bayesstats grubin — Gelman–Rubin convergence diagnostics

Description

`bayesstats grubin` calculates Gelman–Rubin convergence diagnostics for model parameters and functions of model parameters using current Bayesian estimation results containing at least two Markov chains.

Quick start

Gelman–Rubin convergence diagnostics for all model parameters after a Bayesian regression model using four chains

```
bayes, nchains(4): regress y x1
bayesstats grubin
```

Same as above, but only for model parameters `{y:x1}` and `{sigma2}`

```
bayesstats grubin {y:x1} {sigma2}
```

Gelman–Rubin convergence diagnostics for functions of scalar model parameters

```
bayesstats grubin ({y:x1}-{y:_cons}) (sd:sqrt({sigma2}))
```

Menu

Statistics > Bayesian analysis > Gelman–Rubin convergence diagnostics

Syntax

Convergence statistics for all model parameters

```
bayesstats grubin [ , options showreffects[ (reref) ] ]

bayesstats grubin _all [ , options showreffects[ (reref) ] ]
```

Convergence statistics for selected model parameters

```
bayesstats grubin paramspec [ , options ]
```

Convergence statistics for functions of model parameters

```
bayesstats grubin exprspec [ , options ]
```

Full syntax

```
bayesstats grubin spec [spec ...] [ , options ]
```

paramspec can be one of the following:

{*eqname*:*param*} refers to a parameter *param* with equation name *eqname*;

{*eqname*:} refers to all model parameters with equation name *eqname*;

{*eqname*:*paramlist*} refers to parameters with names in *paramlist* and with equation name *eqname*; or

{*param*} refers to all parameters named *param* from all equations.

In the above, *param* can refer to a matrix name, in which case it will imply all elements of this matrix. See *Different ways of specifying model parameters* in [BAYES] **Bayesian postestimation** for examples.

exprspec is an optionally labeled expression of model parameters specified in parentheses:

(*[exprlabel:]expr*)

exprlabel is a valid Stata name, and *expr* is a scalar expression that may not contain matrix model parameters. See *Specifying functions of model parameters* in [BAYES] **Bayesian postestimation** for examples.

spec is one of *paramspec* or *exprspec*.

options	Description
sort	list parameters in descending order of their convergence statistics
skip(#)	skip every # observations from the MCMC sample; default is skip(0)
nolegend	suppress table legend
display_options	control spacing, line width, and base and empty cells

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

Options

`sort` specifies that model parameters be listed in descending order of their Gelman–Rubin convergence statistics. This option is useful for models with many parameters, such as multilevel models, to more easily identify the set of parameters with large values of convergence statistics.

`showreffects` and `showreffects(`*reref*`)` are for use after multilevel models, and they specify that the results for all or a list *reref* of random-effects parameters be provided in addition to other model parameters. By default, all random-effects parameters are excluded from the results to conserve computation time. If random-effects parameters are of interest in your study, you should use option `showreffects` to check their convergence diagnostics.

`skip(`#`)` specifies that every # observations from the MCMC sample not be used for computation. The default is `skip(0)` or to use all observations in the MCMC sample. Option `skip()` can be used to subsample or thin the chain. `skip(`#`)` is equivalent to a thinning interval of #+1. For example, if you specify `skip(1)`, corresponding to the thinning interval of 2, the command will skip every other observation in the sample and will use only observations 1, 3, 5, and so on in the computation. If you specify `skip(2)`, corresponding to the thinning interval of 3, the command will skip every 2 observations in the sample and will use only observations 1, 4, 7, and so on in the computation. `skip()` does not thin the chain in the sense of physically removing observations from the sample, as is done by, for example, `bayesmh`'s `thinning()` option. It only discards selected observations from the computation and leaves the original sample unmodified.

`nolegend` suppresses the display of the table legend, which identifies the rows of the table with the expressions they represent.

