` during estimation or on replay to save MCMC data in a Stata dataset:

```
. bayesmh, saving(betabin_mcmc)
note: file betabin_mcmc.dta saved
. estimates store betabin
```

- Check MCMC convergence visually:

```
. bayesgraph diagnostics {theta}
```



- Check MCMC sampling efficiency:

```
. bayesstats ess {theta}
Efficiency summaries    MCMC sample size =     10,000
```

| | ESS | Corr. time | Efficiency |
|---|---|---|---|
| theta | 1624.89 | 6.15 | 0.1625 |

- The goal of interval hypothesis testing is to estimate the posterior probability that a parameter lies in a certain interval. For an interval hypothesis $H$: $\theta \in (a, b)$, what is $p(H|y)$?
- A point hypothesis $H$: $\theta = a$ is only applicable to discrete parameters. For continuous parameters, its probability is zero.
- No distinction between the null and alternative hypotheses: if $P\{H_0\colon \theta \in (a, b)\} = p$, then $P\{H_a\colon \theta \notin (a, b)\} = 1 - p$. No need to assume that the null hypothesis is true.
- A conclusion is not an acceptance or rejection of the null hypothesis but an explicit probability statement about the tested hypothesis of interest.

- Test an interval hypothesis $H$: $\theta < 0.1$:

```
. bayestest interval {theta}, upper(0.1)
Interval tests      MCMC sample size =    10,000
       prob1 : {theta} < 0.1
```

| | Mean | Std. Dev. | MCSE |
|---|---|---|---|
| prob1 | .9314 | 0.25279 | .0058726 |

- The estimate of the posterior probability that $\theta$ is less than 0.1 is 0.93 with an MCSE of 0.006.

- Test multiple interval hypotheses in one statement:

```
. bayestest interval ({theta}, upper(0.1)) ({theta}, upper(0.2))

Interval tests     MCMC sample size =    10,000

       prob1 : {theta} < 0.1
       prob2 : {theta} < 0.2
```

| | Mean | Std. Dev. | MCSE |
|---|---|---|---|
| prob1 | .9314 | 0.25279 | .0058726 |
| prob2 | .9988 | 0.03462 | .0008111 |

- Motivating example used a beta prior for $\theta$.
- Sensitivity analysis to the choice of the priors is very important in Bayesian analysis.
- Consider an alternative prior—a power prior.

- Power priors are based on similar historical data $y_0$.
- Idea: raise the likelihood function of the historical data to the power $\alpha_0$, where $0 \leq \alpha_0 \leq 1$.
- $\alpha_0$ quantifies the uncertainty in $y_0$ by controlling the heaviness of the tails of the prior distribution.
- $\alpha_0 = 0$ means no information from the historical data and $\alpha_0 = 1$ means that the historical data have as much weight as the observed data.

- Suppose that in another similar city, a random sample of 15 subjects was selected and 1 subject had a disease.
- Let's consider a competing power prior:

$$p(\theta) \sim \{\mathrm{BinomialPMF}(15, 1, \theta)\}^{\alpha_0}$$

- Let $\alpha_0 = 0.5$.

- `bayesmh` does not have built-in power priors but we can use `prior()`'s suboption `density()` to specify our own prior.

```
. set seed 14
. bayesmh y, likelihood(dbinomial({theta},20))              ///
>   prior({theta}, density(sqrt(binomialp(15,1,{theta})))) ///
>   saving(powerbin_mcmc)

Burn-in ...
Simulation ...
Model summary
------------------------------------------------------------------------------
Likelihood:
  y ~ binomial({theta},20)
Prior:
  {theta} ~ density(sqrt(binomialp(15,1,{theta})))
------------------------------------------------------------------------------
```

```
Bayesian binomial model                         MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling        Burn-in          =      2,500
                                                MCMC sample size =     10,000
                                                Number of obs    =          1
                                                Acceptance rate  =      .3991
Log marginal likelihood = -3.4613334            Efficiency       =      .1196
```

| | Mean | Std. Dev. | MCSE | Median | Equal-tailed [95% Cred. | Interval] |
|---|---|---|---|---|---|---|
| theta | .0503336 | .0393522 | .001138 | .0409455 | .0036139 | .1528106 |

```
file powerbin_mcmc.dta saved
. estimates store powerbin
```

- The posterior mean estimate of $\theta$, 0.05, under this power prior is slightly larger.

