bayes — Bayesian regression models using the bayes prefix

Description

The bayes prefix fits Bayesian regression models. It provides Bayesian support for many likelihood-based estimation commands. The bayes prefix uses default or user-supplied priors for model parameters and estimates parameters using MCMC by drawing simulation samples from the corresponding posterior model. Also see [BAYES] **bayesmh** and [BAYES] **bayesmh evaluators** for fitting more general Bayesian models.

Quick start

Bayesian linear regression of y on x, using default normal priors for the regression coefficients and an inverse-gamma prior for the variance

```
bayes: regress y x
```

Same as above, but use a standard deviation of 10 instead of 100 for the default normal priors and shape of 2 and scale of 1 instead of values of 0.01 for the default inverse-gamma prior

```
bayes, normalprior(10) igammaprior(2 1): regress y x
```

Same as above, but simulate four chains

```
bayes, normalprior(10) igammaprior(2 1) nchains(4): regress y x
```

Bayesian logistic regression of y on x1 and x2, showing model summary without performing estimation

```
bayes, dryrun: logit y x1 x2
```

Same as above, but estimate model parameters and use uniform priors for all regression coefficients

```
bayes, prior({y: x1 x2 _cons}, uniform(-10,10)): logit y x1 x2
```

Same as above, but use a shortcut notation to refer to all regression coefficients

```
bayes, prior({y:}, uniform(-10,10)): logit y x1 x2
```

Same as above, but report odds ratios and use uniform priors for the slopes and a normal prior for the intercept

```
bayes, prior({y: x1 x2}, uniform(-10,10)) ///
    prior({y:_cons}, normal(0,10)) or: logit y x1 x2
```

Report odds ratios for the logit model on replay

```
bayes, or
```

Bayesian ordered logit regression of y on x1 and x2, saving simulation results to simdata.dta and using a random-number seed for reproducibility

```
bayes, saving(simdata) rseed(123): ologit y x1 x2 x3
```

Bayesian multinomial regression of y on x1 and x2, specifying 20,000 MCMC samples, setting length of the burn-in period to 5,000, and requesting that a dot be displayed every 500 simulations

```
bayes, mcmcsize(20000) burnin(5000) dots(500): mlogit y x1 x2
```

Bayesian Poisson regression of y on x1 and x2, putting regression slopes in separate blocks and showing block summary

```
bayes, block({y:x1}) block({y:x2}) blocksummary: poisson y x1 x2
```

Bayesian multivariate regression of y1 and y2 on x1, x2, and x3, using Gibbs sampling and requesting 90% HPD credible interval instead of the default 95% equal-tailed credible interval

```
bayes, gibbs clevel(90) hpd: mvreg y1 y2 = x1 x2 x3
```

Same as above, but use mvreg's option level() instead of bayes's option clevel()

```
bayes, gibbs hpd: mvreg y1 y2 = x1 x2 x3, level(90)
```

Suppress estimates of the covariance matrix from the output

```
bayes, noshow(Sigma, matrix)
```

Bayesian Weibull regression of stset survival-time outcome on x1 and x2, specifying starting values of 1 for {y:x1} and of 2 for {y:x2}

```
bayes, initial({y:x1} 1 {y:x2} 2): streg x1 x2, distribution(weibull)
```

Bayesian panel-data regression of y on x1 and x2 with random intercepts by id, after xtseting id as the panel variable

```
xtset id
bayes: xtreg y x1 x2
```

Bayesian two-level linear regression of y on x1 and x2 with random intercepts by id

```
bayes: mixed y x1 x2 || id:
```

Menu

Statistics > Bayesian analysis > Regression models > *estimation_command*

Syntax

`bayes` [, *bayesopts*] : *estimation_command* [, *estopts*]

estimation_command is a likelihood-based estimation command, and *estopts* are command-specific estimation options; see [BAYES] **Bayesian estimation** for a list of supported commands, and see the command-specific entries for the supported estimation options, *estopts*.

bayesopts	Description
Priors	
* `gibbs`	specify Gibbs sampling; available only with `regress`, `xtreg`, or `mvreg` for certain prior combinations
* `normalprior(#)`	specify standard deviation of default normal priors for regression coefficients and other real scalar parameters; default is `normalprior(100)`
* `igammaprior(# #)`	specify shape and scale of default inverse-gamma prior for variances; default is `igammaprior(0.01 0.01)`
* `iwishartprior(# [...])`	specify degrees of freedom and, optionally, scale matrix of default inverse-Wishart prior for unstructured random-effects covariance
+* `sigma(#)`	specify a fixed scale σ with `qreg`; default is random σ parameter with inverse-gamma prior
`prior(`*priorspec*`)`	prior for model parameters; this option may be repeated
`dryrun`	show model summary without estimation
Simulation	
`nchains(#)`	number of chains; default is to simulate one chain
`mcmcsize(#)`	MCMC sample size; default is `mcmcsize(10000)`
`burnin(#)`	burn-in period; default is `burnin(2500)`
`thinning(#)`	thinning interval; default is `thinning(1)`
`rseed(#)`	random-number seed
`exclude(`*paramref*`)`	specify model parameters to be excluded from the simulation results
`restubs(`*restub1 restub2 ...*`)`	specify stubs for random-effects parameters for all levels; allowed only with multilevel models
Blocking	
* `blocksize(#)`	maximum block size; default is `blocksize(50)`
`block(`*paramref* [, *blockopts*]`)`	specify a block of model parameters; this option may be repeated
`blocksummary`	display block summary
* `noblocking`	do not block parameters by default
Initialization	
`initial(`*initspec*`)`	specify initial values for model parameters with a single chain
`init#(`*initspec*`)`	specify initial values for #th chain; requires `nchains()`
`initall(`*initspec*`)`	specify initial values for all chains; requires `nchains()`
`nomleinitial`	suppress the use of maximum likelihood estimates as starting values
`initrandom`	specify random initial values
`initsummary`	display initial values used for simulation
* `noisily`	display output from the estimation command during initialization

Adaptation	
adaptation(*adaptopts*)	control the adaptive MCMC procedure
scale(#)	initial multiplier for scale factor; default is scale(2.38)
covariance(*cov*)	initial proposal covariance; default is the identity matrix
Reporting	
clevel(#)	set credible interval level; default is clevel(95)
hpd	display HPD credible intervals instead of the default equal-tailed credible intervals
eform_option	display coefficient table in exponentiated form
remargl	compute log marginal-likelihood for random-effects models
batch(#)	specify length of block for batch-means calculations; default is batch(0)
saving(*filename*[, replace])	save simulation results to *filename*.dta
nomodelsummary	suppress model summary
nomesummary	suppress multilevel-structure summary; allowed only with multilevel models
chainsdetail	display detailed simulation summary for each chain
[no]dots	suppress dots or display dots every 100 iterations and iteration numbers every 1,000 iterations; default is command-specific
dots(#[, every(#)])	display dots as simulation is performed
[no]show(*paramref*)	specify model parameters to be excluded from or included in the output
showreffects[(*reref*)]	specify that all or a subset of random-effects parameters be included in the output; allowed only with panel-data and multilevel commands
melabel	display estimation table using the same row labels as *estimation_command*; allowed only with multilevel commands
nogroup	suppress table summarizing groups; allowed only with multilevel models
notable	suppress estimation table
noheader	suppress output header
title(*string*)	display *string* as title above the table of parameter estimates
display_options	control spacing, line width, and base and empty cells
Advanced	
search(*search_options*)	control the search for feasible initial values
corrlag(#)	specify maximum autocorrelation lag; default varies
corrtol(#)	specify autocorrelation tolerance; default is corrtol(0.01)

* Starred options are specific to the bayes prefix; other options are common between bayes and bayesmh.

The full specification of iwishartprior() is iwishartprior(# [*matname*] [, relevel(*levelvar*)]).

Options prior() and block() may be repeated.

priorspec and *paramref* are defined in [BAYES] **bayesmh**.

paramref may contain factor variables; see [U] **11.4.3 Factor variables**.

collect is allowed; see [U] **11.1.10 Prefix commands**.

See [U] **20 Estimation and postestimation commands** for more capabilities of estimation commands.

Options

Priors

`gibbs` specifies that Gibbs sampling be used to simulate model parameters instead of the default adaptive Metropolis–Hastings sampling. This option is allowed only with the `regress`, `xtreg`, and `mvreg` estimation commands. It is available only with certain prior combinations such as normal prior for regression coefficients and an inverse-gamma prior for the variance. Specifying the `gibbs` option is equivalent to specifying `block()`'s `gibbs` suboption for all default blocks of parameters. If you use the `block()` option to define your own blocks of parameters, the `gibbs` option will have no effect on those blocks, and an MH algorithm will be used to update parameters in those blocks unless you also specify `block()`'s `gibbs` suboption.

With panel-data and multilevel linear models, Gibbs sampling is used by default for regression coefficients and variance components, and Metropolis–Hastings sampling is used for random effects. For panel-data linear models, you can specify option `gibbs` to use Gibbs sampling also for random effects.

`normalprior(#)` specifies the standard deviation of the default normal priors. The default is `normalprior(100)`. The normal priors are used for scalar parameters defined on the whole real line; see *Default priors* for details.

`igammaprior(# #)` specifies the shape and scale parameters of the default inverse-gamma priors. The default is `igammaprior(0.01 0.01)`. The inverse-gamma priors are used for positive scalar parameters such as a variance; see *Default priors* for details. Instead of a number #, you can specify a missing value (`.`) to refer to the default value of 0.01.

`iwishartprior(#` [*matname*] [`, relevel(`*levelvar*`)`]`)` specifies the degrees of freedom and, optionally, the scale matrix *matname* of the default inverse-Wishart priors used for unstructured covariances of random effects with multilevel models. The degrees of freedom # is a positive real scalar with the default value of $d+1$, where d is the number of random-effects terms at the level of hierarchy *levelvar*. Instead of a number #, you can specify a missing value (`.`) to refer to the default value. Matrix name *matname* is the name of a positive-definite Stata matrix with the default of $I(d)$, the identity matrix of dimension d. If `relevel(`*levelvar*`)` is omitted, the specified parameters are used for inverse-Wishart priors for all levels with unstructured random-effects covariances. Otherwise, they are used only for the prior for the specified level *levelvar*. See *Default priors* for details.

`sigma(#)` specifies a fixed scale in a Bayesian quantile regression. The scale must be a positive number. This option can be used when the scale is known. By default, the scale is considered a random parameter with an inverse-gamma prior with shape and scale parameters of 0.01.

`prior(`*priorspec*`)` specifies a prior distribution for model parameters. This option may be repeated. A prior may be specified for any of the model parameters, except the random-effects parameters in multilevel models. Model parameters with the same prior specifications are placed in a separate block. Model parameters that are not included in prior specifications are assigned default priors; see *Default priors* for details. Model parameters may be scalars or matrices, but both types may not be combined in one prior statement. If multiple scalar parameters are assigned a single univariate prior, they are considered independent, and the specified prior is used for each parameter. You may assign a multivariate prior of dimension d to d scalar parameters. Also see *Referring to model parameters* in [BAYES] **bayesmh**.

All `prior()` distributions are allowed, but they are not guaranteed to correspond to proper posterior distributions for all likelihood models. You need to think carefully about the model you are building and evaluate its convergence thoroughly; see *Convergence of MCMC* in [BAYES] **bayesmh**.

`dryrun` specifies to show the summary of the model that would be fit without actually fitting the model. This option is recommended for checking specifications of the model before fitting the model. The model summary reports the information about the likelihood model and about priors for all model parameters.

Simulation

`nchains(#)` specifies the number of Markov chains to simulate. You must specify at least two chains. By default, only one chain is produced. Simulating multiple chains is useful for convergence diagnostics and to improve precision of parameter estimates. Four chains are often recommended in the literature, but you can specify more or less depending on your objective. The reported estimation results are based on all chains. You can use `bayesstats summary` with option `sepchains` to see the results for each chain. The reported acceptance rate, efficiencies, and log marginal-likelihood are averaged over all chains. You can use option `chainsdetail` to see these simulation summaries for each chain. Also see *Convergence diagnostics using multiple chains* in [BAYES] **bayesmh** and *Gelman–Rubin convergence diagnostic* in [BAYES] **bayesstats grubin**.

`mcmcsize(#)` specifies the target MCMC sample size. The default MCMC sample size is `mcmcsize(10000)`. The total number of iterations for the MH algorithm equals the sum of the burn-in iterations and the MCMC sample size in the absence of thinning. If thinning is present, the total number of MCMC iterations is computed as `burnin()` $+$ (`mcmcsize()` -1) $\times$ `thinning()` $+1$. Computation time of the MH algorithm is proportional to the total number of iterations. The MCMC sample size determines the precision of posterior summaries, which may be different for different model parameters and will depend on the efficiency of the Markov chain. With multiple chains, `mcmcsize()` applies to each chain. Also see *Burn-in period and MCMC sample size* in [BAYES] **bayesmh**.

`burnin(#)` specifies the number of iterations for the burn-in period of MCMC. The values of parameters simulated during burn-in are used for adaptation purposes only and are not used for estimation. The default is `burnin(2500)`. Typically, burn-in is chosen to be as long as or longer than the adaptation period. The burn-in period may need to be larger for multilevel models because these models introduce high-dimensional random-effects parameters and thus require longer adaptation periods. With multiple chains, `burnin()` applies to each chain. Also see *Burn-in period and MCMC sample size* in [BAYES] **bayesmh** and *Convergence of MCMC* in [BAYES] **bayesmh**.

`thinning(#)` specifies the thinning interval. Only simulated values from every $(1 + k \times \#)$th iteration for $k = 0, 1, 2, \ldots$ are saved in the final MCMC sample; all other simulated values are discarded. The default is `thinning(1)`; that is, all simulation values are saved. Thinning greater than one is typically used for decreasing the autocorrelation of the simulated MCMC sample. With multiple chains, `thinning()` applies to each chain.

`rseed(#)` sets the random-number seed. This option can be used to reproduce results. With one chain, `rseed(#)` is equivalent to typing `set seed` # prior to calling the `bayes` prefix; see [R] **set seed**. With multiple chains, you should use `rseed()` for reproducibility; see *Reproducing results* in [BAYES] **bayesmh**.

`exclude(`*paramref*`)` specifies which model parameters should be excluded from the final MCMC sample. These model parameters will not appear in the estimation table, and postestimation features for these parameters and log marginal-likelihood will not be available. This option is useful for suppressing nuisance model parameters. For example, if you have a factor predictor variable with many levels but you are only interested in the variability of the coefficients associated with its levels, not their actual values, then you may wish to exclude this factor variable from the simulation results. If you simply want to omit some model parameters from the output, see the noshow() option. *paramref* can include individual random-effects parameters.

restubs(*restub1 restub2* ...) specifies the stubs for the names of random-effects parameters. You must specify stubs for all levels—one stub per level. This option overrides the default random-effects stubs. See *Likelihood model* for details about the default names of random-effects parameters.

Blocking

blocksize(#) specifies the maximum block size for the model parameters; default is blocksize(50). This option does not apply to random-effects parameters. Each group of random-effects parameters is placed in one block, regardless of the number of random-effects parameters in that group.

block(*paramref* [, *blockopts*]) specifies a group of model parameters for the blocked MH algorithm. By default, model parameters, except the random-effects parameters, are sampled as independent blocks of 50 parameters or of the size specified in option blocksize(). Regression coefficients from different equations are placed in separate blocks. Auxiliary parameters such as variances and correlations are sampled as individual separate blocks, whereas the cutpoint parameters of the ordinal-outcome regressions are sampled as one separate block. With multilevel models, each group of random-effects parameters is placed in a separate block, and the block() option is not allowed with random-effects parameters. The block() option may be repeated to define multiple blocks. Different types of model parameters, such as scalars and matrices, may not be specified in one block(). Parameters within one block are updated simultaneously, and each block of parameters is updated in the order it is specified; the first specified block is updated first, the second is updated second, and so on. See *Improving efficiency of the MH algorithm—blocking of parameters* in [BAYES] **bayesmh**.

blockopts include gibbs, split, scale(), covariance(), and adaptation().

gibbs specifies to use Gibbs sampling to update parameters in the block. This option is allowed only for hyperparameters and only for specific combinations of prior and hyperprior distributions; see *Gibbs sampling for some likelihood-prior and prior-hyperprior configurations* in [BAYES] **bayesmh**. For more information, see *Gibbs and hybrid MH sampling* in [BAYES] **bayesmh**. gibbs may not be combined with scale(), covariance(), or adaptation().

split specifies that all parameters in a block are treated as separate blocks. This may be useful for levels of factor variables.

scale(#) specifies an initial multiplier for the scale factor corresponding to the specified block. The initial scale factor is computed as $\#/\sqrt{n_p}$ for continuous parameters and as $\#/n_p$ for discrete parameters, where n_p is the number of parameters in the block. The default is scale(2.38). If specified, this option overrides the respective setting from the scale() option specified with the command. scale() may not be combined with gibbs.

covariance(*matname*) specifies a scale matrix *matname* to be used to compute an initial proposal covariance matrix corresponding to the specified block. The initial proposal covariance is computed as *rho*×*Sigma*, where *rho* is a scale factor and *Sigma* = *matname*. By default, *Sigma* is the identity matrix. If specified, this option overrides the respective setting from the covariance() option specified with the command. covariance() may not be combined with gibbs.

`adaptation(tarate())` and `adaptation(tolerance())` specify block-specific TAR and acceptance tolerance. If specified, they override the respective settings from the `adaptation()` option specified with the command. `adaptation()` may not be combined with `gibbs`.

`blocksummary` displays the summary of the specified blocks. This option is useful when `block()` is specified.

`noblocking` requests that no default blocking is applied to model parameters. By default, model parameters are sampled as independent blocks of 50 parameters or of the size specified in option `blocksize()`. For multilevel models, this option has no effect on random-effects parameters; blocking is always applied to them.

Initialization

`initial(`*initspec*`)` specifies initial values for the model parameters to be used in the simulation. With multiple chains, this option is equivalent to specifying option `init1()`. You can specify a parameter name, its initial value, another parameter name, its initial value, and so on. For example, to initialize a scalar parameter `alpha` to 0.5 and a 2x2 matrix `Sigma` to the identity matrix `I(2)`, you can type

```
bayes, initial({alpha} 0.5 {Sigma,m} I(2)) : ...
```

You can also specify a list of parameters using any of the specifications described in *Referring to model parameters* in [BAYES] **bayesmh**. For example, to initialize all regression coefficients from equations `y1` and `y2` to zero, you can type

```
bayes, initial({y1:} {y2:} 0) : ...
```

The general specification of *initspec* is

paramref *initval* [*paramref* *initval* [...]]

where *initval* is a number, a Stata expression that evaluates to a number, or a Stata matrix for initialization of matrix parameters.

Curly braces may be omitted for scalar parameters but must be specified for matrix parameters. Initial values declared using this option override the default initial values or any initial values declared during parameter specification in the `likelihood()` option. See *Initial values* for details.

`init#(`*initspec*`)` specifies initial values for the model parameters for the #th chain. This option requires option `nchains()`. `init1()` overrides the default initial values for the first chain, `init2()` for the second chain, and so on. You specify initial values in `init#()` just like you do in option `initial()`. See *Initial values* for details.

`initall(`*initspec*`)` specifies initial values for the model parameters for all chains. This option requires option `nchains()`. You specify initial values in `initall()` just like you do in option `initial()`. You should avoid specifying fixed initial values in `initall()` because then all chains will use the same initial values. `initall()` is useful to specify random initial values when you define your own priors within `prior()`'s `density()` and `logdensity()` suboptions. See *Initial values* for details.