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

Remarks and examples

Remarks are presented under the following headings:

Gelman–Rubin convergence diagnostic
Using bayesstats grubin

Gelman–Rubin convergence diagnostic

The Gelman–Rubin convergence diagnostic, R_c, assesses MCMC convergence by analyzing differences between multiple Markov chains. The convergence is assessed by comparing the estimated between-chains and within-chain variances for each model parameter. Large differences between these variances indicate nonconvergence. See Gelman and Rubin (1992) and Brooks and Gelman (1998) for details.

Large values of R_c indicate nonconvergence of MCMC. Literature suggests that the values of this diagnostic should be less than 1.2 for all model parameters to declare MCMC convergence. In practice, a more stringent convergence rule, $R_c < 1.1$, is often used.

Gelman–Rubin diagnostic relies on a Student's t approximation of the marginal posterior distribution of a model parameter. When this assumption is suspect, it is recommended to transform the parameter such that its marginal posterior distribution is better approximated by a Student's t distribution before obtaining the diagnostic. For example, for the variance parameter, it is better to compute the diagnostic for the log variance.

Using bayesstats grubin

The bayesstats grubin command computes the Gelman–Rubin convergence diagnostic for each model parameter using multiple MCMC samples or chains from a common posterior model. This command requires at least two chains. Multiple chains can be obtained by using the nchains() option with Bayesian estimation commands ([BAYES] **Bayesian estimation**). When you simulate multiple chains to assess convergence, it is important to use overdispersed initial values (Gelman and Rubin 1992, Brooks and Gelman 1998). See *Specifying initial values* in [BAYES] **bayesmh** and *Initial values* in [BAYES] **bayes** for details.

When typed without arguments, the command displays results for all model parameters. Alternatively, you can specify a subset of model parameters following the command name; see *Different ways of specifying model parameters* in [BAYES] **Bayesian postestimation**. You can also obtain results for scalar functions of model parameters; see *Specifying functions of model parameters* in [BAYES] **Bayesian postestimation**. Also see example 2.

For multilevel models, similarly to other Bayesian postestimation commands, bayesstats grubin does not report convergence statistics for the random-effects parameters by default. You can use the showreffects option to see them for all random-effects parameters or the showreffects(*reref*) option for a subset *reref* of random-effects parameters of interest. See *Multilevel models* in [BAYES] **bayes** for more information about MCMC convergence in multilevel models.

For models with many parameters such as multilevel models, you can use the sort option to list model parameters in descending order of their convergence statistics Rc. The parameters with the largest values of Rc will be listed first, making it easier to verify their convergence.

▷ Example 1: Convergence diagnostics for all parameters

Recall our analysis of womenwage.dta using the bayes: regress command from example 1 in [BAYES] **bayes**. We fit a linear regression model to the response variable wage with predictor age. Here we use option nchains(3) to simulate three Markov chains to formally check convergence of model parameters. To ensure reproducibility of multiple chains, we also specify the rseed(15) option. Specifying set seed is not sufficient for reproducibility with multiple chains; see *Reproducing results* in [BAYES] **bayesmh** for details.

```
. use https://www.stata-press.com/data/r19/womenwage
(Wages of women)
. bayes, nchains(3) rseed(15): regress wage age
Chain 1
  Burn-in ...
  Simulation ...
Chain 2
  Burn-in ...
  Simulation ...
Chain 3
  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                    Number of chains    =          3
Random-walk Metropolis-Hastings sampling      Per MCMC chain:
                                                   Iterations     =     12,500
                                                   Burn-in        =      2,500
                                                   Sample size    =     10,000
                                              Number of obs       =        488
                                              Avg acceptance rate =      .3673
                                              Avg efficiency: min =      .1409
                                                              avg =      .1735
                                                              max =      .2294
Avg log marginal-likelihood = -1810.1557      Max Gelman-Rubin Rc =          1
```

	Mean	Std. dev.	MCSE	Median	Equal-tailed [95% cred. interval]	
wage						
age	.4003528	.0599411	.000922	.4002037	.2804134	.5188627
_cons	5.999502	1.769855	.026358	6.025288	2.571305	9.517341
sigma2	90.80977	5.822896	.070195	90.49567	79.92114	102.7621

```
Note: Default priors are used for model parameters.
Note: Default initial values are used for multiple chains.
```

Compared with example 1 in [BAYES] **bayes**, the precision of the posterior means almost doubled with more chains, judging by the MCMC standard errors. For example, the MCSE estimate for `{sigma2}` drops from 0.12 to 0.07.