- Compute model posterior probabilities to see which model is more likely given the observed data.

```
. bayestest model powerbin betabin
Bayesian model tests
```

| | log(ML) | P(M) | P(M\|y) |
|---|---|---|---|
| powerbin | -3.4613 | 0.5000 | 0.0913 |
| betabin | -1.1637 | 0.5000 | 0.9087 |

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

- The beta-binomial model appears to be more likely given the data than the model using the power prior.
- We can compare any models as long as they have proper posterior distributions and use the same data for fitting.

- Let's compare our models using the Bayes factor:

```
. bayesstats ic powerbin betabin
Bayesian information criteria
```

| | DIC | log(ML) | log(BF) |
|---|---|---|---|
| powerbin | 2.137519 | -3.461333 | . |
| betabin | 1.96168 | -1.163673 | 2.29766 |

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

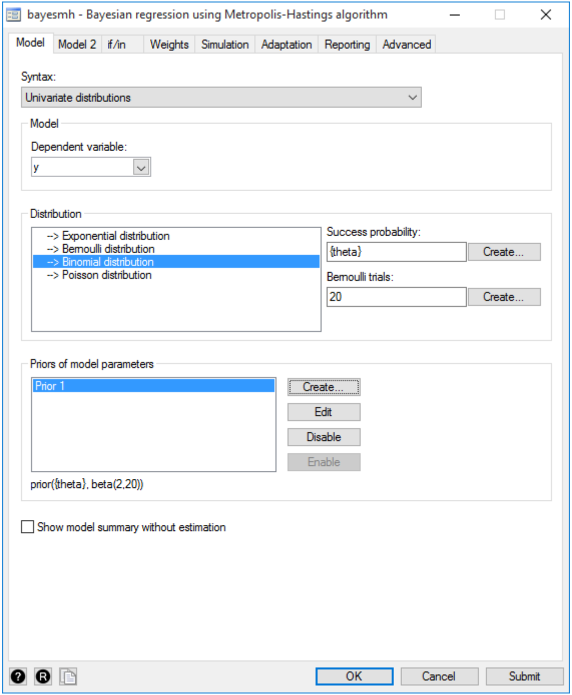
- The natural logarithm of the estimated Bayes factor is 2.3.
- Using the rule of Kass and Raftery (1995), there is some evidence that the beta-binomial model is better because $2 \times 2.3 = 4.6$ is between 2 and 6.

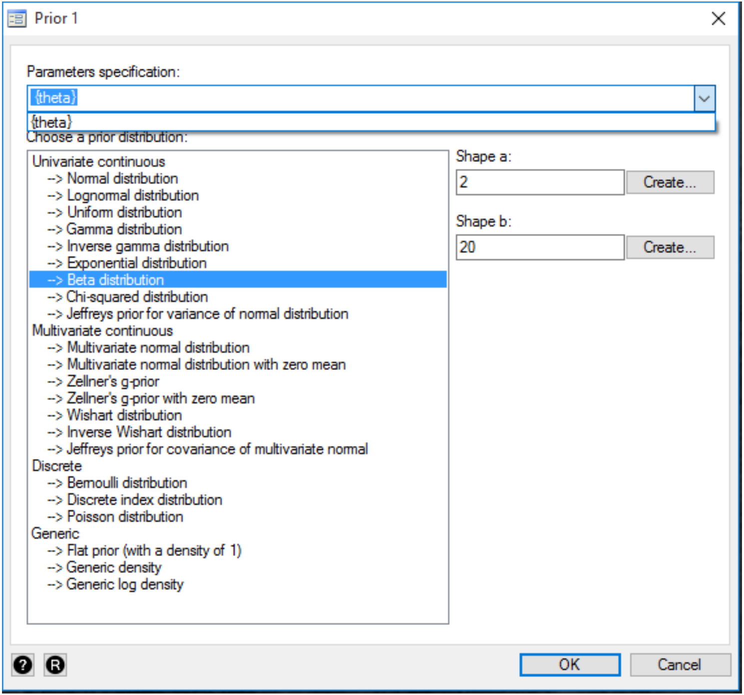
- Perform Bayesian analysis by using the command line.
- Or, use a powerful point-and-click interface.
- You can access the interface by typing:

```
. db bayesmh
```

or from the **Statistics** menu.