`nomleinitial` suppresses using maximum likelihood estimates (MLEs), or linear programming estimates for `bayes: qreg`, as starting values for model parameters. With multiple chains, this option and discussion below apply only to the first chain. By default, when no initial values are specified, MLE values from *estimation_command* are used as initial values. For multilevel commands, MLE estimates are used only for regression coefficients. Random effects are assigned zero values, and random-effects variances and covariances are initialized with ones and zeros, respectively. If `nomleinitial` is specified and no initial values are provided, the command uses ones for positive scalar parameters, zeros for other scalar parameters, and identity matrices for matrix parameters. `nomleinitial` may be useful for providing an alternative starting state when checking convergence of MCMC. This option cannot be combined with `initrandom`.

`initrandom` specifies that the model parameters be initialized randomly. Random initial values are generated from the prior distributions of the model parameters. If you want to use fixed initial values for some of the parameters, you can specify them in the `initial()` option or during parameter declarations in the `likelihood()` option. Random initial values are not available for parameters with `flat`, `jeffreys`, `density()`, `logdensity()`, and `jeffreys()` priors; you must provide your own initial values for such parameters. This option cannot be combined with `nomleinitial`. See *Specifying initial values* in [BAYES] **bayesmh** for details.

`initsummary` specifies that the initial values used for simulation be displayed.

`noisily` specifies that the output from the estimation command be shown during initialization. The estimation command is executed once to set up the model and calculate initial values for model parameters.

Adaptation

`adaptation(`*adaptopts*`)` controls adaptation of the MCMC procedure. Adaptation takes place every prespecified number of MCMC iterations and consists of tuning the proposal scale factor and proposal covariance for each block of model parameters. Adaptation is used to improve sampling efficiency. Provided defaults are based on theoretical results and may not be sufficient for all applications. See *Adaptation of the MH algorithm* in [BAYES] **bayesmh** for details about adaptation and its parameters.

adaptopts are any of the following options:

`every(`#`)` specifies that adaptation be attempted every #th iteration. The default is `every(100)`. To determine the adaptation interval, you need to consider the maximum block size specified in your model. The update of a block with k model parameters requires the estimation of a $k \times k$ covariance matrix. If the adaptation interval is not sufficient for estimating the $k(k+1)/2$ elements of this matrix, the adaptation may be insufficient.

`maxiter(`#`)` specifies the maximum number of adaptive iterations. Adaptation includes tuning of the proposal covariance and of the scale factor for each block of model parameters. Once the TAR is achieved within the specified tolerance, the adaptation stops. However, no more than # adaptation steps will be performed. The default is variable and is computed as max{25, `floor(burnin()/adaptation(every()))`}.

`maxiter()` is usually chosen to be no greater than (`mcmcsize()` + `burnin()`)/`adaptation(every())`.

`miniter(`#`)` specifies the minimum number of adaptive iterations to be performed regardless of whether the TAR has been achieved. The default is `miniter(5)`. If the specified `miniter()` is greater than `maxiter()`, then `miniter()` is reset to `maxiter()`. Thus, if you specify `maxiter(0)`, then no adaptation will be performed.

`alpha(`#`)` specifies a parameter controlling the adaptation of the AR. `alpha()` should be in [0, 1]. The default is `alpha(0.75)`.

`beta(`#`)` specifies a parameter controlling the adaptation of the proposal covariance matrix. `beta()` must be in [0,1]. The closer `beta()` is to zero, the less adaptive the proposal covariance. When `beta()` is zero, the same proposal covariance will be used in all MCMC iterations. The default is `beta(0.8)`.

gamma(#) specifies a parameter controlling the adaptation rate of the proposal covariance matrix. gamma() must be in [0,1]. The larger the value of gamma(), the less adaptive the proposal covariance. The default is gamma(0).

tarate(#) specifies the TAR for all blocks of model parameters; this is rarely used. tarate() must be in (0,1). The default AR is 0.234 for blocks containing continuous multiple parameters, 0.44 for blocks with one continuous parameter, and 1/*n_maxlev* for blocks with discrete parameters, where *n_maxlev* is the maximum number of levels for a discrete parameter in the block.

tolerance(#) specifies the tolerance criterion for adaptation based on the TAR. tolerance() should be in (0,1). Adaptation stops whenever the absolute difference between the current AR and TAR is less than tolerance(). The default is tolerance(0.01).

scale(#) specifies an initial multiplier for the scale factor for all blocks. The initial scale factor is computed as $\#/\sqrt{n_p}$ for continuous parameters and $\#/n_p$ for discrete parameters, where n_p is the number of parameters in the block. The default is scale(2.38).

covariance(*cov*) specifies a scale matrix *cov* to be used to compute an initial proposal covariance matrix. The initial proposal covariance is computed as $\rho \times \Sigma$, where ρ is a scale factor and $\Sigma =$ *matname*. By default, Σ is the identity matrix. Partial specification of Σ is also allowed. The rows and columns of *cov* should be named after some or all model parameters. According to some theoretical results, the optimal proposal covariance is the posterior covariance matrix of model parameters, which is usually unknown. This option does not apply to the blocks containing random-effects parameters.

Reporting

clevel(#) specifies the credible level, as a percentage, for equal-tailed and HPD credible intervals. The default is clevel(95) or as set by [BAYES] **set clevel**.

hpd displays the HPD credible intervals instead of the default equal-tailed credible intervals.

eform_option causes the coefficient table to be displayed in exponentiated form; see [R] ***eform_option***. The estimation command determines which *eform_option* is allowed (eform(*string*) and eform are always allowed).

remargl specifies to compute the log marginal-likelihood for panel-data and multilevel models. It is not reported by default for these models. Bayesian panel-data and multilevel models contain many parameters because, in addition to regression coefficients and variance components, they also estimate individual random effects. The computation of the log marginal-likelihood involves the inverse of the determinant of the sample covariance matrix of all parameters and loses its accuracy as the number of parameters grows. For high-dimensional models such as multilevel models, the computation of the log marginal-likelihood can be time consuming, and its accuracy may become unacceptably low. Because it is difficult to access the levels of accuracy of the computation for all panel-data and multilevel models, the log marginal-likelihood is not reported by default. For models containing a small number of random effects, you can use the remargl option to compute and display the log marginal-likelihood.

batch(#) specifies the length of the block for calculating batch means and an MCSE using batch means. The default is batch(0), which means no batch calculations. When batch() is not specified, the MCSE is computed using effective sample sizes instead of batch means. batch() may not be combined with corrlag() or corrtol().

`saving(`*filename*[`, replace`]`)` saves simulation results in *filename*`.dta`. The `replace` option specifies to overwrite *filename*`.dta` if it exists. If the `saving()` option is not specified, the `bayes` prefix saves simulation results in a temporary file for later access by postestimation commands. This temporary file will be overridden every time the `bayes` prefix is run and will also be erased if the current estimation results are cleared. `saving()` may be specified during estimation or on replay.

The saved dataset has the following structure. Variable `_chain` records chain identifiers. Variable `_index` records iteration numbers. The `bayes` prefix saves only states (sets of parameter values) that are different from one iteration to another and the frequency of each state in variable `_frequency`. (Some states may be repeated for discrete parameters.) As such, `_index` may not necessarily contain consecutive integers. Remember to use `_frequency` as a frequency weight if you need to obtain any summaries of this dataset. Values for each parameter are saved in a separate variable in the dataset. Variables containing values of parameters without equation names are named as `eq0_p#`, following the order in which parameters are declared in the `bayes` prefix. Variables containing values of parameters with equation names are named as `eq#_p#`, again following the order in which parameters are defined. Parameters with the same equation names will have the same variable prefix `eq#`. For example,

```
. bayes, saving(mcmc): ...
```

will create a dataset, `mcmc.dta`, with variable names `eq1_p1` for `{y:x1}`, `eq1_p2` for `{y:_cons}`, and `eq0_p1` for `{var}`. Also see macros `e(parnames)` and `e(varnames)` for the correspondence between parameter names and variable names.

In addition, the `bayes` prefix saves variable `_loglikelihood` to contain values of the log likelihood from each iteration and variable `_logposterior` to contain values of the log posterior from each iteration.

`nomodelsummary` suppresses the detailed summary of the specified model. The model summary is reported by default.

`nomesummary` suppresses the summary about the multilevel structure of the model. This summary is reported by default for multilevel commands.

`chainsdetail` specifies that acceptance rates, efficiencies, and log marginal-likelihoods be reported separately for each chain. By default, the header reports these statistics averaged over all chains. This option requires option `nchains()`.

`nodots`, `dots`, and `dots(#)` specify to suppress or display dots during simulation. With multiple chains, these options affect all chains. `dots(#)` displays a dot every # iterations. During the adaptation period, a symbol `a` is displayed instead of a dot. If `dots(..., every(#))` is specified, then an iteration number is displayed every #th iteration instead of a dot or `a`. `dots(, every(#))` is equivalent to `dots(1, every(#))`. `dots` displays dots every 100 iterations and iteration numbers every 1,000 iterations; it is a synonym for `dots(100, every(1000))`. `dots` is the default with multilevel commands, and `nodots` is the default with other commands.

`show(`*paramref*`)` or `noshow(`*paramref*`)` specifies a list of model parameters to be included in the output or excluded from the output, respectively. By default, all model parameters (except random-effects parameters with multilevel models) are displayed. Do not confuse `noshow()` with `exclude()`, which excludes the specified parameters from the MCMC sample. When the `noshow()` option is specified, for computational efficiency, MCMC summaries of the specified parameters are not computed or stored in `e()`. *paramref* can include individual random-effects parameters.

showreffects and showreffects(*reref*) are used with panel-data and multilevel commands and specify that all or a list *reref* of random-effects parameters be included in the output in addition to other model parameters. By default, all random-effects parameters are excluded from the output as if you have specified the noshow() option. This option computes, displays, and stores in e() MCMC summaries for the random-effects parameters.

melabel specifies that the bayes prefix use the same row labels as *estimation_command* in the estimation table. This option is allowed only with multilevel commands. It is useful to match the estimation table output of bayes: *mecmd* with that of *mecmd*. This option implies nomesummary and nomodelsummary.

nogroup suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output header. This option is for use with multilevel commands.

notable suppresses the estimation table from the output. By default, a summary table is displayed containing all model parameters except those listed in the exclude() and noshow() options. Regression model parameters are grouped by equation names. The table includes six columns and reports the following statistics using the MCMC simulation results: posterior mean, posterior standard deviation, MCMC standard error or MCSE, posterior median, and credible intervals.

noheader suppresses the output header either at estimation or upon replay.

title(*string*) specifies an optional title for the command that is displayed above the table of the parameter estimates. The default title is specific to the specified likelihood model.

display_options: vsquish, noemptycells, baselevels, allbaselevels, nofvlabel, fvwrap(#), fvwrapon(*style*), and nolstretch; see [R] **Estimation options**.

Advanced

search(*search_options*) searches for feasible initial values. *search_options* are on, repeat(#), and off.

search(on) is equivalent to search(repeat(500)). This is the default.

search(repeat(*k*)), $k > 0$, specifies the number of random attempts to be made to find a feasible initial-value vector, or initial state. The default is repeat(500). An initial-value vector is feasible if it corresponds to a state with positive posterior probability. If feasible initial values are not found after *k* attempts, an error will be issued. repeat(0) (rarely used) specifies that no random attempts be made to find a feasible starting point. In this case, if the specified initial vector does not correspond to a feasible state, an error will be issued.

search(off) prevents the command from searching for feasible initial values. We do not recommend specifying this option.

corrlag(#) specifies the maximum autocorrelation lag used for calculating effective sample sizes. The default is $\min\{500, \texttt{mcmcsize()}/2\}$. The total autocorrelation is computed as the sum of all lag-k autocorrelation values for k from 0 to either corrlag() or the index at which the autocorrelation becomes less than corrtol() if the latter is less than corrlag(). Options corrlag() and batch() may not be combined.

`corrtol(#)` specifies the autocorrelation tolerance used for calculating effective sample sizes. The default is `corrtol(0.01)`. For a given model parameter, if the absolute value of the lag-k autocorrelation is less than `corrtol()`, then all autocorrelation lags beyond the kth lag are discarded. Options `corrtol()` and `batch()` may not be combined.

Remarks and examples

Remarks and examples are presented under the following headings:

For a general introduction to Bayesian analysis, see [BAYES] **Intro**. For a general introduction to Bayesian estimation using adaptive MH and Gibbs algorithms, see [BAYES] **bayesmh**. See [BAYES] **Bayesian estimation** for a list of supported estimation commands. For a quick overview example of all Bayesian commands, see *Overview example* in [BAYES] **Bayesian commands**.

Using the bayes prefix

The `bayes` prefix provides Bayesian estimation for many likelihood-based regression models. Simply prefix your estimation command with `bayes` to get Bayesian estimates—`bayes:` *estimation_command*; see [BAYES] **Bayesian estimation** for a list of supported commands. Also see [BAYES] **bayesmh** for other Bayesian models.

Similarly to the `bayesmh` command, the `bayes` prefix sets up a Bayesian posterior model, uses MCMC to simulate parameters of this model, and summarizes and reports results. The process of specifying a Bayesian model is similar to that described in *Setting up a posterior model* in [BAYES] **bayesmh**, except the likelihood model is now determined by the specified *estimation_command* and default priors are used for model parameters. The `bayes` prefix and the `bayesmh` command share the same methodology of MCMC simulation and the same summarization and reporting of simulation results; see [BAYES] **bayesmh** for details. In the following sections, we provide information specific to the `bayes` prefix.

Likelihood model

With the bayes prefix, the likelihood component of the Bayesian model is determined by the prefixed estimation command, and all posterior model parameters are defined by the likelihood model. For example, the parameters of the model

```
. bayes: streg age smoking, distribution(lognormal)
```

are the regression coefficients and auxiliary parameters you see when you fit

```
. streg age smoking, distribution(lognormal)
```

All estimation commands have regression coefficients as their model parameters. Some commands have additional parameters such as variances and correlation coefficients.

The bayes prefix typically uses the likelihood parameterization and the naming convention of the estimation command to define model parameters, but there are exceptions. For example, the truncreg command uses the standard deviation parameter {sigma} to parameterize the likelihood, whereas bayes: truncreg uses the variance parameter {sigma2}.

Most model parameters are scalar parameters supported on the whole real line such as regression coefficients, log-transformed positive parameters, and atanh-transformed correlation coefficients. For example, positive scalar parameters are the variance parameters in bayes: regress, bayes: tobit, and bayes: truncreg, and matrix parameters are the covariance matrix {Sigma, matrix} in bayes: mvreg and covariances of random effects in multilevel commands such as bayes: meglm.

The names of model parameters are provided in the model summary displayed by the bayes prefix. Knowing these names is useful when specifying the prior distributions, although the bayes prefix does provide default priors; see *Default priors*. You can use the dryrun option with the bayes prefix to see the names of model parameters prior to the estimation. In general, the names of regression coefficients are formed as {*depvar*:*indepvar*}, where *depvar* is the name of the specified dependent variable and *indepvar* is the name of an independent variable. There are exceptions such as bayes: streg, for which *depvar* is replaced with _t. Variance parameters are named {sigma2}, log-standard-deviation parameters are named {lnsigma}, atanh-transformed correlation parameters are named {athrho}, and the covariance matrix of bayes: mvreg is named {Sigma, matrix} (or {Sigma, m} for short).

For panel-data and multilevel models such as bayes: xtreg and bayes: meglm, in addition to regression coefficients and variance components, the bayes prefix also estimates random-effects parameters. This is different from the corresponding frequentist commands, such as xtreg and meglm, in which random effects are integrated out and thus are not among the final model parameters. (They can be predicted after estimation.) As such, the bayes prefix has its own naming convention for model parameters of multilevel commands. Before moving on to Bayesian analysis of multilevel models, you should be familiar with the syntax of the multilevel commands; see, for example, *Syntax* in [ME] **meglm**.

For panel-data models, the regression coefficients are labeled as usual, {*depvar*:*indepvar*}. Random-effects parameters are labeled as {U[*panelvar*]} (or simply {U}), where *panelvar* is the panel variable. For multinomial logistic models, each outcome can have its own random effect, so the random effects are labeled as {U1[*panelvar*]}, {U2[*panelvar*]}, etc. (or simply {U1}, {U2}, etc.), for each outcome level except the baseline outcome. See command-specific entries for the naming convention of additional parameters such as cutpoints with ordinal models. Also see *Different ways of specifying model parameters* for how to refer to individual random effects during postestimation. For examples, see *Panel-data models*.

For multilevel models, the regression coefficients are labeled as usual, {*depvar*:*indepvar*}. Random-effects parameters are labeled as outlined in tables 1 and 2. You can change the default names by specifying the `restubs()` option. The common syntax of {*rename*} is {*restub#*}, where *restub* is a capital letter, `U` for the level specified first, or a sequence of capital letters that is unique to each random-effects level, and # refers to the group of random effects at that level: 0 for random intercepts, 1 for random coefficients associated with the variable specified first in the random-effects equation, 2 for random coefficients associated with the variable specified second, and so on. The full syntax of {*rename*}, {*fullrename*}, is {*restub*#`[`*levelvar*`]`}, where *levelvar* is the variable identifying the level of hierarchy and is often omitted from the specification for brevity. Random effects at the observation level or crossed effects, specified as `_all: R.`*varname* with multilevel commands, are labeled as `{U0}`, `{V0}`, `{W0}`, and so on. Random effects at nesting levels, or nested effects, are labeled using a sequence of capital letters starting with the letter corresponding to the top level. For example, the multilevel model

```
. bayes: melogit y x1 x2 || id1: x1 x2 || id2: x1 || id3:
```

will have random-effects parameters `{U0}`, `{U1}`, and `{U2}` to represent, respectively, random intercepts, random coefficients for `x1`, and random coefficients for `x2` at the `id1` level; parameters `{UU0}` and `{UU1}` for random intercepts and random coefficients for `x1` at the `id2` level; and random intercepts `{UUU0}` at the `id3` level. See *Multilevel models* for more examples. Also see *Different ways of specifying model parameters* for how to refer to individual random effects during postestimation.

Table 1. Random effects at nesting levels of hierarchy (nested effects)

Hierarchy	Random effects	{*rename*}
lev1	Random intercepts	`{U0}`
	Random coefficients	`{U1}`, `{U2}`, etc.
lev1>*lev2*	Random intercepts	`{UU0}`
	Random coefficients	`{UU1}`, `{UU2}`, etc.
lev1>*lev2*>*lev3*	Random intercepts	`{UUU0}`
	Random coefficients	`{UUU1}`, `{UUU2}`, etc.
...		

Table 2. Random effects at the observation level, `_all` (crossed effects)

Hierarchy	Random effects	{*rename*}
lev1	Random intercepts	`{U0}`
lev2	Random intercepts	`{V0}`
lev3	Random intercepts	`{W0}`
...		

Variance components for independent random effects are labeled as {*rename*`:sigma2`}. In the above example, there are six variance components: `{U0:sigma2}`, `{U1:sigma2}`, `{U2:sigma2}`, `{UU0:sigma2}`, `{UU1:sigma2}`, and `{UUU0:sigma2}`.

Covariance matrices of correlated random effects are labeled as {*restub*`:Sigma,matrix`} (or {*restub*`:Sigma,m`} for short), where *restub* is the letter stub corresponding to the level at which random effects are defined. For example, if we specify an unstructured covariance for the random effects at the `id1` and `id2` levels (with `cov(un)` short for `covariance(unstructured)`)

```
. bayes: melogit y x1 x2 || id1: x1 x2, cov(un) || id2: x1, cov(un) || id3:
```

we will have two covariance matrix parameters, a 3×3 covariance {U:Sigma,m} at the id1 level and a 2×2 covariance {UU:Sigma,m} at the id2 level, and the variance component {UUU0:sigma2} at the id3 level.

For Gaussian multilevel models such as bayes: mixed, the error variance component is labeled as {e.*depvar*:sigma2}.

