Title

> **bayesmh postestimation** — Postestimation tools for bayesmh

[Postestimation commands](#)   [Remarks and examples](#)   [Also see](#)

Postestimation commands

The following postestimation commands are available after `bayesmh`:

Command	Description
bayesgraph	graphical summaries and convergence diagnostics
bayesstats ess	effective sample sizes and related statistics
bayesstats summary	Bayesian summary statistics for model parameters and their functions
bayesstats ic	Bayesian information criteria and Bayes factors
bayestest model	hypothesis testing using model posterior probabilities
bayestest interval	interval hypothesis testing
* estimates	cataloging estimation results

* `estimates table` and `estimates stats` are not appropriate with `bayesmh` estimation results.

Remarks and examples

Remarks are presented under the following headings:

> *Different ways of specifying model parameters*
> *Specifying functions of model parameters*
> *Storing estimation results after bayesmh*

After estimation, you can use `bayesgraph` to check convergence of MCMC visually. Once convergence is established, you can use `bayesstats summary` to obtain Bayesian summaries such as posterior means and standard deviations of model parameters and functions of model parameters; `bayesstats ess` to compute effective sample sizes and related statistics for model parameters and functions of model parameters; and `bayesstats ic` to compute Bayesian information criteria and Bayes factors for model parameters and their functions. You can use `bayestest model` to test hypotheses by comparing posterior probabilities of models. You can also use `bayestest interval` to test interval hypotheses about parameters and functions of parameters.

For an overview example of postestimation commands, see *Overview example* in [BAYES] **bayes**.

Different ways of specifying model parameters

Many `bayesmh` postestimation commands such as `bayesstats summary` and `bayesgraph` allow you to specify model parameters for which you want to see the results. To see results for all parameters, simply type a postestimation command without arguments after `bayesmh` estimation, for example,

    . bayesstats summary

or you could type

    . bayesstats summary _all

To manually list all model parameters, type

```
. bayesstats summary {param1} {param2} ...
```

or

```
. bayesstats summary {param1 param2} ...
```

The only exception is the `bayesgraph` command when there is more than one model parameter. In that case, `bayesgraph` requires that you either specify `_all` to request all model parameters or specify the model parameters of interest.

You can refer to a single model parameter in the same way you define parameters in the `bayesmh` command. For example, for a parameter with name `param` and no equation name, you can use `{param}`. For a parameter with name `param` and equation name `eqname`, you can use its full name `{eqname:name}`, where the equation name and the parameter name are separated with a colon. With postestimation commands, you can also omit the equation name when referring to the parameter with an equation name.

In the presence of more than one model parameter, you have several ways for referring to multiple parameters at once. If parameters have the same equation name, you can refer to all the parameters with that equation name as follows.

Suppose that you have three parameters with the same equation name `eqname`. Then the specification

```
. bayesstats summary {eqname:param1} {eqname:param2} {eqname:param3}
```

is the same as the specification

```
. bayesstats summary {eqname:}
```

or the specification

```
. bayesstats summary {eqname:param1 param2 param3}
```

The above specification is useful if we want to refer to a subset of parameters with the same equation name. For example, in the above, if we wanted to use only `param1` and `param2`, we could type

```
. bayesstats summary {eqname:param1 param2}
```

There is also a convenient way to refer to the parameters with the same name but different equation names. For example, typing

```
. bayesstats summary {eqname1:param} {eqname2:param}
```

is the same as simply typing

```
. bayesstats summary {param}
```

You can mix and match all the specifications above in one call to a postestimation command. You can also specify expressions of model parameters; see *Specifying functions of model parameters* for details.

Note that if `param` refers to a matrix model parameter, then the results will be provided for all elements of the matrix. For example, if `param` is the name of a 2×2 matrix, then typing

```
. bayesstats summary {param}
```

implies the following:

```
. bayesstats summary {param_1_1} {param_1_2} {param_2_1} {param_2_2}
```

Specifying functions of model parameters

You can use `bayesmh` postestimation commands to obtain results for functions or expressions of model parameters. Each expression must be specified in parentheses. An expression can be any Stata expression, but it may not include matrix model parameters. However, you may include individual elements of matrix model parameters. You may provide labels for your expressions.

For example, we can obtain results for the exponentiated parameter `{param}` as follows:

```
. bayesstats summary (exp({param}))
```

Note that we specified the expression in parentheses.

We can include a label, say, `myexp`, in the above by typing

```
. bayesstats summary (myexp: exp({param}))
```

We can specify multiple expressions by typing

```
. bayesstats summary (myexp: exp({param}) (sd: sqrt({var})))
```

If `param` is a matrix, we can specify expressions, including its elements, but not the matrix itself in the following:

```
. bayesstats summary (exp({param_1_1})) (exp({param_1_2})) ...
```

Storing estimation results after bayesmh

The `bayesmh` command stores various `e()` results such as scalars, macros, and matrices in memory like any other estimation command. Unlike other estimation commands, `bayesmh` also saves the resulting simulation dataset containing MCMC samples of parameters to disk. Many `bayesmh` postestimation commands such as `bayesstats summary` and `bayesstats ess` require access to this file. If you do not specify the `saving()` option with `bayesmh`, the command saves simulation results in a temporary Stata dataset. This file is being replaced with the new simulation results each time `bayesmh` is run. To save your simulation results, you must specify the `saving()` option with `bayesmh`, in which case your simulation results are saved to the specified file in the specified location and will not be overridden by the next call to `bayesmh`.

You can specify the `saving()` option during estimation by typing

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

or on replay by typing

```
. bayesmh, saving()
```

As you can with other estimation commands, you can use `estimates store` to store `bayesmh` estimation results in memory and `estimates save` to save them to disk, but you must first use the `saving()` option with `bayesmh` to save simulation data in a permanent dataset. For example, type

```
. bayesmh ..., likelihood() prior() ... saving(bmh_simdata)
. estimates store model1
```

or, after `bayesmh` estimation, type

```
. bayesmh, saving(bmh_simdata)
. estimates store model1
```

Once you create a permanent dataset, it is your responsibility to erase it after it is no longer needed. `estimates drop` and `estimates clear` will drop estimation results only from memory; they will not erase the simulation files you saved.

```
. estimates drop model1
. erase bmh_simdata.dta
```

See [R] **estimates** for more information about commands managing estimation results. `estimates table` and `estimates stats` are not appropriate after `bayesmh`.

Also see

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

[BAYES] **bayesmh evaluators** — User-defined evaluators with bayesmh

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

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

[BAYES] **Glossary**

[U] **20 Estimation and postestimation commands**