(NEXT SLIDE)


bayesmh - Bayesian regression using Metropolis-Hastings algorithm
Model
Model 2
if/in
Weights
Simulation
Adaptation
Reporting
Advanced
Syntax:
Univariate distributions
Model
Dependent variable:
y
Distribution
--> Exponential distribution
--> Bernoulli distribution
--> Binomial distribution
--> Poisson distribution
Success probability:
{theta}
Create...
Bernoulli trials:
20
Create...
Priors of model parameters
Prior 1
Create...
Edit
Disable
Enable
prior({theta}, beta(2,20))
Show model summary without estimation
OK
Cancel
Submit


Prior 1
Parameters specification:
{theta}
{theta}
Choose a prior distribution:
Univariate continuous
--> Normal distribution
--> Lognormal distribution
--> Uniform distribution
--> Gamma distribution
--> Inverse gamma distribution
--> Exponential distribution
--> Beta distribution
--> Chi-squared distribution
--> Jeffreys prior for variance of normal distribution
Multivariate continuous
--> Multivariate normal distribution
--> Multivariate normal distribution with zero mean
--> Zellner's g-prior
--> Zellner's g-prior with zero mean
--> Wishart distribution
--> Inverse Wishart distribution
--> Jeffreys prior for covariance of multivariate normal
Discrete
--> Bernoulli distribution
--> Discrete index distribution
--> Poisson distribution
Generic
--> Flat prior (with a density of 1)
--> Generic density
--> Generic log density
Shape a:
2
Create...
Shape b:
20
Create...
OK
Cancel

# User-written Bayesian models

- One of the questions we received shortly after releasing `bayesmh` is "How do I fit Bayesian hurdle models?"
- A hurdle model (Cragg model) is used to model a bounded dependent variable. It combines a selection model that determines the boundary points of the dependent variable with an outcome model that determines its nonbounded values.
- Hurdle models are not currently among the built-in `bayesmh` models.
- But, we can program them using `bayesmh`'s user-defined evaluators.
- Below I provide two types of log-likelihood evaluators: one using Stata's command `churdle` (new in Stata 14) to compute the log likelihood and the other programming the log likelihood from scratch.

- We consider a subset of the `fitness` data from **[R] churdle**.
- We consider a simple linear hurdle model.
- We model the decision to exercise or not as a function of an individual's average commute to work.
- Once a decision to exercise is made, we model the number of hours an individual exercises per day as a function of age.

- Likelihood model:

$$\begin{aligned}
\texttt{hours}_i &= (\beta_0 + \beta_1 \texttt{age}_i + \nu_i) \times \texttt{hours0}_i \\
\texttt{hours0}_i &= \begin{cases} 1 & \text{if } (\gamma_0 + \gamma_1 \texttt{commute}_i + \epsilon_i) \\ 0 & \text{otherwise} \end{cases} \\
\nu_i &\sim \text{TruncatedNormal}(0, \sigma^2, -\beta_0 - \beta_1 \texttt{age}_i, \infty) \\
\epsilon_i &\sim \text{Normal}(0,1)
\end{aligned}$$

- Prior distributions:

$$\begin{aligned}
\beta_0,\, \beta_1,\, \gamma_0,\, \gamma_1 &\sim 1 \\
\ln(\sigma) &\sim 1
\end{aligned}$$

- Data:

```
. webuse fitness10

. describe

Contains data from fitness10.dta
  obs:         1,983
 vars:             4                          14 Feb 2016 16:27
 size:        19,830
------------------------------------------------------------------------------
              storage   display    value
variable name   type    format     label      variable label
------------------------------------------------------------------------------
age             byte    %9.0g                 person´s age
commute         float   %9.0g                 hours commuted
hours           float   %9.0g                 hours exercised daily
hours0          byte    %8.0g                 (hours==0)
------------------------------------------------------------------------------
Sorted by:
```