Also see command-specific entries for the naming convention of additional parameters such as cutpoints with ordinal models or overdispersion parameters with negative binomial models.

Default priors

For convenience, the bayes prefix provides default priors for model parameters. The priors are chosen to be general across models and are fairly uninformative for a typical combination of a likelihood model and dataset. However, the default priors may not always be appropriate. You should always inspect their soundness and, if needed, override the prior specification for some or all model parameters using the prior() option.

All scalar parameters supported on the whole real line, such as regression coefficients and log-transformed positive parameters, are assigned a normal distribution with zero mean and variance σ^2_{prior}, $N(0, \sigma^2_{\text{prior}})$, where σ_{prior} is given by the normalprior() option. The default value for σ_{prior} is 100, and thus the default priors for these parameters are $N(0, 10000)$. These priors are fairly uninformative for parameters of moderate size but may become informative for large-scale parameters. See the *Linear regression: A case of informative default priors* example below.

All positive scalar parameters, such as the variance parameters in bayes: regress and bayes: tobit, are assigned an inverse-gamma prior with shape parameter α and scale parameter β, InvGamma(α, β). The default values for α and β are 0.01, and thus the default prior for these parameters is InvGamma$(0.01, 0.01)$.

All cutpoint parameters of ordinal-outcome models, such as bayes: ologit and bayes: oprobit are assigned flat priors, improper uniform priors with a constant density of 1, equivalent to specifying the flat prior option. The reason for this choice is that the cutpoint parameters are sensitive to the range of the outcome variables, which is usually unknown a priori.

For panel-data models except bayes: xtpoisson and bayes: xtnbreg, the random effects are assigned normal priors with zero mean and variance {var_U}, and {var_U} is assigned an inverse-gamma prior InvGamma$(0.01, 0.01)$. For a Poisson model, the random effects are assigned an exponential gamma prior with a hyperprior parameter {alpha} having an inverse-gamma prior InvGamma$(0.01, 0.01)$. For a negative binomial model, the random effects are assigned a beta prior with hyperparameters {r} and {s}, which are assigned a Pareto-type prior as described in *Methods and formulas* of [BAYES] **bayes: xtnbreg**.

For multilevel models with independent and identity random-effects covariance structures, variances of random effects are assigned inverse-gamma priors, InvGamma$(0.01, 0.01)$. For unstructured random-effects covariances, covariance matrix parameters are assigned fairly uninformative inverse-Wishart priors, InvWishart$(d + 1, I(d))$, where d is the dimension of the random-effects covariance matrix and $I(d)$ is the identity matrix of dimension d. Setting the degrees-of-freedom parameter of the inverse-Wishart prior to $d + 1$ is equivalent to specifying uniform on $(-1, 1)$ distributions for the individual correlation parameters.

The model summary displayed by the bayes prefix describes the chosen default priors, which you can see prior to estimation if you specify bayes's dryrun option. You can use the prior() option repeatedly to override the default prior specifications for some or all model parameters.

Initial values

By default, the bayes prefix uses the ML estimates from the prefixed estimation command as initial values for all scalar model parameters.

For example, the specification

```
. bayes: logit y x
```

will use the ML estimates from

```
. logit y x
```

as default initial values for the regression coefficients.

You can override the default initial values by using the initial() option; see *Specifying initial values* in [BAYES] **bayesmh**.

If the nomleinitial option is specified, instead of using the estimates from the prefixed command, all scalar model parameters are initialized with zeros, except for the variance parameters, which are initialized with ones.

The covariance matrix parameter {Sigma, matrix} of bayes: mvreg is always initialized with the identity matrix.

For panel-data and multilevel models, regression coefficients are initialized using the ML estimates from the corresponding model without random effects, variances of random effects are initialized with ones, covariances of random effects are initialized with zeros, and random effects themselves are initialized with zeros.

With multiple chains, the following default initialization takes place. The first chain is initialized as described above. The subsequent chains use random initial values. In general, random initial values are generated from the prior distributions. For some improper priors such as flat and jeffreys, to avoid extremely large values, random initial values are sampled from a normal distribution with the mean centered at the initial values of the first chain and with standard deviations proportional to the magnitudes of the respective initial estimates.

See *Specifying initial values* in [BAYES] **bayesmh** for more information about default initial values and for how to specify your own.

Command-specific options

Not all command-specific options, that is, options specified with the estimation command, are applicable within the Bayesian framework. One example is the group of maximum-likelihood optimization options such as technique() and gradient. For a list of supported options, refer to the entry specific to each command; see [BAYES] **Bayesian estimation** for a list of commands.

Some of the command-specific reporting options, such as *eform_option* and display options, can be specified either with *estimation_command* or with the bayes prefix. For example, to obtain estimates of odds ratios instead of coefficients after the logit model, you can specify the or option with the command

```
. bayes: logit y x, or
```

or with the bayes prefix

```
. bayes, or: logit y x
```

You can also specify this option on replay with the bayes prefix

```
. bayes: logit y x
. bayes, or
```

Introductory example

We start with a simple linear regression model applied to womenwage.dta, which contains income data for a sample of working women.

```
. use https://www.stata-press.com/data/r19/womenwage
(Wages of women)
```

Suppose we want to regress women's yearly income, represented by the wage variable, on their age, represented by the age variable. We can fit this model using the regress command.

```
. regress wage age

      Source |       SS           df       MS      Number of obs   =       488
-------------+----------------------------------   F(1, 486)       =     43.53
       Model |  3939.49247         1  3939.49247   Prob > F        =    0.0000
    Residual |  43984.4891       486   90.503064   R-squared       =    0.0822
-------------+----------------------------------   Adj R-squared   =    0.0803
       Total |  47923.9816       487   98.406533   Root MSE        =    9.5133

------------------------------------------------------------------------------
        wage | Coefficient  Std. err.      t    P>|t|     [95% conf. interval]
-------------+----------------------------------------------------------------
         age |    .399348   .0605289     6.60   0.000     .2804173    .5182787
       _cons |   6.033077   1.791497     3.37   0.001     2.513041    9.553112
------------------------------------------------------------------------------
```

▷ Example 1: Bayesian simple linear regression

We can fit a corresponding Bayesian regression model by simply adding bayes: in front of the regress command. Because the bayes prefix is simulation based, we set a random-number seed to get reproducible results.

```
. set seed 15
. bayes: regress wage age
Burn-in ...
Simulation ...
Model summary
------------------------------------------------------------------------------
Likelihood:
  wage ~ regress(xb_wage,{sigma2})
Priors:
  {wage:age _cons} ~ normal(0,10000)                                      (1)
          {sigma2} ~ igamma(.01,.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_wage.
Bayesian linear regression                       MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
                                                 Number of obs    =        488
                                                 Acceptance rate  =      .3739
                                                 Efficiency:  min =      .1411
                                                              avg =      .1766
Log marginal-likelihood = -1810.1432                          max =      .2271

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
wage         |
         age |  .4008591   .0595579   .001586   .4005088   .2798807   .5183574
       _cons |  5.969069   1.737247   .043218   5.997571    2.60753   9.396475
-------------+----------------------------------------------------------------
      sigma2 |  90.76252   5.891887   .123626   90.43802   79.71145   102.8558
------------------------------------------------------------------------------
Note: Default priors are used for model parameters.
```

The Bayesian model has two regression coefficient parameters, `{wage:age}` and `{wage:_cons}`, and a positive scalar parameter, `{sigma2}`, representing the variance of the error term. The model summary shows the default priors used for the model parameters: `normal(0, 10000)` for the regression coefficients and `igamma(0.01, 0.01)` for the variance parameter. The default priors are provided for convenience and should be used with caution. These priors are fairly uninformative in this example, but this may not always be the case; see the example in *Linear regression: A case of informative default priors*.

The first two columns of the `bayes` prefix's estimation table report the posterior means and standard deviations of the model parameters. We observe that for the regression coefficients `{wage:age}` and `{wage:_cons}`, the posterior means and standard deviations are very similar to the least-square estimates and their standard errors as reported by the `regress` command. The posterior mean estimate for `{sigma2}`, 90.76, is close to the residual mean squared estimate, 90.50, listed in the ANOVA table of the `regress` command. The estimation table of the `bayes` prefix also reports Monte Carlo standard errors (MCSEs), medians, and equal-tailed credible intervals.

The Bayesian estimates are stochastic in nature and, by default, are based on an MCMC sample of size 10,000. It is important to verify that the MCMC simulation has converged; otherwise, the Bayesian estimates cannot be trusted. The simulation efficiencies reported in the header of the estimation table can serve as useful initial indicators of convergence problems. The minimum efficiency in our example is about 0.14, and the average efficiency is about 0.17. These numbers are typical for the MH sampling algorithm used by bayes and do not indicate convergence problems; see example 1 in [BAYES] **bayesstats grubin** for convergence diagnostics using multiple chains for this example. Also see *Convergence of MCMC* in [BAYES] **bayesmh** for details about convergence diagnostics.

◁

▷ Example 2: Predictions

There are several postestimation commands available after the bayes prefix; see [BAYES] **Bayesian postestimation**. Among them is the bayesstats summary command, which we can use to compute simple predictions. Suppose that we want to predict the expected wage of a 40-year-old woman conditional on the above fitted posterior model. Based on our model, this expected wage corresponds to the linear combination $\{\texttt{wage}:\texttt{\_cons}\} + \{\texttt{wage}:\texttt{age}\} \times 40$. We name this expression wage40 and supply it to the bayesstats summary command.

```
. bayesstats summary (wage40: {wage:_cons} + {wage:age}*40)

Posterior summary statistics                      MCMC sample size =    10,000

      wage40 : {wage:_cons} + {wage:age}*40
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
wage40	22.00343	.81679	.024045	21.99231	20.39435	23.6718

The posterior mean estimate for the expected wage is about 22 with a 95% credible interval between 20.39 and 23.67.

◁

▷ Example 3: Gibbs sampling

The bayes prefix uses adaptive MH as its default sampling algorithm. However, in the special case of linear regression, a more efficient Gibbs sampling is available. We can request Gibbs sampling by specifying the gibbs option.

```
. set seed 15
. bayes, gibbs: regress wage age
Burn-in ...
Simulation ...
Model summary
------------------------------------------------------------------------------
Likelihood:
  wage ~ normal(xb_wage,{sigma2})
Priors:
  {wage:age _cons} ~ normal(0,10000)                                       (1)
          {sigma2} ~ igamma(.01,.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_wage.
Bayesian linear regression                       MCMC iterations  =     12,500
Gibbs sampling                                   Burn-in          =      2,500
                                                 MCMC sample size =     10,000
                                                 Number of obs    =        488
                                                 Acceptance rate  =          1
                                                 Efficiency:  min =          1
                                                              avg =          1
Log marginal-likelihood =  -1810.087                          max =          1
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
wage						
age	.3999669	.0611328	.000611	.4005838	.2787908	.518693
_cons	6.012074	1.804246	.018042	6.000808	2.488816	9.549921
sigma2	90.84221	5.939535	.059395	90.54834	79.8132	103.0164

```
Note: Default priors are used for model parameters.
```

The posterior summary results obtained by Gibbs sampling and MH sampling are very close except for the MCSEs. The Gibbs sampler reports substantially lower MCSEs than the default sampler because of its higher efficiency. In fact, in this example, the Gibbs sampler achieves the highest possible efficiency of 1.

◁

Linear regression: A case of informative default priors

Our example in *Introductory example* used the default priors, which were fairly uninformative for those data and that model. This may not always be true. Consider a linear regression model using the familiar auto.dta. Let us regress the response variable price on the covariate length and factor variable foreign.

```
. use https://www.stata-press.com/data/r19/auto, clear
(1978 automobile data)

. regress price length i.foreign

      Source |       SS           df       MS      Number of obs   =        74
-------------+----------------------------------   F(2, 71)        =     16.35
       Model |  200288930         2   100144465   Prob > F        =    0.0000
    Residual |  434776467        71  6123612.21   R-squared       =    0.3154
-------------+----------------------------------   Adj R-squared   =    0.2961
       Total |  635065396        73  8699525.97   Root MSE        =    2474.6

------------------------------------------------------------------------------
       price | Coefficient  Std. err.      t    P>|t|     [95% conf. interval]
-------------+----------------------------------------------------------------
      length |   90.21239   15.83368     5.70   0.000     58.64092    121.7839
             |
     foreign |
    Foreign  |   2801.143    766.117     3.66   0.000     1273.549    4328.737
       _cons |  -11621.35   3124.436    -3.72   0.000     -17851.3   -5391.401
------------------------------------------------------------------------------
```

▷ Example 4: Default priors

We first fit a Bayesian regression model using the bayes prefix with default priors. Because the range of the outcome variable price is at least an order of magnitude larger than the range of the predictor variables length and foreign, we anticipate that some of the model parameters may have large scale, and longer adaptation may be necessary for the MCMC algorithm to reach optimal sampling for these parameters. We allow for longer adaptation by increasing the burn-in period from the default value of 2,500 to 5,000.

```
. set seed 15

. bayes, burnin(5000): regress price length i.foreign

Burn-in ...
Simulation ...

Model summary
------------------------------------------------------------------------------
Likelihood:
  price ~ regress(xb_price,{sigma2})

Priors:
  {price:length 1.foreign _cons} ~ normal(0,10000)                         (1)
                        {sigma2} ~ igamma(.01,.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_price.
```

```
Bayesian linear regression                       MCMC iterations  =     15,000
Random-walk Metropolis-Hastings sampling         Burn-in          =      5,000
                                                 MCMC sample size =     10,000
                                                 Number of obs    =         74
                                                 Acceptance rate  =      .3272
                                                 Efficiency:  min =     .05887
                                                              avg =      .1093
Log marginal-likelihood = -699.23257                          max =      .1958

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
price        |
      length |  33.03301    1.80186   .060848   33.07952   29.36325   36.41022
             |
     foreign |
    Foreign  |  32.77011   98.97104   4.07922    34.3237  -164.1978   222.0855
       _cons | -8.063175   102.9479   3.34161  -9.110308  -205.9497   196.9341
-------------+----------------------------------------------------------------
      sigma2 |   7538628    1297955   29334.9    7414320    5379756   1.04e+07
------------------------------------------------------------------------------
Note: Default priors are used for model parameters.
```

The posterior mean estimates of the regression coefficients are smaller (in absolute value) than the corresponding estimates from the `regress` command, because the default prior for the coefficients, `normal(0, 10000)`, is informative and has a strong shrinkage effect. For example, the least-square estimate of the constant term from `regress` is about −11,621, and its scale is much larger than the default prior standard deviation of 100. As a result, the default prior shrinks the estimate of the constant toward 0 and, specifically, to −8.06.

You should be aware that the default priors are provided for convenience and are not guaranteed to be uninformative in all cases. They are designed to have little effect on model parameters, the maximum likelihood estimates of which are of moderate size, say, less than 100 in absolute value. For large-scale parameters, as in this example, the default priors can become informative.

◁

▷ Example 5: Flat priors

Continuing with example 4, we can override the default priors using the `prior()` option. We can, for example, apply the completely uninformative `flat` prior, a prior with the density of 1, for the coefficient parameters.

```
. set seed 15
. bayes, prior({price:}, flat) burnin(5000): regress price length i.foreign
Burn-in ...
Simulation ...
Model summary
------------------------------------------------------------------------------
Likelihood:
  price ~ regress(xb_price,{sigma2})
Priors:
  {price:length 1.foreign _cons} ~ 1 (flat)                                (1)
                        {sigma2} ~ igamma(.01,.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_price.
Bayesian linear regression                       MCMC iterations  =     15,000
Random-walk Metropolis-Hastings sampling         Burn-in          =      5,000
                                                 MCMC sample size =     10,000
                                                 Number of obs    =         74
                                                 Acceptance rate  =      .3404
                                                 Efficiency:  min =     .07704
                                                              avg =      .1086
Log marginal-likelihood = -669.62603                          max =      .1898
------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
price        |
      length |  89.51576   16.27187   .586237   89.60969   57.96996   122.7961
             |
     foreign |
    Foreign  |  2795.683   770.6359   26.0589   2787.139   1305.773   4298.785
       _cons | -11478.83   3202.027   113.271  -11504.65  -17845.87  -5244.189
-------------+----------------------------------------------------------------
      sigma2 |   6270294    1089331   25002.1    6147758    4504695    8803268
------------------------------------------------------------------------------
Note: Default priors are used for some model parameters.
```

The posterior mean estimates for the coefficient parameters are now close to the least-square estimates from `regress`. For example, the posterior mean estimate for `{price:_cons}` is about −11,479, whereas the least-square estimate is −11,621.

However, the `flat` priors should be used with caution. Flat priors are improper and may result in improper posterior distributions for which Bayesian inference cannot be carried out. You should thus choose the priors carefully, accounting for the properties of the likelihood model.

◁

▷ Example 6: Zellner's g-prior

A type of prior specific to the normal linear regression model is Zellner's g-prior. We can apply it to our example using the `zellnersg0()` prior. For this prior, we need to specify the dimension of the prior, which is the number of regression coefficients (3), a degree of freedom (50) and the variance parameter of the error term in the regression model, `{sigma2}`; the mean parameter is assumed to be 0 by `zellnersg0()`. See example 9 in [BAYES] **bayesmh** for more details about Zellner's g-prior.

```
. set seed 15
. bayes, prior({price:}, zellnersg0(3, 50, {sigma2})) burnin(5000):
> regress price length i.foreign
Burn-in ...
Simulation ...
Model summary
------------------------------------------------------------------------------
Likelihood:
  price ~ regress(xb_price,{sigma2})
Priors:
  {price:length 1.foreign _cons} ~ zellnersg(3,50,0,{sigma2})             (1)
                       {sigma2} ~ igamma(.01,.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_price.
Bayesian linear regression                       MCMC iterations  =     15,000
Random-walk Metropolis-Hastings sampling         Burn-in          =      5,000
                                                 MCMC sample size =     10,000
                                                 Number of obs    =         74
                                                 Acceptance rate  =      .3019
                                                 Efficiency:  min =     .06402
                                                              avg =       .105
Log marginal-likelihood = -697.84862                          max =      .1944
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
price						
length	87.53039	16.24762	.569888	87.72965	55.5177	119.9915
foreign						
Foreign	2759.267	794.043	31.3829	2793.241	1096.567	4202.283
_cons	-11223.95	3211.553	113.34	-11308.39	-17534.25	-4898.139
sigma2	6845242	1159035	26286.9	6716739	4978729	9521252

```
Note: Default priors are used for some model parameters.
```

We see that using this Zellner's g-prior has little effect on the coefficient parameters, and the simulated posterior mean estimates are close to the least-square estimates from `regress`.

◁

Logistic regression with perfect predictors

Let's revisit the example in *Logistic regression model: A case of nonidentifiable parameters* of [BAYES] **bayesmh**. The example uses heartswitz.dta to model the binary outcome disease, the presence of a heart disease, using the predictor variables restecg, isfbs, age, and male. The dataset is a sample from Switzerland.

```
. use https://www.stata-press.com/data/r19/heartswitz, clear
(Subset of Switzerland heart disease data from UCI Machine Learning Repository)
```

▷ Example 7: Perfect prediction

The logistic regression model for these data is

```
. logit disease restecg isfbs age male
```
(output omitted)

To fit a Bayesian logistic regression, we prefix the logit command with bayes. We also specify the noisily option to show the estimation output of the logit command, which is run by the bayes prefix to set up the model and compute starting values for the parameters.

```
. set seed 15

. bayes, noisily: logit disease restecg isfbs age male

note: restecg != 0 predicts success perfectly;
      restecg omitted and 17 obs not used.

note: isfbs != 0 predicts success perfectly;
      isfbs omitted and 3 obs not used.

note: male != 1 predicts success perfectly;
      male omitted and 2 obs not used.