In the presence of multiple chains, the `bayes` prefix automatically reports in the header the maximum value of the Gelman–Rubin convergence statistics across all parameters. In practice, we want to see this value be close to 1; if it is less than 1.1, the chains are considered to have converged. This convergence rule is satisfied in our example.

To compute the Gelman–Rubin statistics for all model parameters, we type `bayesstats grubin` without arguments after the `bayes` prefix.

```
. bayesstats grubin
Gelman-Rubin convergence diagnostic
Number of chains     =            3
MCMC size, per chain =       10,000
Max Gelman-Rubin Rc  =     1.000323
------------------------------------
                |          Rc
----------------+-------------------
wage            |
          age   |    1.000062
        _cons   |    1.000323
----------------+-------------------
       sigma2   |    1.000253
------------------------------------
Convergence rule: Rc < 1.1
```

Just like the `bayes` prefix, the `bayesstats grubin` command reports in the header the maximum value of `Rc` across all parameters. This is particularly useful as a quick convergence check for models with many parameters: if the maximum `Rc` is less than 1.2 or 1.1, then this convergence rule is satisfied by all parameters. In our example, the maximum `Rc` is 1.0003 and is less than 1.1, so the convergence criterion is met for all parameters.

The table reports the `Rc` estimates for each model parameter. As we already determined based on the maximum `Rc`, the convergence diagnostics for all model parameters are less than 1.1. This suggests that all chains have converged.

◁

▷ Example 2: Convergence diagnostics for functions of parameters

Continuing with example 1, we can compute the Gelman–Rubin statistics for functions of parameters. Let's compute the convergence diagnostic for the log-transformed variance parameter `{sigma2}`.

```
. bayesstats grubin (lnsigma2: ln({sigma2}))
Gelman-Rubin convergence diagnostic
Number of chains     =            3
MCMC size, per chain =       10,000
Max Gelman-Rubin Rc  =     1.000268
    lnsigma2 : ln({sigma2})
------------------------------------
                |          Rc
----------------+-------------------
    lnsigma2    |    1.000268
------------------------------------
Convergence rule: Rc < 1.1
```

Again, the convergence diagnostic for the log-transformed variance is less than 1.1 indicating no convergence problems with the transformed parameter. This also suggests that `{sigma2}` does not have convergence problems.

◁

In our examples, we used the default initial values provided by `bayes:` with multiple chains; see *Initial values* in [BAYES] **bayes**. To fully explore MCMC convergence, particularly when a posterior distribution is suspected to have multiple modes, you should use overdispersed initial values. See *Multiple chains using overdispersed initial values* in [BAYES] **bayesmh** for an example of how to specify overdispersed initial values.

Of course, it is important to explore convergence visually as well; see *Convergence diagnostics using multiple chains* in [BAYES] **bayesmh**.

Stored results

`bayesstats grubin` stores the following in `r()`:

Scalars

`r(mcmcsize)`	MCMC sample size of each chain
`r(nchains)`	number of MCMC chains
`r(Rc_max)`	maximum convergence diagnostic

Matrices

`r(Rc)`	convergence diagnostics `Rc`
`r(t_df)`	degrees of freedom of a t distribution
`r(B)`	between-chains variances
`r(W)`	within-chain variances
`r(V)`	total variances