- We use `churdle` (line 5 of the program) to compute the log-likelihood values at each MCMC iteration:

```
. program mychurdle1
  1.          version 14.1
  2.          args llf
  3.          tempname b
  4.          mat `b´ = ($MH_b, $MH_p)
  5.          cap churdle linear $MH_y1 $MH_y1x1 if $MH_touse, ///
>                          select($MH_y2x1) ll(0) from(`b´) iterate(0)
  6.          if _rc {
  7.                  if (_rc==1) { // handle break key
  8.                          exit _rc
  9.                  }
 10.                  scalar `llf´ = .
 11.          }
 12.          else {
 13.                  scalar `llf´ = e(ll)
 14.          }
 15. end
```

- Model fitting:

```
. set seed 14

. gen byte hours0 = (hours==0)

. bayesmh (hours age) (hours0 commute),          ///
>   llevaluator(mychurdle1, parameters({lnsig})) ///
>   prior({hours:} {hours0:} {lnsig}, flat) dots

Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaa. done
Simulation 10000 .........1000.........2000.........3000.........4000.........5
> 000.........6000.........7000.........8000.........9000.........10000 done

Model summary
------------------------------------------------------------------------------
Likelihood:
  hours hours0 ~ mychurdle1(xb_hours,xb_hours0,{lnsig})

Priors:
       {hours:age _cons} ~ 1 (flat)                                        (1)
  {hours0:commute _cons} ~ 1 (flat)                                        (2)
                 {lnsig} ~ 1 (flat)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_hours.
(2) Parameters are elements of the linear form xb_hours0.
```

```
Bayesian regression                                 MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling            Burn-in          =      2,500
                                                    MCMC sample size =     10,000
                                                    Number of obs    =      1,983
                                                    Acceptance rate  =      .2752
                                                    Efficiency:  min =     .04197
                                                                 avg =     .06659
Log marginal likelihood = -2772.4136                             max =     .08861
```

| | Mean | Std. Dev. | MCSE | Median | Equal-tailed [95% Cred. | Interval] |
|---|---|---|---|---|---|---|
| **hours** | | | | | | |
| age | .0051872 | .0027702 | .000093 | .0052248 | -.0002073 | .0104675 |
| _cons | 1.163384 | .1219417 | .005135 | 1.16685 | .9203519 | 1.388663 |
| **hours0** | | | | | | |
| commute | -.0716184 | .1496757 | .005623 | -.0758964 | -.3733355 | .2181717 |
| _cons | .1454332 | .084041 | .003066 | .1451574 | -.0222543 | .3128047 |
| lnsig | .1341657 | .034162 | .001668 | .1336526 | .0634106 | .2021694 |

- This model took about 25 minutes to run.

- The corresponding log likelihood programmed from scratch:

```
. program mychurdle2
  1.           version 14.1
  2.           args lnf xb xg lnsig
  3.           tempname sig
  4.           scalar `sig' = exp(`lnsig')
  5.           tempvar lnfj
  6.           qui gen double `lnfj' = normal(`xg')  if $MH_touse
  7.           qui replace `lnfj'    = log(1 - `lnfj') if $MH_y1 <= 0 & $MH_touse
  8.           qui replace `lnfj'    = log(`lnfj') - log(normal(`xb'/`sig')) ///
>                                  + log(normalden($MH_y1,`xb',`sig'))     ///
>                                    if $MH_y1 > 0 & $MH_touse
  9.           summarize `lnfj' if $MH_touse, meanonly
 10.           if r(N) < $MH_n {
 11.               scalar `lnf' = .
 12.               exit
 13.           }
 14.           scalar `lnf' = r(sum)
 15. end
```

- Model fitting:

```
. set seed 14
. bayesmh (hours age) (hours0 commute),                         ///
>         llevaluator(mychurdle2, parameters({lnsig}) )   ///
>         prior({hours:} {hours0:} {lnsig}, flat) dots

Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaa. done
Simulation 10000 .........1000.........2000.........3000.........4000.........5
> 000.........6000.........7000.........8000.........9000.........10000 done
Model summary
------------------------------------------------------------------------------
Likelihood:
  hours hours0 ~ mychurdle2(xb_hours,xb_hours0,{lnsig})
Priors:
       {hours:age _cons} ~ 1 (flat)                                         (1)
  {hours0:commute _cons} ~ 1 (flat)                                         (2)
                 {lnsig} ~ 1 (flat)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_hours.
(2) Parameters are elements of the linear form xb_hours0.
```

```
Bayesian regression                              MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
                                                 Number of obs    =      1,983
                                                 Acceptance rate  =      .2752
                                                 Efficiency:  min =     .04197
                                                              avg =     .06659
Log marginal likelihood = -2772.4136                          max =     .08861
```