Iteration 0:  Log likelihood = -4.2386144
Iteration 1:  Log likelihood = -4.2358116
Iteration 2:  Log likelihood = -4.2358076
Iteration 3:  Log likelihood = -4.2358076

Logistic regression                                     Number of obs =     26
                                                        LR chi2(1)    =   0.01
                                                        Prob > chi2   = 0.9403
Log likelihood = -4.2358076                             Pseudo R2     = 0.0007
```

disease	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
restecg	0	(omitted)				
isfbs	0	(omitted)				
age	-.0097846	.1313502	-0.07	0.941	-.2672263	.2476572
male	0	(omitted)				
_cons	3.763893	7.423076	0.51	0.612	-10.78507	18.31285

```
Burn-in ...
Simulation ...

Model summary
------------------------------------------------------------------------------
Likelihood:
  disease ~ logit(xb_disease)

Prior:
  {disease:age _cons} ~ normal(0,10000)                                    (1)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_disease.

Bayesian logistic regression                     MCMC iterations  =     12,500
```

```
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
                                                 Number of obs    =         26
                                                 Acceptance rate  =      .2337
                                                 Efficiency:  min =      .1076
                                                              avg =      .1113
Log marginal-likelihood = -14.795726                          max =       .115

------------------------------------------------------------------------------
             |                                                Equal-tailed
     disease |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
     restecg |  (omitted)
       isfbs |  (omitted)
         age | -.0405907   .1650514   .004868  -.0328198  -.4005246   .2592641
        male |  (omitted)
       _cons |  6.616447   9.516872   .290075   5.491008  -8.852858   28.99392
------------------------------------------------------------------------------
Note: Default priors are used for model parameters.
```

As evident from the output of the `logit` command, the covariates `restecg`, `isfbs`, and `male` are omitted because of perfect prediction. Although these predictors cannot be identified using the likelihood alone, they can be identified, potentially, in a posterior model with an informative prior. The default prior `normal(0, 10000)`, used by the `bayes` prefix for the regression coefficients, is not informative enough to resolve the perfect prediction, and we must override it with a more informative prior.

◁

▷ Example 8: Informative prior

In the example in *Logistic regression model: A case of nonidentifiable parameters* of [BAYES] **bayesmh**, we use information from another similar dataset, `hearthungary.dta`, to come up with informative priors for the regression coefficients. We use the same priors with the `bayes` prefix. We specify the `asis` option with the `logit` command to prevent dropping the perfect predictors from the model. We also specify the `nomleinitial` option to prevent the `bayes` prefix from trying to obtain ML estimates to use as starting values; reliable ML estimates cannot be provided by the `logit` command when the perfect predictors are retained.

```
. set seed 15
. bayes, prior({disease:restecg age}, normal(0,10))
> prior({disease:isfbs male}, normal(1,10))
> prior({disease:_cons}, normal(-4,10)) nomleinitial:
> logit disease restecg isfbs age male, asis
Burn-in ...
Simulation ...
Model summary
------------------------------------------------------------------------------
Likelihood:
  disease ~ logit(xb_disease)
Priors:
  {disease:restecg age} ~ normal(0,10)                                     (1)
   {disease:isfbs male} ~ normal(1,10)                                     (1)
        {disease:_cons} ~ normal(-4,10)                                    (1)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_disease.
```

```
Bayesian logistic regression                     MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
                                                 Number of obs    =         48
                                                 Acceptance rate  =      .2121
                                                 Efficiency:  min =     .01885
                                                              avg =     .04328
Log marginal-likelihood = -11.006071                          max =     .06184

------------------------------------------------------------------------------
             |                                                Equal-tailed
     disease |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
     restecg |  1.965122   2.315475   .115615   1.655961  -2.029873   6.789415
       isfbs |  1.708631   2.726071   .113734   1.607439  -3.306837   7.334592
         age |  .1258811   .0707431   .003621   .1245266  -.0016807   .2719748
        male |  .2671381   2.237349   .162967   .3318061  -4.106425   4.609955
       _cons | -2.441911   2.750613   .110611  -2.538183  -7.596747   3.185172
------------------------------------------------------------------------------
```

For this posterior model with informative priors, we successfully estimate all regression parameters in the logistic regression model.

The informative prior in this example is based on information from an independent dataset, `hearthungary.dta`, which is a sample of observations on the same heart condition and predictor attributes as `heartswitz.dta` but sampled from Hungary's population. Borrowing information from independent datasets to construct informative priors is justified only when the datasets are compatible with the currently analyzed data.

◁

Multinomial logistic regression

Consider the health insurance dataset, sysdsn1.dta, to model the insurance outcome, insure, which takes the values Indemnity, Prepaid, and Uninsure, using the predictor variables age, male, nonwhite, and site. This model is considered in more detail in example 4 in [R] **mlogit**.

```
. use https://www.stata-press.com/data/r19/sysdsn1, clear
(Health insurance data)
```

First, we use the mlogit command to fit the model

```
. mlogit insure age male nonwhite i.site, nolog
Multinomial logistic regression                         Number of obs =    615
                                                        LR chi2(10)   =  42.99
                                                        Prob > chi2   = 0.0000
Log likelihood = -534.36165                             Pseudo R2     = 0.0387

------------------------------------------------------------------------------
      insure | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
Indemnity    |  (base outcome)
-------------+----------------------------------------------------------------
Prepaid      |
         age |   -.011745   .0061946    -1.90   0.058    -.0238862    .0003962
        male |   .5616934   .2027465     2.77   0.006     .1643175    .9590693
    nonwhite |   .9747768   .2363213     4.12   0.000     .5115955    1.437958
             |
        site |
           2 |   .1130359   .2101903     0.54   0.591    -.2989296    .5250013
           3 |  -.5879879   .2279351    -2.58   0.010    -1.034733   -.1412433
             |
       _cons |   .2697127   .3284422     0.82   0.412    -.3740222    .9134476
-------------+----------------------------------------------------------------
Uninsure     |
         age |  -.0077961   .0114418    -0.68   0.496    -.0302217    .0146294
        male |   .4518496   .3674867     1.23   0.219     -.268411     1.17211
    nonwhite |   .2170589   .4256361     0.51   0.610    -.6171725     1.05129
             |
        site |
           2 |  -1.211563   .4705127    -2.57   0.010    -2.133751   -.2893747
           3 |  -.2078123   .3662926    -0.57   0.570    -.9257327     .510108
             |
       _cons |  -1.286943   .5923219    -2.17   0.030    -2.447872   -.1260134
------------------------------------------------------------------------------
```

Next, we use the bayes prefix to perform Bayesian estimation of the same multinomial logistic regression model.

```
. set seed 15
. bayes: mlogit insure age male nonwhite i.site
Burn-in ...
Simulation ...
Model summary
------------------------------------------------------------------------------
Likelihood:
  Prepaid Uninsure ~ mlogit(xb_Prepaid,xb_Uninsure)
Priors:
   {Prepaid:age male nonwhite i.site _cons} ~ normal(0,10000)              (1)
  {Uninsure:age male nonwhite i.site _cons} ~ normal(0,10000)              (2)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_Prepaid.
(2) Parameters are elements of the linear form xb_Uninsure.
Bayesian multinomial logistic regression         MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Base outcome: Indemnity                          Number of obs    =        615
                                                 Acceptance rate  =      .2442
                                                 Efficiency:  min =     .01992
                                                              avg =     .03086
Log marginal-likelihood = -614.49286                          max =     .05659
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
Prepaid						
age	-.0125521	.006247	.000396	-.0125871	-.024602	-.0005809
male	.5462718	.2086422	.012818	.5573004	.1263754	.9271802
nonwhite	.9796293	.2275709	.015746	.9737777	.53642	1.401076
site						
2	.098451	.214039	.012887	.0994476	-.3172914	.5260208
3	-.6043961	.2348319	.011596	-.6072807	-1.045069	-.1323191
_cons	.3183984	.3309283	.021325	.3219128	-.3423583	.956505
Uninsure						
age	-.008377	.0118479	.000581	-.0082922	-.0323571	.0140366
male	.4687524	.3537416	.02376	.4748359	-.2495656	1.147333
nonwhite	.1755361	.42708	.022566	.198253	-.7214481	.938098
site						
2	-1.298562	.4746333	.033628	-1.27997	-2.258622	-.4149035
3	-.2057122	.3533365	.020695	-.2009649	-.904768	.4924401
_cons	-1.305083	.5830491	.02451	-1.296332	-2.463954	-.1758435

```
Note: Default priors are used for model parameters.
```

For this model and these data, the default prior specification of the bayes prefix is fairly uninformative and, as a result, the posterior mean estimates for the parameters are close to the ML estimates obtained with mlogit.

We can report posterior summaries for the relative-risk ratios instead of the regression coefficients. This is equivalent to applying an exponential transformation, $\exp(b)$, to the simulated values of each of the regression coefficients, b, and then summarizing them. We can obtain relative-risk ratio summaries by replaying the bayes command with the rrr option specified. We use the already available simulation results from the last estimation and do not refit the model. We could have also specified the rrr option during the estimation.

```
. bayes, rrr
Model summary
------------------------------------------------------------------------------
Likelihood:
  Prepaid Uninsure ~ mlogit(xb_Prepaid,xb_Uninsure)

Priors:
   {Prepaid:age male nonwhite i.site _cons} ~ normal(0,10000)             (1)
  {Uninsure:age male nonwhite i.site _cons} ~ normal(0,10000)             (2)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_Prepaid.
(2) Parameters are elements of the linear form xb_Uninsure.

Bayesian multinomial logistic regression         MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Base outcome: Indemnity                          Number of obs    =        615
                                                 Acceptance rate  =      .2442
                                                 Efficiency:  min =     .02149
                                                              avg =     .03181
Log marginal-likelihood = -614.49286                          max =     .06007

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |        RRR   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
Prepaid      |
         age |   .9875456   .0061686   .000391   .9874918   .9756982   .9994192
        male |   1.764212   .3634348   .022268   1.745953   1.134708   2.527372
    nonwhite |   2.732931   .6240495   .042568   2.647929   1.709875   4.059566
             |
        site |
          2  |   1.129077   .2450092   .015242   1.104561   .7281185   1.692189
          3  |   .5617084   .1338774    .00665   .5448304   .3516675   .8760614
             |
       _cons |   1.451983   .4904589   .029972   1.379764   .7100938    2.60259
-------------+----------------------------------------------------------------
Uninsure     |
         age |   .9917276   .0117452   .000575    .991742   .9681608   1.014136
        male |   1.699605   .6045513   .040763    1.60775   .7791391   3.149782
    nonwhite |   1.301138   .5448086   .027742   1.219271   .4860479   2.555117
             |
        site |
          2  |   .3045686   .1461615   .009698   .2780457   .1044944   .6604046
          3  |   .8663719   .3155926    .01806   .8179411   .4046357   1.636304
             |
       _cons |   .3203309   .1976203   .008063   .2735332   .0850978   .8387492
------------------------------------------------------------------------------
Note: _cons estimates baseline relative risk for each outcome.
Note: Default priors are used for model parameters.
```

Generalized linear model

Consider the insecticide experiment dataset, beetle.dta, to model the number of beetles killed, r, on the number of subjected beetles, n; the type of beetles, beetle; and the log-dose of insecticide, ldose. More details can be found in example 2 of [R] glm.

```
. use https://www.stata-press.com/data/r19/beetle, clear
```

Consider a generalized linear model with a binomial family and a complementary log–log link function for these data.

```
. glm r i.beetle ldose, family(binomial n) link(cloglog) nolog
Generalized linear models                         Number of obs    =        24
Optimization     : ML                             Residual df      =        20
                                                  Scale parameter =         1
Deviance         =  73.76505595                   (1/df) Deviance =  3.688253
Pearson          =   71.8901173                   (1/df) Pearson  =  3.594506

Variance function: V(u) = u*(1-u/n)               [Binomial]
Link function    : g(u) = ln(-ln(1-u/n))          [Complementary log-log]

                                                  AIC             =   6.74547
Log likelihood   = -76.94564525                   BIC             =  10.20398

------------------------------------------------------------------------------
             |                 OIM
           r | Coefficient  std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
      beetle |
  Red flour  |  -.0910396   .1076132    -0.85   0.398    -.3019576    .1198783
   Mealworm  |  -1.836058   .1307125   -14.05   0.000     -2.09225   -1.579867
             |
       ldose |   19.41558   .9954265    19.50   0.000     17.46458    21.36658
       _cons |  -34.84602    1.79333   -19.43   0.000    -38.36089   -31.33116
------------------------------------------------------------------------------
```

To fit a Bayesian generalized linear model with default priors, we type

```
. set seed 15
. bayes: glm r i.beetle ldose, family(binomial n) link(cloglog)
Burn-in ...
Simulation ...
Model summary
------------------------------------------------------------------------------
Likelihood:
  r ~ glm(xb_r)
Prior:
  {r:i.beetle ldose _cons} ~ normal(0,10000)                              (1)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_r.
Bayesian generalized linear models               MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Family: binomial n                               Number of obs    =         24
Link:   complementary log-log                    Scale parameter  =          1
                                                 Acceptance rate  =      .2003
                                                 Efficiency:  min =     .03414
                                                              avg =     .05094
Log marginal-likelihood =  -102.9776                          max =     .08012
```

r	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
beetle						
Red flour	-.0903569	.106067	.004527	-.093614	-.2964984	.112506
Mealworm	-1.843952	.130297	.004603	-1.848374	-2.091816	-1.594582
ldose	19.52814	.9997765	.054106	19.52709	17.6146	21.6217
_cons	-35.04832	1.800461	.096777	-35.0574	-38.81427	-31.61378

```
Note: Default priors are used for model parameters.
```

The posterior mean estimates of the regression parameters are not that different from the ML estimates obtained with `glm`.

If desired, we can request highest posterior density intervals be reported instead of default equal-tailed credible intervals by specifying the hpd option. We can also change the credible-interval level; for example, to request 90% credible intervals, we specify the clevel(90) option. We also could specify these options during estimation.

```
. bayes, clevel(90) hpd
Model summary
------------------------------------------------------------------------------
Likelihood:
  r ~ glm(xb_r)
Prior:
  {r:i.beetle ldose _cons} ~ normal(0,10000)                               (1)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_r.
Bayesian generalized linear models               MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Family: binomial n                               Number of obs    =         24
Link:   complementary log-log                    Scale parameter  =          1
                                                 Acceptance rate  =      .2003
                                                 Efficiency:  min =     .03414
                                                              avg =     .05094
Log marginal-likelihood =  -102.9776                          max =     .08012
------------------------------------------------------------------------------
             |                                                  HPD
           r |      Mean   Std. dev.     MCSE     Median  [90% cred. interval]
-------------+----------------------------------------------------------------
      beetle |
   Red flour | -.0903569    .106067   .004527  -.093614  -.2444412   .1020305
    Mealworm | -1.843952    .130297   .004603 -1.848374   -2.03979  -1.620806
             |
       ldose |  19.52814   .9997765   .054106  19.52709   17.86148   21.16389
       _cons | -35.04832   1.800461   .096777  -35.0574  -37.96057  -32.00411
------------------------------------------------------------------------------
Note: Default priors are used for model parameters.
```

Truncated Poisson regression

The semiconductor manufacturing dataset, probe.dta, contains observational data of failure rates, failure, of silicon wafers with width, width, and depth, depth, tested at four different probes, probe. A wafer is rejected if more than 10 failures are detected. See example 2 in [R] **tpoisson**.

```
. use https://www.stata-press.com/data/r19/probe, clear
(Silicon wafers)
```

We fit a truncated Poisson regression model with a truncation point of 10. We suppress the constant regression term from the likelihood equation using the noconstant option to retain all four probe levels by including ibn.probe in the list of covariates, which declares probe to be a factor variable with no base level.

```
. tpoisson failures ibn.probe depth width, noconstant ll(10) nolog

Truncated Poisson regression
Limits:  lower =         10                     Number of obs     =         88
         upper =       +inf                     Wald chi2(6)      =   11340.50
Log likelihood = -239.35746                     Prob > chi2       =     0.0000

------------------------------------------------------------------------------
    failures | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
       probe |
          1  |   2.714025   .0752617    36.06   0.000     2.566515    2.861536
          2  |   2.602722   .0692732    37.57   0.000     2.466949    2.738495
          3  |   2.725459   .0721299    37.79   0.000     2.584087    2.866831
          4  |   3.139437   .0377137    83.24   0.000     3.065519    3.213354
             |
       depth |  -.0005034   .0033375    -0.15   0.880    -.0070447     .006038
       width |   .0330225    .015573     2.12   0.034     .0025001     .063545
------------------------------------------------------------------------------
```

▷ Example 9: Default priors

We first apply the bayes prefix with default priors to perform Bayesian estimation of the model. The estimation takes a little longer, so we specify the dots option to see the progress.

```
. set seed 15
. bayes, dots: tpoisson failures ibn.probe depth width, noconstant ll(10)
Burn-in 2500 aaaaaaaaa1000.........2000..... done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done
Model summary
------------------------------------------------------------------------------
Likelihood:
  failures ~ tpoisson(xb_failures)
Prior:
  {failures:i.probe depth width} ~ normal(0,10000)                         (1)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_failures.
Bayesian truncated Poisson regression            MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Limits: Lower =         10                       Number of obs    =         88
        Upper =       +inf                       Acceptance rate  =      .1383
                                                 Efficiency:  min =    .004447
                                                              avg =     .01322
Log marginal-likelihood = -288.22663                          max =     .04082
```

failures	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
probe						
1	2.689072	.0696122	.008596	2.688881	2.557394	2.833737
2	2.581567	.0644141	.00966	2.588534	2.436973	2.701187
3	2.712054	.0695932	.006415	2.717959	2.55837	2.844429
4	3.13308	.0397521	.004592	3.133433	3.055979	3.208954
depth	-.000404	.0033313	.000165	-.000504	-.0067928	.0061168
width	.036127	.0165308	.001821	.0360637	.001239	.067552

```
Note: Default priors are used for model parameters.
Note: There is a high autocorrelation after 500 lags.
```

With the default prior specification, the posterior mean estimates for the regression parameters are similar to the ML estimates obtained with the tpoisson command. However, the bayes prefix issues a high autocorrelation warning note and reports a minimum efficiency of only 0.004. The posterior model with default priors seems to be somewhat challenging for the MH sampler. We could allow for longer burn-in and increase the MCMC sample size to improve the MCMC convergence and increase the estimation precision. Instead, we will provide an alternative prior specification that will increase the model flexibility and improve its fit to the data.