Methods and formulas

Suppose we have M chains of length T. For a model parameter θ, let $\{\theta_{jt}\}_{t=1}^{T}$ be the jth simulated chain drawn from the marginal posterior distribution of θ, $j = 1, \ldots, M$. Let $\hat{\theta}_j$ and $\hat{s}_j^2$ be the respective sample posterior mean and variance of the mth chain, and let the overall sample posterior mean be $\hat{\theta} = (1/M)\sum_{j=1}^{M}\hat{\theta}_j$. The between-chains and within-chain variances are given by

$$B = \frac{T}{M-1}\sum_{j=1}^{M}(\hat{\theta}_j - \hat{\theta})^2$$

$$W = \frac{1}{M}\sum_{j=1}^{M}\hat{s}_j^2$$

When the chains are strongly stationary, that is, all chains draw samples from the target posterior distribution, the weighted average of W and B

$$\hat{\sigma}^2 = \frac{T-1}{T}W + \frac{1}{T}B$$

is an unbiased estimator of the marginal posterior variance of θ.

Gelman and Rubin (1992) approximate the target distribution of θ by a Student's t distribution with mean $\hat{\theta}$ and scale $\sqrt{\hat{V}}$, where

$$\hat{V} = \frac{T-1}{T}W + \frac{M+1}{MT}B$$

They define the so-called "scale" reduction factor as the ratio of $\hat{V}$ and $\sigma^2 = \text{Var}(\theta)$. They further estimate σ^2 by W and use the ratio of $\hat{V}$ and W as an estimator of the scale reduction factor, known as the potential scale reduction factor. If the M chains have converged to the target posterior distribution, then the potential scale reduction factor should be close to 1.

Brooks and Gelman (1998) propose the corrected estimator of the potential scale reduction factor, R_c, that accounts for sampling variability:

$$R_c = \sqrt{\frac{\hat{d}+3}{\hat{d}+1}\frac{\hat{V}}{W}}$$

where $\hat{d}$ is the estimated degrees of freedom of the approximating Student's t distribution for θ

$$\hat{d} = \frac{2\hat{V}^2}{\widehat{\text{Var}}(\hat{V})}$$

and

$$\widehat{\text{Var}}(\hat{V}) = \left(\frac{T-1}{T}\right)^2 \frac{1}{M}\widehat{\text{Var}}(\hat{s}_j^2) + \left(\frac{M+1}{MT}\right)^2 \frac{2}{M-1}B^2 + 2\frac{(M+1)(T-1)}{M^2T}\left\{\widehat{\text{Cov}}(\hat{s}_j^2, \hat{\theta}_j^2) - 2\hat{\theta}\,\widehat{\text{Cov}}(\hat{s}_j^2, \hat{\theta}_j)\right\}$$

$\widehat{\text{Var}}(\hat{s}_j^2)$ is the sample variance of $\hat{s}_j^2$'s, $j = 1, \ldots, M$. $\widehat{\text{Cov}}(\hat{s}_j^2, \hat{\theta}_j^2)$ and $\widehat{\text{Cov}}(\hat{s}_j^2, \hat{\theta}_j)$ are the sample covariances of $\hat{s}_j^2$'s and $\hat{\theta}_j^2$'s and $\hat{s}_j^2$'s and $\hat{\theta}_j$'s, respectively.

Brooks and Gelman (1998) suggested to use the criterion $R_c < 1.2$ for all model parameters to declare MCMC convergence. In practice, a more stringent convergence criterion, $R_c < 1.1$, is often used. If a convergence criterion is not met, longer chains or other means for improving the convergence are needed.

References

Balov, N. 2016. Gelman–Rubin convergence diagnostic using multiple chains. *The Stata Blog: Not Elsewhere Classified.* https://blog.stata.com/2016/05/26/gelman-rubin-convergence-diagnostic-using-multiple-chains/.

———. 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/.

Brooks, S. P., and A. Gelman. 1998. General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics* 7: 434–455. https://doi.org/10.1080/10618600.1998.10474787.

Gelman, A., and D. B. Rubin. 1992. Inference from iterative simulation using multiple sequences. *Statistical Science* 7: 457–472. https://doi.org/10.1214/ss/1177011136.

Also see

[BAYES] **bayes** — Bayesian regression models using the bayes prefix

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

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

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

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

[BAYES] **bayesstats summary** — Bayesian summary statistics