| | Mean | Std. Dev. | MCSE | Median | Equal-tailed [95% Cred. | Interval] |
|---|---|---|---|---|---|---|
| hours | | | | | | |
| age | .0051872 | .0027702 | .000093 | .0052248 | -.0002073 | .0104675 |
| _cons | 1.163384 | .1219417 | .005135 | 1.16685 | .9203519 | 1.388663 |
| hours0 | | | | | | |
| commute | -.0716184 | .1496757 | .005623 | -.0758964 | -.3733355 | .2181717 |
| _cons | .1454332 | .084041 | .003066 | .1451574 | -.0222543 | .3128047 |
| lnsig | .1341657 | .034162 | .001668 | .1336526 | .0634106 | .2021694 |

- This model took only 20 seconds!

# Conclusion

- Bayesian analysis is a powerful tool that allows you to incorporate prior information about model parameters into your analysis.
- It provides intuitive and direct interpretations of results by using probability statements about parameters.
- It provides a way to assign an actual probability to any hypothesis of interest.

- Use `bayesmh` for estimation: choose one of the built-in models or program your own.
- Use postestimation features for checking MCMC convergence, estimating functions of model parameters, and performing hypothesis testing and model comparison.
- Work interactively using the command line or use the point-and-click interface.
- Check out the **[BAYES] Bayesian analysis** manual for more examples and details about Bayesian analysis.

New features added to `bayesmh` since Stata 14 shipped.

- Option `reffects()` for more efficient simulation of two-level random-effects models;
- Suboption `reffects` within option `block()` for more efficient simulation of multilevel models;
- More convenient fitting of probability distributions using `dexponential()`, `dbernoulli()`, `dbinomial()`, and `dpoisson()`;
- Option `initrandom`, which is useful for generating multiple chains;
- And more.

Type

```
. update all
```

in Stata 14 to get free access to these new features.

- The Stata blog Bayesian entries:
  - *Bayesian modeling: Beyond Stata's built-in models*
    http://blog.stata.com/2015/05/26/bayesian-modeling-beyond-statas-built-in-models/
  - *Bayesian binary item response theory models using bayesmh*
    http://blog.stata.com/2016/01/18/bayesian-binary-item-response-theory-models-using-bayesmh/
  - *Gelman–Rubin convergence diagnostic using multiple chains*
    http://blog.stata.com/2016/05/26/gelman-rubin-convergence-diagnostic-using-multiple-chains/
    Type

    ```
    . net install grubin, from("http://www.stata.com/users/nbalov")
    ```

    to install a user-written command, `grubin`, that computes the Gelman–Rubin diagnostic using multiple chains.

- The Stata News Bayesian articles:
  - *Bayesian analysis*
    http://www.stata.com/stata-news/news30-1/bayesian-analysis/
  - *Bayesian "ranom-effects" models*
    http://www.stata.com/stata-news/news30-2/bayesian-random-effects/
  - *Bayesian IRT—4PL model*
    http://www.stata.com/stata-news/news31-1/bayesian-irt/
    (forthcoming)

- The Stata YouTube channel:
  - *Introduction to Bayesian analysis, part 1: The basic concepts*
    https://www.youtube.com/watch?v=tHIZMJJT4fY
  - *Introduction to Bayesian analysis, part 2: MCMC and the Metropolis-Hastings algorithm*
    https://www.youtube.com/watch?v=IAAZwh6PSNM
  - *Bayesian analysis in Stata*
    https://www.youtube.com/watch?v=-8StHqIaUeY
  - *Graphical user interface for Bayesian analysis in Stata*
    https://www.youtube.com/watch?v=zno7iU6WHtY

Hoff, P. D. 2009. *A First Course in Bayesian Statistical Methods*. New York: Springer.

Kass, R. E., and A. E. Raftery. 1995. Bayes factors. *Journal of the American Statistical Association* 90: 773–795.