◁

▷ Example 10: Hyperpriors

We now assume that the four probe coefficients, {failures:ibn.probe}, have a normal prior distribution with mean parameter {probe_mean} and a variance of 10,000. It is reasonable to assume that all four probes have positive failure rates and that {probe_mean} is a positive hyperparameter. We decide to assign {probe_mean} a gamma(2, 1) hyperprior, which is a distribution with a positive domain and a mean of 2. We use this prior for the purpose of illustration; this prior is not informative for this model and these data. We initialize {probe_mean} with 1 to give it a starting value compatible with its hyperprior.

```
. set seed 15
. bayes, prior({failures:ibn.probe}, normal({probe_mean}, 10000))
> prior({probe_mean}, gamma(2, 1)) initial({probe_mean} 1) dots:
> tpoisson failures ibn.probe depth width, noconstant ll(10)
Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done
Model summary
------------------------------------------------------------------------------
Likelihood:
  failures ~ tpoisson(xb_failures)
Priors:
      {failures:i.probe} ~ normal({probe_mean},10000)                      (1)
  {failures:depth width} ~ normal(0,10000)                                 (1)
Hyperprior:
  {probe_mean} ~ gamma(2,1)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_failures.
Bayesian truncated Poisson regression            MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Limits: Lower =         10                       Number of obs    =         88
        Upper =       +inf                       Acceptance rate  =       .304
                                                 Efficiency:  min =     .04208
                                                              avg =      .0775
Log marginal-likelihood = -287.91504                          max =       .127
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
failures						
probe						
1	2.703599	.0770656	.003757	2.704613	2.551404	2.848774
2	2.592738	.0711972	.002796	2.594628	2.446274	2.728821
3	2.716223	.0755001	.003549	2.719622	2.568376	2.863064
4	3.137069	.0388127	.001317	3.136773	3.062074	3.211616
depth	-.000461	.0033562	.000109	-.0004457	-.0067607	.0062698
width	.0337508	.0152654	.000532	.0337798	.003008	.0622191
probe_mean	2.051072	1.462867	.041051	1.71286	.2211973	5.809428

```
Note: Default priors are used for some model parameters.
```

The MCMC simulation achieves an average efficiency of about 8% with no indication of convergence problems. The posterior mean estimates for the regression parameters are similar to the ML estimates; moreover, the MCMC standard errors are much lower than those achieved by the previous model with default priors. By introducing the hyperparameter {probe_mean}, we have improved the goodness of fit of the model.

◁

Zero-inflated negative binomial model

In this example, we consider a Bayesian model using zero-inflated negative binomial likelihood. We revisit example 1 in [R] **zinb**, which models the number of fish caught by visitors to a national park. The probability that a particular visitor fished is assumed to depend on the variables `child` and `camper`, which are supplied as covariates to the `inflate()` option of `zinb`.

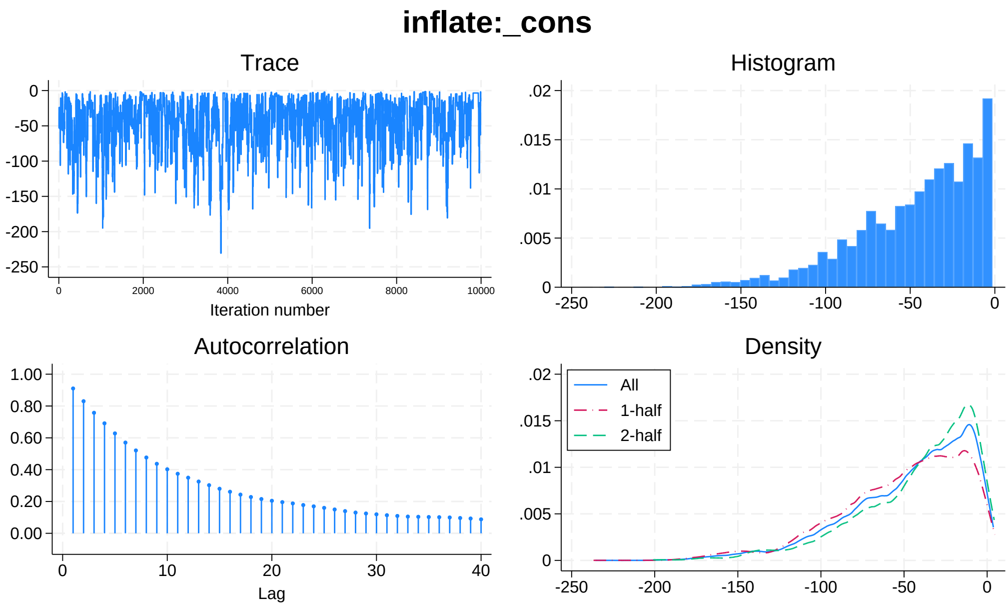
```
. use https://www.stata-press.com/data/r19/fish, clear
(Fictional fishing data)

. zinb count persons livebait, inflate(child camper) nolog

Zero-inflated negative binomial regression              Number of obs =    250
Inflation model: logit                                  Nonzero obs   =    108
                                                        Zero obs      =    142
                                                        LR chi2(2)    =  82.23
Log likelihood = -401.5478                              Prob > chi2   = 0.0000

------------------------------------------------------------------------------
       count | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
count        |
     persons |   .9742984   .1034938     9.41   0.000     .7714543    1.177142
    livebait |   1.557523   .4124424     3.78   0.000     .7491503    2.365895
       _cons |  -2.730064    .476953    -5.72   0.000    -3.664874   -1.795253
-------------+----------------------------------------------------------------
inflate      |
       child |   3.185999   .7468551     4.27   0.000      1.72219    4.649808
      camper |  -2.020951    .872054    -2.32   0.020    -3.730146   -.3117567
       _cons |  -2.695385   .8929071    -3.02   0.003     -4.44545   -.9453189
-------------+----------------------------------------------------------------
    /lnalpha |   .5110429   .1816816     2.81   0.005     .1549535    .8671323
-------------+----------------------------------------------------------------
       alpha |   1.667029   .3028685                      1.167604    2.380076
------------------------------------------------------------------------------
```

Let's fit a Bayesian model with default normal prior distributions.

```
. set seed 15

. bayes, dots: zinb count persons livebait, inflate(child camper)

Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done

Model summary
------------------------------------------------------------------------------
Likelihood:
  count ~ zinb(xb_count,xb_inflate,{lnalpha})

Priors:
  {count:persons livebait _cons} ~ normal(0,10000)                         (1)
    {inflate:child camper _cons} ~ normal(0,10000)                         (2)
                       {lnalpha} ~ normal(0,10000)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_count.
(2) Parameters are elements of the linear form xb_inflate.

Bayesian zero-inflated negative binomial model  MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling        Burn-in          =      2,500
                                                MCMC sample size =     10,000
Inflation model: logit                          Number of obs    =        250
                                                Acceptance rate  =      .3084
                                                Efficiency:  min =     .03716
                                                             avg =      .0791
Log marginal-likelihood = -438.47876                         max =      .1613

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
count        |
     persons |  .9851217   .1084239   .003601    .985452   .7641609   1.203561
    livebait |  1.536074   .4083865   .013509   1.515838    .753823     2.3539
       _cons | -2.805915   .4700702   .014974  -2.795244   -3.73847   -1.89491
-------------+----------------------------------------------------------------
inflate      |
       child |  46.95902   36.33974   1.87977   38.77997   3.612863   138.3652
      camper |   -46.123   36.34857   1.88567  -37.66796  -137.4568  -2.544566
       _cons | -46.62439   36.36232   1.88355   -38.5171  -137.5522  -3.272469
-------------+----------------------------------------------------------------
     lnalpha |  .7055935   .1591234   .003962   .7048862   .3959316   1.025356
------------------------------------------------------------------------------
Note: Default priors are used for model parameters.
```

The posterior mean estimates for the main regression coefficients {count:persons}, {count:livebait}, and {count:_cons} are relatively close to the ML estimates from the zinb command, but the inflation coefficients, {inflate:child}, {inflate:camper}, and {inflate:_cons}, are quite different. For example, zinb estimates {inflate:_cons} are about −2.7, whereas the corresponding posterior mean estimate is about −46.6. To explain this large discrepancy, we draw the diagnostic plot of {inflate:_cons}.

```
. bayesgraph diagnostic {inflate:_cons}
```



The marginal posterior distribution of {inflate:_cons} is highly skewed to the left, and it is apparent that its posterior mean is much smaller than its posterior mode. In large samples, under proper noninformative priors, the posterior mode estimator and the ML estimator are equivalent. Therefore, it is not surprising that the posterior mean of {inflate:_cons} is much smaller than its ML estimate. We can obtain a rough estimate of the posterior mode in this example.

First, we need to save the simulation results in a dataset, say, sim_zinb.dta. You can do this during estimation or on replay by specifying the saving() option with the bayes prefix.

```
. bayes, saving(sim_zinb)
note: file sim_zinb.dta saved.
```

Next, we load the dataset and identify the variable that represents the parameter {inflate:_cons}.

```
. use sim_zinb, clear

. describe

Contains data from sim_zinb.dta
 Observations:         6,874
    Variables:            12                  27 Mar 2025 16:32
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
_chain          int     %8.0g                 Chain identifier
_index          long    %12.0g                Iteration number
_loglikelihood  double  %10.0g                Log likelihood
_logposterior   double  %10.0g                Log posterior
eq1_p1          double  %10.0g                {count:persons}
eq1_p2          double  %10.0g                {count:livebait}
eq1_p3          double  %10.0g                {count:_cons}
eq2_p1          double  %10.0g                {inflate:child}
eq2_p2          double  %10.0g                {inflate:camper}
eq2_p3          double  %10.0g                {inflate:_cons}
eq0_p1          double  %10.0g                {lnalpha}
_frequency      long    %12.0g                Frequency weight
-------------------------------------------------------------------------------
Sorted by:
```

Variable eq2_p3 with the variable label {inflate:_cons} contains MCMC estimates for the {inflate:_cons} parameter.

We use the egen's mode() function to generate a constant variable, mode, which contains the mode estimate for {inflate:_cons}.

```
. egen mode = mode(eq2_p3)

. display mode[1]
-3.417458
```

The mode estimate for {inflate:_cons} is about −3.42, and it is indeed much closer to the ML estimate of −2.70 than its posterior mean estimate.

The inflation parameter α in the likelihood of the zero-inflated negative binomial model is log-transformed, and it is represented by {lnalpha} in our posterior model. To summarize the simulation result for α directly, we can use the bayesstats summary command to exponentiate {lnalpha}.

```
. bayesstats summary (alpha: exp({lnalpha}))

Posterior summary statistics                      MCMC sample size =    10,000

       alpha : exp({lnalpha})

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
       alpha |  2.050889   .3292052   .008191   2.023616   1.485768   2.788087
------------------------------------------------------------------------------
```

Parametric survival model

Consider example 7 in [ST] **streg**, which analyzes the effect of a hip-protection device, age, and sex on the risk of hip fractures in patients. The survival dataset is `hip3.dta` with time to event variable `time1` and failure variable `fracture`. The data are already `stset`.

```
. use https://www.stata-press.com/data/r19/hip3, clear
(Hip-fracture study)

. stset
-> stset time1, id(id) failure(fracture) time0(time0)

Survival-time data settings

           ID variable: id
         Failure event: fracture!=0 & fracture<.
Observed time interval: (time0, time1]
     Exit on or before: failure

--------------------------------------------------------------------------
        206  total observations
          0  exclusions
--------------------------------------------------------------------------
        206  observations remaining, representing
        148  subjects
         37  failures in single-failure-per-subject data
      1,703  total analysis time at risk and under observation
                                                At risk from t =         0
                                     Earliest observed entry t =         0
                                          Last observed exit t =        39
```

It is assumed that the hazard curves for men and women have different shapes. We use the `streg` command to fit a model with Weibull survival distribution and the ancillary variable `male` to account for the difference between men and women.

```
. streg protect age, distribution(weibull) ancillary(male) nolog

        Failure _d: fracture
  Analysis time _t: time1
       ID variable: id

Weibull PH regression

No. of subjects =   148                              Number of obs =    206
No. of failures =    37
Time at risk    = 1,703
                                                     LR chi2(2)    =  39.80
Log likelihood = -69.323532                          Prob > chi2   = 0.0000

------------------------------------------------------------------------------
          _t | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
_t           |
     protect |  -2.130058   .3567005    -5.97   0.000    -2.829178   -1.430938
         age |   .0939131   .0341107     2.75   0.006     .0270573    .1607689
       _cons |  -10.17575   2.551821    -3.99   0.000    -15.17722   -5.174269
-------------+----------------------------------------------------------------
ln_p         |
        male |  -.4887189    .185608    -2.63   0.008    -.8525039   -.1249339
       _cons |   .4540139   .1157915     3.92   0.000     .2270667    .6809611
------------------------------------------------------------------------------
```

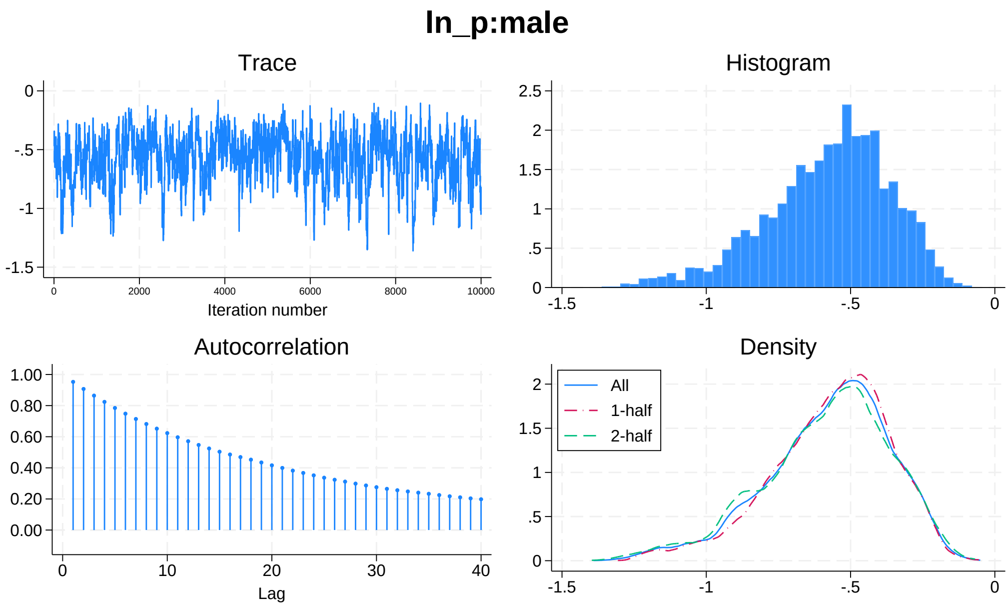
We then perform Bayesian analysis of the same model using the `bayes` prefix. We apply more conservative normal priors, `normal(0, 100)`, by specifying the `normalprior(10)` option. To allow for longer adaptation of the MCMC sampler, we increase the burn-in period to 5,000, `burnin(5000)`.

```
. set seed 15
. bayes, normalprior(10) burnin(5000) dots:
> streg protect age, distribution(weibull) ancillary(male)
        Failure _d: fracture
  Analysis time _t: time1
       ID variable: id
Burn-in 5000 aaaaaaaaa1000aaaaaaaaa2000aaaaaaaaa3000aaaaaaaaa4000aaaaaaaaa5000
> done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done
Model summary
------------------------------------------------------------------------------
Likelihood:
  _t ~ streg_weibull(xb__t,xb_ln_p)
Priors:
  {_t:protect age _cons} ~ normal(0,100)                                   (1)
       {ln_p:male _cons} ~ normal(0,100)                                   (2)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb__t.
(2) Parameters are elements of the linear form xb_ln_p.
Bayesian Weibull PH regression                   MCMC iterations  =     15,000
Random-walk Metropolis-Hastings sampling         Burn-in          =      5,000
                                                 MCMC sample size =     10,000
No. of subjects =        148                     Number of obs    =        206
No. of failures =         37
Time at risk    =       1703
                                                 Acceptance rate  =      .3418
                                                 Efficiency:  min =        .01
                                                              avg =     .03421
Log marginal-likelihood = -91.348814                          max =     .05481
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
_t						
protect	-2.114715	.3486032	.017409	-2.105721	-2.818483	-1.46224
age	.0859305	.0328396	.001403	.0862394	.0210016	.1518009
_cons	-9.57056	2.457818	.117851	-9.551418	-14.49808	-4.78585
ln_p						
male	-.5753907	.2139477	.014224	-.5468488	-1.07102	-.2317242
_cons	.4290642	.11786	.011786	.4242712	.203933	.6548229

The posterior mean estimates for the regression parameters {_t:protect}, {_t:age}, and {_t:_cons} are close to the estimates reported by the streg command. However, the estimate for {ln_p:male} is somewhat different. If we inspect the diagnostic plot for {ln_p:male}, we will see that the reason for this is the asymmetrical shape of its marginal posterior distribution.

```
. bayesgraph diagnostic {ln_p:male}
```



As evident from the density plot, the posterior distribution of {ln_p:male} is skewed to the left, so the posterior mean estimate, −0.58, is expected to be smaller than the ML estimate, −0.49, given that we used fairly uninformative priors; see *Zero-inflated negative binomial model* for the comparison of posterior mean, posterior mode, and ML estimates for highly skewed posterior distributions.

Heckman selection model

▷ Example 11

A representative example of a Heckman selection model is provided by wagenwk.dta, which contains observations on the income of women who choose to work. See example 1 in [R] **heckman**.

```
. use https://www.stata-press.com/data/r19/womenwk, clear
```

The women's income (wage) is assumed to depend on their education (educ) and their age (age). In addition, the selection decision, or the choice of a woman to work, is assumed to depend on their marital status (married), number of children (children), education, and age. We fit this selection model using the heckman command.

```
. heckman wage educ age, select(married children educ age) nolog
Heckman selection model                          Number of obs     =      2,000
(regression model with sample selection)               Selected    =      1,343
                                                       Nonselected =        657

                                                 Wald chi2(2)      =     508.44
Log likelihood = -5178.304                       Prob > chi2       =     0.0000

------------------------------------------------------------------------------
        wage | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
wage         |
   education |   .9899537   .0532565    18.59   0.000     .8855729    1.094334
         age |   .2131294   .0206031    10.34   0.000     .1727481    .2535108
       _cons |   .4857752   1.077037     0.45   0.652    -1.625179     2.59673
-------------+----------------------------------------------------------------
select       |
     married |   .4451721   .0673954     6.61   0.000     .3130794    .5772647
    children |   .4387068   .0277828    15.79   0.000     .3842534    .4931601
   education |   .0557318   .0107349     5.19   0.000     .0346917    .0767718
         age |   .0365098   .0041533     8.79   0.000     .0283694    .0446502
       _cons |  -2.491015   .1893402   -13.16   0.000    -2.862115   -2.119915
-------------+----------------------------------------------------------------
     /athrho |   .8742086   .1014225     8.62   0.000     .6754241    1.072993
    /lnsigma |   1.792559    .027598    64.95   0.000     1.738468     1.84665
-------------+----------------------------------------------------------------
         rho |   .7035061   .0512264                      .5885365    .7905862
       sigma |   6.004797   .1657202                       5.68862    6.338548
      lambda |   4.224412   .3992265                      3.441942    5.006881
------------------------------------------------------------------------------
LR test of indep. eqns. (rho = 0): chi2(1) = 61.20        Prob > chi2 = 0.0000
```

We then apply the bayes prefix to perform Bayesian estimation of the Heckman selection model.

```
. set seed 15

. bayes, dots: heckman wage educ age, select(married children educ age)

Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done

Model summary
------------------------------------------------------------------------------
Likelihood:
  wage ~ heckman(xb_wage,xb_select,{athrho} {lnsigma})

Priors:
                   {wage:education age _cons} ~ normal(0,10000)          (1)
  {select:married children education age _cons} ~ normal(0,10000)       (2)
                             {athrho lnsigma} ~ normal(0,10000)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_wage.
(2) Parameters are elements of the linear form xb_select.

Bayesian Heckman selection model                 MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
                                                 Number of obs    =      2,000
                                                       Selected   =      1,343
                                                       Nonselected =       657
                                                 Acceptance rate  =      .3484
                                                 Efficiency:  min =     .02314
                                                              avg =     .03657
Log marginal-likelihood = -5260.2024                          max =     .05013
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
wage						
education	.9919131	.051865	.002609	.9931531	.8884407	1.090137
age	.2131372	.0209631	.001071	.2132548	.1720535	.2550835
_cons	.4696264	1.089225	.0716	.4406188	-1.612032	2.65116
select						
married	.4461775	.0681721	.003045	.4456493	.3178532	.5785857
children	.4401305	.0255465	.001156	.4402145	.3911135	.4903804
education	.0559983	.0104231	.000484	.0556755	.0360289	.076662
age	.0364752	.0042497	.000248	.0362858	.0280584	.0449843
_cons	-2.494424	.18976	.011327	-2.498414	-2.861266	-2.114334
athrho	.868392	.099374	.005961	.8699977	.6785641	1.062718
lnsigma	1.793428	.0269513	.001457	1.793226	1.740569	1.846779

Note: Default priors are used for model parameters.

The posterior mean estimates for the Bayesian model with default normal priors are similar to the ML estimates obtained with the heckman command.

We can calculate posterior summaries for the correlation parameter, ρ, and the standard error, σ, in their natural scale by inverse-transforming the model parameters {athrho} and {lnsigma} using the bayesstats summary command. We also include posterior summaries for the selectivity effect $\lambda = \rho\sigma$.

```
. bayesstats summary (rho:1-2/(exp(2*{athrho})+1))
> (sigma:exp({lnsigma}))
> (lambda:exp({lnsigma})*(1-2/(exp(2*{athrho})+1)))
Posterior summary statistics                      MCMC sample size =     10,000

         rho : 1-2/(exp(2*{athrho})+1)
       sigma : exp({lnsigma})
      lambda : exp({lnsigma})*(1-2/(exp(2*{athrho})+1))
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
rho	.6970522	.0510145	.003071	.701373	.5905851	.7867018
sigma	6.012205	.1621422	.008761	6.008807	5.700587	6.339366
lambda	4.196646	.3937209	.024351	4.212609	3.411479	4.946325

Again, the posterior mean estimates of ρ, σ, and λ agree with the ML estimates reported by heckman.

◁

Multilevel models

The bayes prefix supports several multilevel commands such as mixed and meglm; see [BAYES] **Bayesian estimation**. Multilevel models introduce effects at different levels of hierarchy such as hospital effects and doctor-nested-within-hospital effects, which are often high-dimensional. These effects are commonly referred to as random effects in frequentist models. Bayesian multilevel models estimate random effects together with other model parameters. In contrast, frequentist multilevel models integrate random effects out, but provide ways to predict them after estimation, conditional on other estimated model parameters. Thus, in addition to regression coefficients and variance components (variances and covariances of random effects), Bayesian multilevel models include random effects themselves as model parameters. With a slight abuse of the terminology, we will sometimes refer to regression coefficients as fixed effects, keeping in mind that they are still random quantities from a Bayesian perspective.

Multilevel models are more difficult to simulate from because of the existence of high-dimensional random-effects parameters. They typically require longer burn-in periods to achieve convergence and larger MCMC sample sizes to obtain precise estimates of random effects and variance components.

Prior specification is particularly important for multilevel models. Using noninformative priors for all model parameters will likely result in nonconvergence or high autocorrelation of the MCMC sample, especially with small datasets. The default priors provided by the bayes prefix are chosen to be fairly uninformative, which may often lead to low simulation efficiencies for model parameters and, especially, for variance components; see *Default priors*. So, do not be surprised to see high autocorrelation with default priors, and be prepared to investigate various prior specifications during your analysis. For example, you may need to use the iwishartprior() option to increase the degrees of freedom and to specify a different scale matrix of the inverse-Wishart prior distribution used for the covariance matrices of random effects.

To change the default priors, you will need to know the names of the model parameters. See *Likelihood model* to learn how the bayes prefix labels the parameters. You can specify your own name stubs for the groups of random-effects parameters using the restubs() option. After simulation, see *Different ways of specifying model parameters* for how to refer to individual random effects to evaluate MCMC convergence or to obtain their MCMC summaries.

By default, the bayes prefix does not compute or display MCMC summaries of individual random effects to conserve computation time and space. You can specify the showreffects() or show() option to compute and display them for chosen groups of random effects.

Also, the bayes prefix does not compute the log marginal-likelihood by default for multilevel models. The computation involves the inverse of the determinant of the sample covariance matrix of all parameters and loses accuracy as the number of parameters grows. For high-dimensional models such as multilevel models, the computation can be time consuming, and its accuracy may become unacceptably low. Because it is difficult to access the levels of accuracy of the computation for all multilevel models, the log marginal-likelihood is not computed by default. For multilevel models containing a small number of random effects, you can use the remargl option to compute and display it.

Assessing convergence of MCMC for multilevel models is challenging because of the high dimensionality. Technically, the convergence of all parameters, including the random-effects parameters, must be explored. In practice, this may not always be feasible. Many applications focus on the regression coefficients and variance components and treat random-effects parameters as nuisance. In this case, it may be sufficient to check convergence only for the parameters of interest, especially because their convergence is adversely affected whenever there are convergence problems for many of the random-effects parameters. If the random-effects parameters are of primary interest in your study, you should evaluate their convergence. For models with a small to moderate number of random-effects parameters, it may be beneficial to always check the convergence of the random-effects parameters. Also see *Convergence of MCMC* in [BAYES] **bayesmh**.

Two-level models

Consider example 1 from [ME] **mixed** that analyzed the weight gain of 48 pigs over 9 successive weeks. Detailed Bayesian analysis of these data using bayesmh are presented in *Panel-data and multilevel models* in [BAYES] **bayesmh**. Here, we use bayes: mixed to fit Bayesian two-level random-intercept and random-coefficient models to these data.

```
. use https://www.stata-press.com/data/r19/pig
(Longitudinal analysis of pig weights)
```

▷ Example 12: Random-intercept model, using option melabel

We first consider a simple random-intercept model of dependent variable `weight` on covariate `week` with variable `id` identifying pigs. The random-intercept model assumes that all pigs share a common growth rate but have different initial weight.

For comparison purposes, we first use the `mixed` command to fit this model by maximum likelihood.

```
. mixed weight week || id:
Performing EM optimization ...
Performing gradient-based optimization:
Iteration 0:  Log likelihood = -1014.9268
Iteration 1:  Log likelihood = -1014.9268
Computing standard errors ...
Mixed-effects ML regression                     Number of obs     =        432
Group variable: id                              Number of groups  =         48
                                                Obs per group:
                                                              min =          9
                                                              avg =        9.0
                                                              max =          9
                                                Wald chi2(1)      =   25337.49
Log likelihood = -1014.9268                     Prob > chi2       =     0.0000

------------------------------------------------------------------------------
      weight | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
        week |   6.209896   .0390124   159.18   0.000     6.133433    6.286359
       _cons |   19.35561   .5974059    32.40   0.000     18.18472    20.52651
------------------------------------------------------------------------------

------------------------------------------------------------------------------
  Random-effects parameters  |   Estimate   Std. err.     [95% conf. interval]
-----------------------------+------------------------------------------------
id: Identity                 |
                  var(_cons) |   14.81751   3.124225      9.801716    22.40002
-----------------------------+------------------------------------------------
               var(Residual) |   4.383264   .3163348      3.805112     5.04926
------------------------------------------------------------------------------
LR test vs. linear model: chibar2(01) = 472.65        Prob >= chibar2 = 0.0000
```

To fit a Bayesian analog of this model, we simply prefix the mixed command with bayes. We also specify the melabel option with bayes to label model parameters in the output table as mixed does.

```
. set seed 15

. bayes, melabel: mixed weight week || id:
note: Gibbs sampling is used for regression coefficients and variance
      components.

Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done

Bayesian multilevel regression                   MCMC iterations  =     12,500
Metropolis-Hastings and Gibbs sampling           Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Group variable: id                               Number of groups =         48
                                                 Obs per group:
                                                              min =          9
                                                              avg =        9.0
                                                              max =          9
                                                 Number of obs    =        432
                                                 Acceptance rate  =      .8112
                                                 Efficiency:  min =    .007005
                                                              avg =      .5064
Log marginal-likelihood                                       max =          1
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
weight						
week	6.209734	.0390718	.000391	6.209354	6.133233	6.285611
_cons	19.46511	.6239712	.07455	19.48275	18.2534	20.67396
id						
var(_cons)	15.7247	3.436893	.049048	15.26104	10.31182	23.60471
var(Residual)	4.411155	.3193582	.004397	4.396044	3.834341	5.080979

```
Note: Default priors are used for model parameters.
```

The estimates of posterior means and posterior standard deviations are similar to the ML estimates and standard errors from mixed. The results are also close to those from bayesmh in example 23 in [BAYES] **bayesmh**.

The average efficiency of the simulation is about 51% and there is no indication of any immediate convergence problems, but we should investigate convergence more thoroughly; see, for example, example 5 in [BAYES] **Bayesian commands** and, more generally, *Convergence of MCMC* in [BAYES] **bayesmh**.

Because Bayesian multilevel models are generally slower than other commands, the bayes prefix displays dots by default with multilevel commands. You can specify the nodots option to suppress them.

Also, as we described in *Multilevel models*, the log marginal-likelihood is not computed for multilevel models by default because of the high dimensionality of the models. This is also described in the help file that appears when you click on Log marginal-likelihood in the output header in the Results window. For models with a small number of random effects, you can specify the remargl option to compute the log marginal-likelihood.

An important note about `bayes: mixed` is the default simulation method. Most `bayes` prefix commands use an adaptive MH algorithm to sample model parameters. The high-dimensional nature of multilevel models greatly decreases the simulation efficiency of this algorithm. For Gaussian multilevel models, such as `bayes: mixed`, model parameters can be sampled using a more efficient, albeit slower, Gibbs algorithm under certain prior distributions. The default priors used for regression coefficients and variance components allow the `bayes` prefix to use Gibbs sampling for these parameters with the `mixed` command. If you change the prior distributions or the default blocking structure for some parameters, Gibbs sampling may not be available for those parameters and an adaptive MH sampling will be used instead.

◁

▷ Example 13: Random-intercept model, default output

When we specified the `melabel` option with `bayes` in example 12, we intentionally suppressed some of the essential output from `bayes: mixed`. Here is what we would have seen had we not specified `melabel`.

```
. bayes
Multilevel structure
------------------------------------------------------------------------------
id
    {U0}: random intercepts
------------------------------------------------------------------------------

Model summary
------------------------------------------------------------------------------
Likelihood:
  weight ~ normal(xb_weight,{e.weight:sigma2})

Priors:
  {weight:week _cons} ~ normal(0,10000)                                    (1)
                 {U0} ~ normal(0,{U0:sigma2})                              (1)
    {e.weight:sigma2} ~ igamma(.01,.01)

Hyperprior:
  {U0:sigma2} ~ igamma(.01,.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_weight.
```

```
Bayesian multilevel regression                   MCMC iterations  =     12,500
Metropolis-Hastings and Gibbs sampling           Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Group variable: id                               Number of groups =         48
                                                 Obs per group:
                                                              min =          9
                                                              avg =        9.0
                                                              max =          9
                                                 Number of obs    =        432
                                                 Acceptance rate  =      .8112
                                                 Efficiency:  min =    .007005
                                                              avg =      .5064
Log marginal-likelihood                                       max =          1

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
weight       |
        week |  6.209734   .0390718   .000391   6.209354   6.133233   6.285611
       _cons |  19.46511   .6239712    .07455   19.48275    18.2534   20.67396
-------------+----------------------------------------------------------------
id           |
   U0:sigma2 |   15.7247   3.436893   .049048   15.26104   10.31182   23.60471
-------------+----------------------------------------------------------------
e.weight     |
      sigma2 |  4.411155   .3193582   .004397   4.396044   3.834341   5.080979
------------------------------------------------------------------------------
Note: Default priors are used for model parameters.
```

Let's go over the default output in detail, starting with the model summary. For multilevel models, in addition to the model summary, which describes the likelihood model and prior distributions, the `bayes` prefix displays information about the multilevel structure of the model.

```
Multilevel structure
------------------------------------------------------------------------------
id
    {U0}: random intercepts
------------------------------------------------------------------------------
```

Our multilevel model has one set of random effects, labeled as `U0`, which represent random intercepts at the `id` level. Recall that in Bayesian models, random effects are not integrated out but estimated together with other model parameters. So, `{U0}`, or using its full name `{U0[id]}`, represent random-effects parameters in our model. See *Likelihood model* to learn about the default naming convention for random-effects parameters.

According to the model summary below, the likelihood of the model is a normal linear regression with the linear predictor containing regression parameters `{weight:week}` and `{weight:_cons}` and random-effects parameters `{U0}`, and with the error variance labeled as `{e.weight:sigma2}`. Regression coefficients `{weight:week}` and `{weight:_cons}` have default normal priors with zero means and variances of 10,000. The random intercepts `{U0}` are normally distributed with mean zero and variance `{U0:sigma2}`. The variance components, error variance `{e.weight:sigma2}`, and random-intercept variance `{U0:sigma2}` have default inverse-gamma priors, $\text{InvGamma}(0.01, 0.01)$. The random-intercept variance is a hyperparameter in our model.

```
Model summary
------------------------------------------------------------------------------
Likelihood:
  weight ~ normal(xb_weight,{e.weight:sigma2})

Priors:
  {weight:week _cons} ~ normal(0,10000)                                   (1)
                 {U0} ~ normal(0,{U0:sigma2})                             (1)
    {e.weight:sigma2} ~ igamma(.01,.01)

Hyperprior:
  {U0:sigma2} ~ igamma(.01,.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_weight.
```

The default output table of `bayes: mixed` uses the names of model parameters as they are defined by the `bayes` prefix.

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
weight						
week	6.209734	.0390718	.000391	6.209354	6.133233	6.285611
_cons	19.46511	.6239712	.07455	19.48275	18.2534	20.67396
id						
U0:sigma2	15.7247	3.436893	.049048	15.26104	10.31182	23.60471
e.weight						
sigma2	4.411155	.3193582	.004397	4.396044	3.834341	5.080979

```
Note: Default priors are used for model parameters.
```

Becoming familiar with the native parameter names of the `bayes` prefix is important for prior specification and for later postestimation. The `melabel` option is provided for easier comparison of the results between the `bayes` prefix and the corresponding frequentist multilevel command.

◁

▷ Example 14: Displaying random effects

By default, the `bayes` prefix does not compute or display MCMC summaries for the random-effects parameters to conserve space and computational time. You can specify the `showreffects` option to display all random effects or the `showreffects()` or `show()` option to display specific random effects. For example, continuing example 13, we can display the random-effects estimates for the first five pigs as follows.

```
. bayes, show({U0[1/5]}) noheader
```

U0[id]	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
1	-1.778442	.8873077	.074832	-1.761984	-3.542545	.0062218
2	.7831408	.8775376	.071421	.7961802	-.9547035	2.491798
3	-2.052634	.9038672	.072325	-2.061559	-3.822966	-.3246834
4	-1.891103	.878177	.075611	-1.858056	-3.642227	-.1028766
5	-3.316584	.8894319	.074946	-3.320502	-5.0469	-1.568927

These posterior mean estimates of random-effects parameters should be comparable with those predicted by `predict, reffects` after `mixed`. Posterior standard deviations, however, will generally be larger than the corresponding standard errors of random effects predicted after `mixed`, because the latter do not incorporate the uncertainty about the estimated model parameters.

You can also use [BAYES] **bayesstats summary** to obtain MCMC summaries of random-effects parameters after estimation:

```
. bayesstats summary {U0[1/5]}
  (output omitted)
```

If you decide to use the `showreffects` option to display all random-effects parameters, beware of the increased computation time for models with many random effects. Then, the `bayes` prefix will compute and display the MCMC summaries for only the first M random-effects parameters, where M is the maximum matrix dimension (`c(max_matdim)`). The number of parameters displayed and stored in `e(b)` cannot exceed `c(max_matdim)`. You can specify the `show()` option with `bayes` or use `bayesstats summary` to obtain results for other random-effects parameters.

◁

▷ Example 15: Random-coefficient model

Continuing example 13, let's consider a random-coefficient model that allows the growth rate to vary among pigs.

Following `mixed`'s specification, we include the random slope for `week` at the `id` level by specifying the `week` variable in the random-effects equation.

```
. set seed 15
. bayes: mixed weight week || id: week
note: Gibbs sampling is used for regression coefficients and variance
      components.
Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done
Multilevel structure
------------------------------------------------------------------------------
id
    {U0}: random intercepts
    {U1}: random coefficients for week
------------------------------------------------------------------------------
Model summary
------------------------------------------------------------------------------
Likelihood:
  weight ~ normal(xb_weight,{e.weight:sigma2})
Priors:
  {weight:week _cons} ~ normal(0,10000)                                    (1)
                 {U0} ~ normal(0,{U0:sigma2})                              (1)
                 {U1} ~ normal(0,{U1:sigma2})                              (1)
    {e.weight:sigma2} ~ igamma(.01,.01)
Hyperpriors:
  {U0:sigma2} ~ igamma(.01,.01)
  {U1:sigma2} ~ igamma(.01,.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_weight.
```

```
Bayesian multilevel regression                   MCMC iterations  =     12,500
Metropolis-Hastings and Gibbs sampling           Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Group variable: id                               Number of groups =         48
                                                 Obs per group:
                                                              min =          9
                                                              avg =        9.0
                                                              max =          9
                                                 Number of obs    =        432
                                                 Acceptance rate  =      .7473
                                                 Efficiency:  min =    .003057
                                                              avg =     .07487
Log marginal-likelihood                                       max =      .1503

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
weight       |
        week |  6.233977   .0801192    .01449   6.237648    6.05268   6.387741
       _cons |  19.44135   .3426786   .044377   19.44532   18.76211   20.11843
-------------+----------------------------------------------------------------
id           |
   U0:sigma2 |  7.055525   1.649394   .050935   6.844225   4.466329   10.91587
   U1:sigma2 |  .3941786   .0901945   .002717   .3825387   .2526756   .6044887
-------------+----------------------------------------------------------------
e.weight     |
      sigma2 |  1.613775   .1261213   .003254   1.609296   1.386427   1.880891
------------------------------------------------------------------------------
Note: Default priors are used for model parameters.
Note: There is a high autocorrelation after 500 lags.
```

In addition to random intercepts {U0}, we now have random coefficients for week, labeled as {U1}, with the corresponding variance parameter {U1:sigma2}. Compared with the random-intercept model, by capturing the variability of slopes on week, we reduced the estimates of the error variance and the random-intercept variance.

The average simulation efficiency decreased to only 7%, and we now see a note about a high autocorrelation after 500 lags. We can use, for example, bayesgraph diagnostics to verify that the high autocorrelation in this example is not an indication of nonconvergence but rather of a slow mixing of our MCMC sample. If we use bayesstats ess, we will see that the coefficient on weight and the constant term have the lowest efficiency, which suggests that these parameters are likely to be correlated with some of the random-effects estimates. If we want to reduce the autocorrelation and improve precision of the estimates for these parameters, we can increase the MCMC sample size by specifying the mcmcsize() option or thin the MCMC chain by specifying the thinning() option.

◁

▷ Example 16: Random-coefficient model, unstructured covariance

In example 15, we assumed independence between random intercepts {U0} and random slopes on week, {U1}. We relax this assumption here by specifying an unstructured covariance matrix.

Before we proceed with estimation, let's review our model summary first by specifying the dryrun option.

```
. bayes, dryrun: mixed weight week || id: week, covariance(unstructured)
Multilevel structure
------------------------------------------------------------------------------
id
    {U0}: random intercepts
    {U1}: random coefficients for week
------------------------------------------------------------------------------

Model summary
------------------------------------------------------------------------------
Likelihood:
  weight ~ normal(xb_weight,{e.weight:sigma2})

Priors:
  {weight:week _cons} ~ normal(0,10000)                                    (1)
            {U0 U1} ~ mvnormal(2,{U:Sigma,m})                              (1)
    {e.weight:sigma2} ~ igamma(.01,.01)

Hyperprior:
  {U:Sigma,m} ~ iwishart(2,3,I(2))
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_weight.
```

The prior distributions for random effects {U0} and {U1} are no longer independent. Instead, they have a joint prior—a bivariate normal distribution with covariance matrix parameter {U:Sigma,m}, which is short for {U:Sigma,matrix}. The random-effects stub U is used to label the covariance matrix. The covariance matrix {U:Sigma,m} is assigned a fairly uninformative inverse-Wishart prior with three degrees of freedom and an identity scale matrix; see *Default priors* for details.

Let's now fit the model but suppress the model summary for brevity.

```
. set seed 15

. bayes, nomodelsummary: mixed weight week || id: week, covariance(unstructured)
note: Gibbs sampling is used for regression coefficients and variance
      components.

Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done

Multilevel structure
------------------------------------------------------------------------------
id
    {U0}: random intercepts
    {U1}: random coefficients for week
------------------------------------------------------------------------------

Bayesian multilevel regression                  MCMC iterations  =     12,500
Metropolis-Hastings and Gibbs sampling          Burn-in          =      2,500
                                                MCMC sample size =     10,000
Group variable: id                              Number of groups =         48
                                                Obs per group:
                                                             min =          9
                                                             avg =        9.0
                                                             max =          9
                                                Number of obs    =        432
                                                Acceptance rate  =      .7009
                                                Efficiency:  min =    .003683
                                                             avg =     .07461
Log marginal-likelihood                                      max =      .1602

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
weight       |
        week |  6.207086   .0878022   .014469   6.204974   6.041093   6.384891
       _cons |  19.39551   .4077822   .050353   19.40187   18.53869    20.1993
-------------+----------------------------------------------------------------
id           |
 U:Sigma_1_1 |  6.872161   1.627769   .061568   6.673481   4.282284   10.62194
 U:Sigma_2_1 | -.0866373   .2702822   .009861  -.0796118   -.645439   .4341423
 U:Sigma_2_2 |   .399525   .0904532   .002488   .3885861   .2575883   .6104775
-------------+----------------------------------------------------------------
e.weight     |
      sigma2 |  1.611889   .1263131   .003155   1.605368   1.381651   1.872563
------------------------------------------------------------------------------
Note: Default priors are used for model parameters.
Note: There is a high autocorrelation after 500 lags.
```

The 95% credible interval for the covariance between {U0} and {U1}, labeled as {U:Sigma_2_1} in the output, is $[-.65, 0.43]$, which suggests independence between {U0} and {U1}.

The high autocorrelation note is due to the lower sampling efficiency of some of the regression coefficients as can be seen from the output of `bayesstats ess`:

```
. bayesstats ess

Efficiency summaries    MCMC sample size =    10,000
                        Efficiency:  min =   .003683
                                     avg =    .07461
                                     max =     .1602

--------------------------------------------------------
                      |        ESS   Corr. time    Efficiency
----------------------+---------------------------------
weight                |
                 week |      36.83       271.55        0.0037
                _cons |      65.58       152.48        0.0066
----------------------+---------------------------------
id                    |
          U:Sigma_1_1 |     698.99        14.31        0.0699
          U:Sigma_2_1 |     751.20        13.31        0.0751
          U:Sigma_2_2 |    1321.67         7.57        0.1322
----------------------+---------------------------------
e.weight              |
               sigma2 |    1602.39         6.24        0.1602
--------------------------------------------------------
```

We explore the impact of this high autocorrelation on MCMC convergence in example 17.

◁

▷ Example 17: Random-coefficient model, multiple chains

We continue with the random-coefficient model with unstructured covariance from example 16. Some of the parameters such as the coefficients `{weight:week}` and `{weight:_cons}` have low sampling efficiency, which raises convergence and precision concerns. Simulating multiple Markov chains of the model may help address these concerns.

We will simulate three chains by specifying the `nchains(3)` option. We will use the `rseed(15)` option to ensure reproducibility with multiple chains; see *Reproducing results* in [BAYES] **bayesmh**. We will also suppress various model summaries by specifying the `nomodelsummary` and `nomesummary` options.

When using multiple chains to assess convergence, it is important to apply overdispersed initial values for different chains. It is difficult to quantify overdispersion because it is specific to the data and model. The default initialization provided by the `bayes: mixed` command may or may not be sufficient. To be certain, we recommend that you provide initial values explicitly, at least for the main parameters of interest. In the following specification, we provide initial values for the two regression coefficients referred to as `{weight:}`, the variance parameter `{e.weight:sigma2}`, and the covariance matrix `{U:Sigma, matrix}`. We try to generate initial values that are sufficiently separated. For example, we use `rnormal(-10, 100)` for the regression coefficients in the second chain and `rnormal(10, 100)` in the third chain. Specifying initial values for the random effects `{U0}` and `{U1}` would be more tedious, so we let them be sampled from their corresponding prior distributions. Because the hyperparameters of these priors have overdispersed initial values, we indirectly provide some overdispersion for the initial random effects as well.

```
. bayes, nchains(3) rseed(15) nomodelsummary nomesummary
> init2({weight:} rnormal(-10,100) {e.weight:sigma2} 0.1 {U:Sigma,m} 100*I(2))
> init3({weight:} rnormal(10,100) {e.weight:sigma2} 100 {U:Sigma,m} (10,-5\-5,10)):
> mixed weight week || id: week, covariance(unstructured)
note: Gibbs sampling is used for regression coefficients and variance
      components.
Chain 1
  Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
  Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done
Chain 2
  Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
  Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done
Chain 3
  Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
  Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done
Bayesian multilevel regression              Number of chains    =          3
Metropolis-Hastings and Gibbs sampling      Per MCMC chain:
                                                Iterations      =     12,500
                                                Burn-in         =      2,500
                                                Sample size     =     10,000
Group variable: id                          Number of groups    =         48
                                            Obs per group:
                                                            min =          9
                                                            avg =        9.0
                                                            max =          9
                                            Number of obs       =        432
                                            Avg acceptance rate =      .6981
                                            Avg efficiency: min =    .003059
                                                            avg =     .07659
                                                            max =      .1663
Log marginal-likelihood                     Max Gelman-Rubin Rc =      1.055

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
weight       |
        week |  6.201475   .0874855   .009133   6.200176   6.032975   6.374917
       _cons |   19.3941   .4344171   .035266   19.38919   18.52954    20.2323
-------------+----------------------------------------------------------------
id           |
 U:Sigma_1_1 |  6.863804     1.6219   .035988   6.653249   4.329726   10.62575
 U:Sigma_2_1 | -.0799526   .2684949   .005546  -.0723027  -.6351419   .4354943
 U:Sigma_2_2 |  .3983365   .0890525   .001378   .3869276    .258562   .6048894
-------------+----------------------------------------------------------------
e.weight     |
      sigma2 |  1.612452   .1254983   .001777   1.605632   1.383175   1.874105
------------------------------------------------------------------------------
Note: Default priors are used for model parameters.
Note: Default initial values are used for multiple chains.
Note: There is a high autocorrelation after 500 lags in at least one of the
      chains.
```

While the sampling efficiency of the chains is about the same as in example 16, having three MCMC samples instead of one improves the precision of the estimation results, as evident from the lower MCMC errors for all model parameters.

Let's compute Gelman–Rubin diagnostics as a convergence check. We can already see in the header of `bayes: mixed` that the maximum Gelman–Rubin statistic `Rc` of 1.055 is close to 1.

```
. bayesstats grubin

Gelman-Rubin convergence diagnostic

Number of chains      =          3
MCMC size, per chain =     10,000
Max Gelman-Rubin Rc   =   1.055383

---------------------------
             |          Rc
-------------+-------------
weight       |
        week |   1.006404
       _cons |   1.055383
-------------+-------------
id           |
 U:Sigma_1_1 |   1.000567
 U:Sigma_2_1 |   1.001168
 U:Sigma_2_2 |   1.002119
-------------+-------------
e.weight     |
      sigma2 |   .9999899
---------------------------
Convergence rule: Rc < 1.1
```

The convergence diagnostic estimates `Rc` for all reported parameters are lower than 1.1, suggesting the convergence of the chains. We can also explore MCMC convergence visually; see [BAYES] **bayesgraph**.

◁

Crossed-effects model

Let's revisit example 4 from [ME] **meglm**, which analyzes salamander cross-breeding data. Two populations of salamanders are considered: whiteside males and females (variables `wsm` and `wsf`) and roughbutt males and females (variables `rbm` and `rbf`). Male and female identifiers are recorded in the `male` and `female` variables. The outcome binary variable `y` indicates breeding success or failure.

In example 4 of [ME] **meglm**, we fit a crossed-effects logistic regression for successful mating, in which the effects of `male` and `female` were crossed. For the purpose of illustration, we will fit a crossed-effects probit regression here using `meglm` with the probit link.

```
. use https://www.stata-press.com/data/r19/salamander

. meglm y wsm##wsf || _all: R.male || female:, family(bernoulli) link(probit)
note: crossed random-effects model specified; option intmethod(laplace)
      implied.

Fitting fixed-effects model:

Iteration 0:  Log likelihood = -223.01026
Iteration 1:  Log likelihood = -222.78736
Iteration 2:  Log likelihood = -222.78735

Refining starting values:

Grid node 0:  Log likelihood = -216.49485
```

```
Fitting full model:

Iteration 0:  Log likelihood = -216.49485  (not concave)
Iteration 1:  Log likelihood = -214.34477  (not concave)
Iteration 2:  Log likelihood = -212.34877  (not concave)
Iteration 3:  Log likelihood = -212.15484
Iteration 4:  Log likelihood = -209.36104  (not concave)
Iteration 5:  Log likelihood = -209.34854
Iteration 6:  Log likelihood = -208.26891
Iteration 7:  Log likelihood = -208.11369
Iteration 8:  Log likelihood = -208.11183
Iteration 9:  Log likelihood = -208.11182

Mixed-effects GLM                               Number of obs     =        360
Family: Bernoulli
Link:   Probit

        Grouping information
        -------------------------------------------------------------
                        |     No. of       Observations per group
         Group variable |     groups    Minimum    Average    Maximum
        ----------------+--------------------------------------------
                   _all |          1        360      360.0        360
                 female |         60          6        6.0          6
        -------------------------------------------------------------

Integration method: laplace

                                                Wald chi2(3)      =      40.58
Log likelihood = -208.11182                     Prob > chi2       =     0.0000

-------------------------------------------------------------------------------
            y | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
--------------+----------------------------------------------------------------
        1.wsm |  -.4122104   .2715152    -1.52   0.129    -.9443705    .1199496
        1.wsf |  -1.720297   .3232692    -5.32   0.000    -2.353893   -1.086701
              |
      wsm#wsf |
         1 1  |   2.121105   .3643124     5.82   0.000     1.407066    2.835144
              |
        _cons |   .5951036   .2297373     2.59   0.010     .1448267     1.04538
--------------+----------------------------------------------------------------
_all>male     |
   var(_cons) |    .386743    .178314                      .1566616     .954734
--------------+----------------------------------------------------------------
female        |
   var(_cons) |   .4464129   .1980076                      .1871475    1.064852
-------------------------------------------------------------------------------
LR test vs. probit model: chi2(2) = 29.35                 Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.
```

To fit the corresponding Bayesian model, we prefix the above command with bayes:.

```
. set seed 15

. bayes: meglm y wsm##wsf || _all: R.male || female:, family(bernoulli)
> link(probit)

Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done

Multilevel structure
------------------------------------------------------------------------------
male
    {U0}: random intercepts

female
    {V0}: random intercepts
------------------------------------------------------------------------------

Model summary
------------------------------------------------------------------------------
Likelihood:
  y ~ meglm(xb_y)

Priors:
  {y:1.wsm 1.wsf 1.wsm#1.wsf _cons} ~ normal(0,10000)                      (1)
                               {U0} ~ normal(0,{U0:sigma2})                (1)
                               {V0} ~ normal(0,{V0:sigma2})                (1)

Hyperpriors:
  {U0:sigma2} ~ igamma(.01,.01)
  {V0:sigma2} ~ igamma(.01,.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_y.

Bayesian multilevel GLM                          MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
```

Group variable	No. of groups	Observations per group Minimum	Average	Maximum
_all	1	360	360.0	360
female	60	6	6.0	6

```
Family: Bernoulli                                     Number of obs    =        360
Link:   probit                                        Acceptance rate  =      .3223
                                                      Efficiency:  min =    .008356
                                                                   avg =     .02043
Log marginal-likelihood                                            max =     .02773

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
y            |
       1.wsm |  -.411886     .28122   .016889  -.4158334  -.9645049    .156521
       1.wsf | -1.722195   .3329918   .023312  -1.713574  -2.381169  -1.094443
             |
     wsm#wsf |
        1 1  |  2.110366   .3671998   .022643    2.09234   1.443113   2.831923
             |
       _cons |  .5858733   .2512646   .015407   .5906893   .0812177   1.077352
-------------+----------------------------------------------------------------
male         |
   U0:sigma2 |  .4291858   .2195246   .024015   .3876708   .1347684   .9648611
-------------+----------------------------------------------------------------
female       |
   V0:sigma2 |  .4928416   .2189307   .019043   .4576824   .1648551   1.003193
------------------------------------------------------------------------------
Note: Default priors are used for model parameters.
```

The variance components for male and female, {U0:sigma2} and {V0:sigma2}, are slightly higher than the corresponding ML estimates, but the regression coefficients are similar.

For an example of Bayesian estimation of a crossed-effects logistic regression model, see Rabe-Hesketh and Skrondal (2022, chap. 16).

Blocked-diagonal covariance structures

The 1989 fertility survey considered in example 5 of [ME] **me** analyzes the use of contraception among Bangladeshi women. The survey contains data from 60 districts, identified by the district variable, and includes demographic factors such as whether the woman is from an urban area (urban), mean-centered age (age), and number of children (children). Here children is a factor variable coded as children = 0 (no children), children = 1 (one child), children = 2 (two children), and children = 3 (three or more children). The outcome variable c_use is a binary indicator for the use of contraception.

We consider a two-level logit model for c_use with a random intercept and random coefficients for indicators of having one, two, or three or more children. As "fixed" predictor variables, we use urban, age, and children.

It seems reasonable to expect positive correlation between the three random coefficients. Following example 5 in [ME] **me**, we will use the covariance(exchangeable) option and repeat district: to specify a blocked-diagonal covariance structure for the random effects.

Let's first run `bayes: melogit` with the `dryrun` option to see the model parameters.

```
. use https://www.stata-press.com/data/r19/bangladesh
(Bangladesh Fertility Survey, 1989)

. bayes, dryrun: melogit c_use i.urban age i.children ||
> district: i.children, covariance(exchangeable) ||
> district:
Multilevel structure
------------------------------------------------------------------------------
district
    {U0}: random intercepts
    {U1}: random coefficients for 1.children
    {U2}: random coefficients for 2.children
    {U3}: random coefficients for 3.children
------------------------------------------------------------------------------

Model summary
------------------------------------------------------------------------------
Likelihood:
  c_use ~ melogit(xb_c_use)

Priors:
 {c_use:1.urban age i.children _cons} ~ normal(0,10000)                   (1)
                                 {U0} ~ normal(0,{U0:sigma2})             (1)
                           {U1 U2 U3} ~ mvn0exchangeable(3,{U:sigma2},{U:rho})
                                         (1)

Hyperpriors:
      {U:rho} ~ uniform(-1,1)
  {U0:sigma2} ~ igamma(.01,.01)
   {U:sigma2} ~ igamma(.01,.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_c_use.
```

The random coefficients `{U1}`, `{U2}`, and `{U3}` are assigned a multivariate normal prior with an exchangeable covariance structure, `mvn0exchangeable()`. This prior introduces two hyperparameters: `{U:sigma2}`, for the diagonal variance term of the covariance matrix, and `{U:rho}`, for the off-diagonal correlation term such that the covariance is equal to `{U:sigma2}`×`{U:rho}`. The random intercept `{U0}` is assigned a normal prior with hyperparameter `{U0:sigma2}` for its variance. It is recommended to assign informative priors to `{U0:sigma2}`, `{U:sigma2}`, and `{U:rho}`. For example, we believe the correlation parameter to be between 0 and 0.5 and thus assign the `uniform(0, 0.5)` prior to `{U:rho}`. In addition, let's say that, from historical data, the mean variability for children random coefficients was found to be about 0.2 and the mean variability for the random intercepts was found to be about 0.25. We may then assign the `igamma(11,2)` prior to `{U:sigma2}` and the `igamma(9,2)` prior to `{U0:sigma2}` to incorporate this prior knowledge. We will also add the `or` option to obtain estimates of the odds ratios.

```
. bayes, prior({U:rho}, uniform(0,0.5)) prior({U:sigma2}, igamma(11,2))
> prior({U0:sigma2}, igamma(9,2)) rseed(17):
> melogit c_use i.urban age i.children ||
> district: i.children, covariance(exchangeable) ||
> district:, or

Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done

Multilevel structure
------------------------------------------------------------------------------
district
    {U0}: random intercepts
    {U1}: random coefficients for 1.children
    {U2}: random coefficients for 2.children
    {U3}: random coefficients for 3.children
------------------------------------------------------------------------------

Model summary
------------------------------------------------------------------------------
Likelihood:
  c_use ~ melogit(xb_c_use)

Priors:
 {c_use:1.urban age i.children _cons} ~ normal(0,10000)                    (1)
                                 {U0} ~ normal(0,{U0:sigma2})              (1)
                           {U1 U2 U3} ~ mvn0exchangeable(3,{U:sigma2},{U:rho})
                                        (1)

Hyperpriors:
      {U:rho} ~ uniform(0,0.5)
   {U:sigma2} ~ igamma(11,2)
  {U0:sigma2} ~ igamma(9,2)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_c_use.
```

```
Bayesian multilevel logistic regression          MCMC iterations  =     12,500
Random-walk Metropolis-Hastings sampling         Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Group variable: district                         Number of groups =         60
                                                 Obs per group:
                                                              min =          2
                                                              avg =       32.2
                                                              max =        118
Family: Bernoulli                                Number of obs    =      1,934
Link:   logit                                    Acceptance rate  =      .2401
                                                 Efficiency:  min =    .009968
                                                              avg =     .02371
Log marginal-likelihood                                       max =     .04605

------------------------------------------------------------------------------
             |                                                Equal-tailed
             | Odds ratio   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
c_use        |
     1.urban |   2.153732   .2632265   .023028   2.135123   1.710943   2.728066
         age |   .9734474   .0076718   .000478   .9736178   .9585345   .9887891
             |
    children |
          1  |   3.043873   .5490154    .03425    3.00129   2.119798   4.241168
          2  |   4.030936   .7761135   .040228   3.949568    2.77722   5.714252
          3  |    3.85945    .724596   .047131   3.778789   2.644804   5.448504
             |
       _cons |   .1850523   .0271077   .002155   .1827656   .1395885    .242633
-------------+----------------------------------------------------------------
district     |
       U:rho |   .3236901   .1286163   .010136   .3422138   .0326351   .4943052
   U0:sigma2 |   .2147372   .0541223   .002522   .2069007   .1315863   .3416939
    U:sigma2 |   .1736623   .0435398   .004361   .1676818   .1039366   .2793393
------------------------------------------------------------------------------
Note: Estimates are transformed only in the first equation to odds ratios.
Note: _cons estimates baseline odds (conditional on zero random effects).
Note: Default priors are used for some model parameters.
```

The posterior odds-ratio estimates for the fixed-effects parameters are close to the estimates reported by the `melogit` command in example 5. Our model reports an estimate of 0.32 for the correlation between random coefficients, a variance of 0.17 for the random coefficients, and a variance of 0.21 for the random intercepts.

Panel-data models

The `bayes` prefix supports several panel-data commands such as `xtreg` and `xtlogit`; see [BAYES] **Bayesian estimation**.

Panel-data models, also known as longitudinal-data models, are used for analyzing cross-sectional time series when there is an explicit time component. Panel-data models require that the panel variable be specified using the `xtset` command. See [XT] **xt** for details.

Panel-data models can also be viewed as two-level random-intercept models, so many comments from *Multilevel models* apply to these models too.

All Bayesian panel-data models include random intercepts, referred to as `{U[`*panelvar*`]}` or simply `{U}`, with the panel variable *panelvar* used as the grouping variable. These intercepts are commonly referred to as random effects in frequentist models.

Random intercepts are assigned default prior distributions specific to the likelihood family of the model. For linear and generalized linear models, the default prior is normal with zero mean and unknown variance {var_U}. Other models have special random-effects priors, and these are described in *Methods and formulas* of the command-specific bayes entries. Positive hyperparameters such as {var_U} are assigned default inverse-gamma priors. Categorical outcome models such as [BAYES] **bayes: xtmlogit** have multiple random effects. In cases when these random effects are correlated, the model includes a matrix hyperparameter {U:Sigma,m} that is assigned a default inverse-Wishart prior.

You can specify your own priors for regression coefficients, random effects, and auxiliary model parameters. To change the default priors, you will need to know the names of the model parameters. See *Likelihood model* to learn how the bayes prefix labels the parameters. You can also use the dryrun option to see the names of model parameters specific to each bayes model before estimation. After estimation, see *Different ways of specifying model parameters* for how to refer to individual random effects to evaluate MCMC convergence or to obtain their MCMC summaries.

Bayesian panel-data models estimate random effects together with regression coefficients and other model parameters. By default, the bayes prefix does not compute or display MCMC summaries of individual random effects to conserve computation time and space. You can specify the showreffects() or show() option to compute and display them for chosen subsets of random effects.

By default, all panel-data models use Gibbs sampling for variance components. Linear panel-data models, bayes: xtreg, additionally use Gibbs sampling for regression coefficients. With bayes: xtreg, we can specify Gibbs sampling also for random effects by using the gibbs option.

Unlike other bayes commands, panel-data models support the [BAYES] **bayespredict** postestimation command to compute Bayesian predictions; see examples in [BAYES] **bayes: xtpoisson** and [BAYES] **bayes: xtmlogit**.

▷ Example 18: Random-effects linear model

In example 12, we considered a random-intercept model analyzing the weight gain of pigs. In that example, the dependent variable, weight, is regressed on variable week, and random intercepts are introduced with respect to the group variable id. Let's fit the same random-intercept model but now using bayes: xtreg. First, we should declare our data as panel data.

```
. use https://www.stata-press.com/data/r19/pig
(Longitudinal analysis of pig weights)
. xtset id
Panel variable: id (balanced)
```

We can use `bayes: xtreg` to fit the same model that we previously fit using `bayes: mixed`. Both commands use the same default priors and the same default sampling method.

```
. bayes, rseed(17): xtreg weight week
note: Gibbs sampling is used for regression coefficients and variance
      components.
Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done
Model summary
------------------------------------------------------------------------------
Likelihood:
  weight ~ normal(xb_weight,{sigma2})
Priors:
  {weight:week _cons} ~ normal(0,10000)                                    (1)
               {U[id]} ~ normal(0,{var_U})                                  (1)
              {sigma2} ~ igamma(0.01,0.01)
Hyperprior:
  {var_U} ~ igamma(0.01,0.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_weight.
Bayesian RE normal regression                    MCMC iterations  =     12,500
Metropolis-Hastings and Gibbs sampling           Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Group variable: id                               Number of groups =         48
                                                 Obs per group:
                                                              min =          9
                                                              avg =        9.0
                                                              max =          9
                                                 Number of obs    =        432
                                                 Acceptance rate  =      .8089
                                                 Efficiency:  min =    .008983
                                                              avg =      .5507
Log marginal-likelihood                                       max =          1
------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
weight       |
        week |  6.209598   .0391057   .000391   6.209511   6.134362    6.28693
       _cons |   19.2624   .5480876   .057828   19.23869   18.18444   20.36098
-------------+----------------------------------------------------------------
       var_U |  15.75035   3.489106   .042737   15.31299   10.28186    23.8984
      sigma2 |  4.417614   .3188951   .004392   4.401373   3.837572    5.07726
------------------------------------------------------------------------------
Note: Default priors are used for model parameters.
```

The results are similar to those from example 12, up to MCMC sampling variation.

To improve efficiency, all panel-data models by default use Gibbs sampling for variance components. Panel-data linear models (`bayes: xtreg`) use Gibbs sampling also for regression coefficients. With `bayes: xtreg`, we can improve sampling efficiency further by specifying the `gibbs` option to use Gibbs sampling also for random effects. Beware that, depending on the number of random effects, this may increase the computation time substantially.

```
. bayes, gibbs rseed(17): xtreg weight week
note: Gibbs sampling is used for all parameters, including random effects.

Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000...... ...9000.........10000 done

Model summary
------------------------------------------------------------------------------
Likelihood:
  weight ~ normal(xb_weight,{sigma2})

Priors:
  {weight:week _cons} ~ normal(0,10000)                                    (1)
              {U[id]} ~ normal(0,{var_U})                                  (1)
             {sigma2} ~ igamma(0.01,0.01)

Hyperprior:
  {var_U} ~ igamma(0.01,0.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_weight.

Bayesian RE normal regression                    MCMC iterations  =     12,500
Gibbs sampling                                   Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Group variable: id                               Number of groups =         48
                                                 Obs per group:
                                                              min =          9
                                                              avg =        9.0
                                                              max =          9
                                                 Number of obs    =        432
                                                 Acceptance rate  =          1
                                                 Efficiency:  min =     .01606
                                                              avg =      .6605
Log marginal-likelihood                                       max =          1
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
weight						
week	6.209921	.0390177	.00039	6.209939	6.132542	6.285744
_cons	19.26382	.6209709	.048995	19.27342	18.0418	20.5063
var_U	15.80222	3.488439	.038688	15.33375	10.3458	24.03719
sigma2	4.412905	.3236225	.00359	4.395282	3.821423	5.095022

```
Note: Default priors are used for model parameters.
```

Using full Gibbs sampling, we see that our estimates of regression coefficients and variance components are similar but that the minimum efficiency is increased to 0.016 from 0.009.

◁

▷ Example 19: Random-effects ordered logit model

Consider example 1 from [XT] **xtologit**, which analyzes data from a smoking prevention project in schools. The dependent variable, tobacco and health knowledge score thk, has four categories. Predictor variables include preintervention score, prethk, classroom curriculum, cc, and television intervention, tv, as well as the interaction of the last two. The school identifier variable school is set as the panel variable.

```
. use https://www.stata-press.com/data/r19/tvsfpors
(Television, School, and Family Project)

. xtset school

Panel variable: school (unbalanced)
```

The bayes: xtologit command is used to fit a Bayesian model. The default prior distribution for regression coefficients is normal with zero mean and variances of 10,000. The default prior distribution for random effects is normal with mean zero and variance {var_U}. The hyperparameter {var_U} is assigned an inverse-gamma hyperprior. The three cutpoints for the ordered logit likelihood, {_cut1}, {_cut2}, and {_cut3}, are assigned a flat prior.

```
. bayes, rseed(17): xtologit thk prethk cc##tv
note: Gibbs sampling is used for variance components.

Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000..... done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done

Model summary
------------------------------------------------------------------------------

Likelihood:
  thk ~ ologit(xb_thk,{_cut1 ... _cut3})

Priors:
  {thk:prethk 1.cc 1.tv 1.cc#1.tv} ~ normal(0,10000)                       (1)
                       {U[school]} ~ normal(0,{var_U})                     (1)
               {_cut1 _cut2 _cut3} ~ 1 (flat)

Hyperprior:
  {var_U} ~ igamma(0.01,0.01)
------------------------------------------------------------------------------

(1) Parameters are elements of the linear form xb_thk.
```

```
Bayesian RE ordered logistic regression          MCMC iterations  =     12,500
Metropolis-Hastings and Gibbs sampling           Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Group variable: school                           Number of groups =         28
                                                 Obs per group:
                                                              min =         18
                                                              avg =       57.1
                                                              max =        137
                                                 Number of obs    =      1,600
                                                 Acceptance rate  =       .506
                                                 Efficiency:  min =     .00404
                                                              avg =     .01548
Log marginal-likelihood                                       max =     .03692

                                                             Equal-tailed
                    Mean   Std. dev.     MCSE     Median  [95% cred. interval]

thk
      prethk    .4024205     .03817   .001987   .4016996   .3289603    .480875
        1.cc    .9329812   .2127196   .019923   .9304351   .5156044   1.367753
        1.tv    .3037174   .2089864    .03288   .2919775  -.0874367   .7099491

       cc#tv
        1 1    -.4663504   .2985113    .02669  -.4502481  -1.057705   .0993408

       _cut1   -.0960417   .1673066   .016383  -.0987278  -.4235516   .2458889
       _cut2    1.151299   .1739417   .020155   1.148734   .8009236    1.49998
       _cut3    2.340316   .1798423   .020381   2.338304   1.994793   2.696972
       var_U    .1089538   .0529856   .002903   .0988449   .0351552   .2362116

Note: Default priors are used for model parameters.
Note: There is a high autocorrelation after 500 lags.
```

The command issues a high autocorrelation warning because of slower convergence for some of the parameters. You can use `bayesstats ess` to find that `{thk:1.tv}` is the parameter that has the lowest ESS. Slower convergence of panel-data models is often caused by the presence of many random effects, which indirectly influences the convergence of regression coefficients as well.

Sometimes, the sampling efficiency can be improved by simply increasing the burn-in period, thus prolonging the adaptation phase of the sampling algorithm. In the next run, we double the default burn-in period.

```
. bayes, burnin(5000) rseed(17): xtologit thk prethk cc##tv
note: Gibbs sampling is used for variance components.
Burn-in 5000 aaaaaaaaa1000aaaaaaaaa2000.........3000.........4000.........5000
> done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done
Model summary
------------------------------------------------------------------------------
Likelihood:
  thk ~ ologit(xb_thk,{_cut1 ... _cut3})
Priors:
  {thk:prethk 1.cc 1.tv 1.cc#1.tv} ~ normal(0,10000)                       (1)
                      {U[school]} ~ normal(0,{var_U})                      (1)
              {_cut1 _cut2 _cut3} ~ 1 (flat)
Hyperprior:
  {var_U} ~ igamma(0.01,0.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_thk.
Bayesian RE ordered logistic regression          MCMC iterations  =     15,000
Metropolis-Hastings and Gibbs sampling           Burn-in          =      5,000
                                                 MCMC sample size =     10,000
Group variable: school                           Number of groups =         28
                                                 Obs per group:
                                                              min =         18
                                                              avg =       57.1
                                                              max =        137
                                                 Number of obs    =      1,600
                                                 Acceptance rate  =      .5038
                                                 Efficiency:  min =    .003954
                                                              avg =       .015
Log marginal-likelihood                                       max =      .0366
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
thk						
prethk	.4043504	.0380502	.001989	.4033533	.3325402	.4827048
1.cc	.9352501	.2010255	.018787	.9288417	.5673248	1.348453
1.tv	.3041591	.2085135	.033158	.3009742	-.117611	.7077558
cc#tv						
1 1	-.4635365	.2798612	.027015	-.4525074	-1.028432	.0712566
_cut1	-.095777	.1627607	.016387	-.0969997	-.426459	.2438933
_cut2	1.15389	.1684856	.019615	1.154469	.8296157	1.499366
_cut3	2.344848	.1762402	.021575	2.34904	1.993787	2.685564
var_U	.1064932	.0524515	.002873	.0964727	.034738	.2305971

Note: Default priors are used for model parameters.

Compared with the frequentist estimates from example 1, the posterior mean estimates of the regression coefficients and cutpoints are not that different. The most noticeable difference is for the random-effects variance {var_U}, which has a posterior mean of about 0.11, slightly higher than the frequentist estimate of 0.07.

We can use bayesstats summary to display posterior estimates for the first five random effects {U[school]} or simply {U}.

```
. bayesstats summary {U[1/5]}
Posterior summary statistics                      MCMC sample size =    10,000
```

U[school]	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred.	interval]
193	.0983182	.2360735	.008371	.0949512	-.3319545	.5649471
194	.0910507	.2044525	.013411	.0850659	-.3085782	.5080763
196	.1609138	.2372827	.010454	.159283	-.3000192	.6540844
197	-.0351616	.2304207	.009844	-.036144	-.5106465	.4080927
198	-.1724522	.2164482	.019579	-.1666214	-.6123599	.2548694

We could also replace the default priors with more informative ones. There are two ways to do this. First, we can simply modify the parameters of the default prior without changing the family of the distribution. For example, we can use the normalprior(1) option to change the prior standard deviation for regression coefficients from 100 to 1.

```
. bayes, normalprior(1) rseed(17): xtologit thk prethk cc##tv
note: Gibbs sampling is used for variance components.

Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done

Model summary
------------------------------------------------------------------------------
Likelihood:
  thk ~ ologit(xb_thk,{_cut1 ... _cut3})

Priors:
  {thk:prethk 1.cc 1.tv 1.cc#1.tv} ~ normal(0,1)                           (1)
                       {U[school]} ~ normal(0,{var_U})                     (1)
               {_cut1 _cut2 _cut3} ~ 1 (flat)

Hyperprior:
  {var_U} ~ igamma(0.01,0.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_thk.

Bayesian RE ordered logistic regression          MCMC iterations  =     12,500
Metropolis-Hastings and Gibbs sampling           Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Group variable: school                           Number of groups =         28
                                                 Obs per group:
                                                              min =         18
                                                              avg =       57.1
                                                              max =        137
                                                 Number of obs    =      1,600
                                                 Acceptance rate  =      .5083
                                                 Efficiency:  min =    .005659
                                                              avg =     .01438
Log marginal-likelihood                                       max =      .0411

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
thk          |
      prethk |  .3972503   .0386982   .003252   .3967045   .3240223   .4752994
        1.cc |  .8628827   .2182787   .029018   .8597381   .4505967   1.275168
        1.tv |  .2691059   .1952139   .020681   .2561737   -.064717   .6803609
             |
       cc#tv |
         1 1 | -.3874974      .2808   .030905  -.3749463   -.954762   .1415334
-------------+----------------------------------------------------------------
       _cut1 | -.1274545   .1812604   .017455  -.1252054  -.4761576   .2116238
       _cut2 |  1.117835   .1811456   .017375   1.120978   .7740603   1.467072
       _cut3 |   2.30662   .1859104   .015007   2.312644   1.958648   2.666062
       var_U |  .1104883   .0550946   .002718    .100217   .0357647    .239713
------------------------------------------------------------------------------
Note: Default priors are used for some model parameters.
```

The magnitudes of the regression coefficient estimates shrink slightly toward 0. Similarly, we can use the `igammaprior()` option to manipulate the shape and scale of the default inverse-gamma prior for `{var_U}`.

Another way of changing the default priors is to specify the `prior()` options for the selected groups of model parameters. For example, we can change the prior for cutpoints from the default flat to normal with mean 1 and variance 1.

```
. bayes, prior({_cut1 _cut2 _cut3}, normal(1, 1))
> normalprior(1) rseed(17): xtologit thk prethk cc##tv
note: Gibbs sampling is used for variance components.

Burn-in 2500 aaaaaaaaa1000aaaaaaaaa2000aaaaa done
Simulation 10000 .........1000.........2000.........3000.........4000.........
> 5000.........6000.........7000.........8000.........9000.........10000 done

Model summary
------------------------------------------------------------------------------
Likelihood:
  thk ~ ologit(xb_thk,{_cut1 ... _cut3})

Priors:
  {thk:prethk 1.cc 1.tv 1.cc#1.tv} ~ normal(0,1)                           (1)
                      {U[school]} ~ normal(0,{var_U})                      (1)
              {_cut1 _cut2 _cut3} ~ normal(1,1)

Hyperprior:
  {var_U} ~ igamma(0.01,0.01)
------------------------------------------------------------------------------
(1) Parameters are elements of the linear form xb_thk.

Bayesian RE ordered logistic regression          MCMC iterations  =     12,500
Metropolis-Hastings and Gibbs sampling           Burn-in          =      2,500
                                                 MCMC sample size =     10,000
Group variable: school                           Number of groups =         28
                                                 Obs per group:
                                                              min =         18
                                                              avg =       57.1
                                                              max =        137
                                                 Number of obs    =      1,600
                                                 Acceptance rate  =      .4909
                                                 Efficiency:  min =    .005571
                                                              avg =     .01344
Log marginal-likelihood                                       max =     .04221

------------------------------------------------------------------------------
             |                                                Equal-tailed
             |      Mean   Std. dev.     MCSE     Median  [95% cred. interval]
-------------+----------------------------------------------------------------
thk          |
      prethk |  .3914625   .0344846    .00462   .3902991   .3256868   .4578337
        1.cc |   .832213   .2079096   .024539   .8433861   .4080022    1.20791
        1.tv |  .1969988   .2044468   .016094   .2080927  -.2166963   .5690862
             |
       cc#tv |
        1 1  | -.3620582   .2739768   .032021   -.377875  -.9000601   .2192883
-------------+----------------------------------------------------------------
       _cut1 | -.1775701   .1673107   .016436  -.1657233  -.5312352   .1188874
       _cut2 |  1.063019   .1684814   .018284   1.074538   .7075167    1.37078
       _cut3 |  2.240986   .1739471   .017195   2.251752   1.881608   2.556478
       var_U |  .1058796   .0550203   .002678   .0952031   .0334108   .2404828
------------------------------------------------------------------------------
Note: Default priors are used for some model parameters.
```

◁

Time-series and DSGE models

The bayes prefix also supports vector autoregression ([BAYES] **bayes: var**), linear DSGE models ([BAYES] **bayes: dsge**), and nonlinear DSGE models ([BAYES] **bayes: dsgenl**). See the corresponding entries for examples of these commands.

Video examples

Introduction to Bayesian statistics, part 1: The basic concepts

Introduction to Bayesian statistics, part 2: MCMC and the Metropolis–Hastings algorithm

A prefix for Bayesian regression in Stata

Bayesian linear regression using the bayes prefix

Bayesian linear regression using the bayes prefix: How to specify custom priors

Bayesian linear regression using the bayes prefix: Checking convergence of the MCMC chain

Bayesian linear regression using the bayes prefix: How to customize the MCMC chain

Stored results

In addition to the results stored by `bayesmh`, the `bayes` prefix stores the following in `e()`:

Scalars

`e(priorsigma)`	standard deviation of default normal priors
`e(priorshape)`	shape of default inverse-gamma priors
`e(priorscale)`	scale of default inverse-gamma priors
`e(blocksize)`	maximum size for blocks of model parameters

Macros

`e(prefix)`	`bayes`
`e(cmdname)`	command name from *estimation_command*
`e(cmd)`	same as `e(cmdname)`
`e(command)`	estimation command line

Methods and formulas

See *Methods and formulas* in [BAYES] **bayesmh**.

References

Balov, N. 2017. Bayesian logistic regression with Cauchy priors using the bayes prefix. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2017/09/08/bayesian-logistic-regression-with-cauchy-priors-using-the-bayes-prefix/.

———. 2020. Bayesian inference using multiple Markov chains. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2020/02/24/bayesian-inference-using-multiple-markov-chains/.

Rabe-Hesketh, S., and A. Skrondal. 2022. *Multilevel and Longitudinal Modeling Using Stata*. 4th ed. College Station, TX: Stata Press.

Also see

[BAYES] **Bayesian estimation** — Bayesian estimation commands

[BAYES] **bayesmh** — Bayesian models using Metropolis–Hastings algorithm

[BAYES] **bayesselect** — Bayesian variable selection for linear regression

[BAYES] **Bayesian postestimation** — Postestimation tools after Bayesian estimation

[BAYES] **Bayesian commands** — Introduction to commands for Bayesian analysis

[BAYES] **Intro** — Introduction to Bayesian analysis

[BAYES] **Glossary**

[U] 20 Estimation and postestimation commands