

# Bayesian software validation

Samantha Cook and Andrew Gelman

*BayesValidate* is a package for testing Bayesian model-fitting software. Generating a sample from the posterior distribution of a Bayesian model often involves complex computational algorithms that are programmed “from scratch.” Errors in these programs can be difficult to detect, because the correct output is not known ahead of time; not all errors lead to crashes or results that are obviously incorrect. Software is often tested by applying it to data sets where the “right answer” is known or approximately known. Cook et al. (2006) extend this strategy to develop statistical assessments of the correctness of Bayesian model-fitting software; this method is implemented in *BayesValidate*. Generally, the validation method involves simulating “true” parameter values from the prior distribution, simulating fake data from the model, performing inference on the fake data, and comparing these inferences to the “true” values. Geweke (2004) presents an alternative simulation-based method for testing Bayesian software.

More specifically, let  $\theta^{(0)}$  represent the “true” parameter value drawn from the prior distribution  $p(\theta)$ . Data  $y$  are drawn from  $p(y|\theta^{(0)})$ , and the posterior sample of size  $L$  to be used for inference,  $\theta^{(1)}, \dots, \theta^{(L)}$ , is drawn using the to-be-tested software. With this sampling scheme,  $\theta^{(0)}$  as well as  $\theta^{(1)}, \dots, \theta^{(L)}$  are, in theory, draws from  $p(\theta|y)$ . If the Bayesian software works correctly, then,  $\theta^{(0)}$  should look like a random draw from the empirical distribution  $\theta^{(1)}, \dots, \theta^{(L)}$ , and therefore the (empirical) posterior quantile of  $\theta^{(0)}$  with respect to  $\theta^{(1)}, \dots, \theta^{(L)}$  should follow a Uniform(0, 1) distribution. Testing the software amounts to testing that the posterior quantiles for scalar parameters of interest are in fact uniformly distributed.

One “replication” of the validation simulation consists of: 1) Generating parameters and data; 2) Generating a sample from the posterior distribution; and 3) calculating posterior quantiles. Performing many replications creates, for each scalar parameter whose posterior distribution is generated by the model-fitting software, a collection of quantiles whose distribution will be uniform if the software works correctly. If  $N_{rep}$  is the number of replications and  $q_1, q_2, \dots, q_{N_{rep}}$  are the quantiles for a scalar parameter, the quantity  $\sum_{i=1}^{N_{rep}} (\Phi^{-1}(q_i))^2$  will follow a  $\chi_{N_{rep}}^2$  distribution if the software works correctly, where  $\Phi^{-1}$  represents the inverse normal cumulative distribution function (CDF). For each scalar parameter, a p-value is then obtained by comparing the sum of the transformed quantiles with the  $\chi_{N_{rep}}^2$  distribu-

tion. *BayesValidate* analyzes each scalar parameter separately, but also creates combined summaries for each vector parameter; these scalar results and summaries are in the graphical output as well.

*BayesValidate* performs a specified number of replications and calculates a p-value for each scalar parameter. The function returns a Bonferroni-adjusted p-value and a graphical display of the  $z_\theta$  statistics, which are the inverse normal CDFs of the p-values. Figures 1 and 2 show the graphical output for two versions of a program written to fit a simple hierarchical normal model with parameters  $\sigma^2$ ,  $\tau^2$ ,  $\mu$ , and  $\alpha_1, \alpha_2, \dots, \alpha_6$ ; one version correctly samples from the posterior distribution and one has an error.

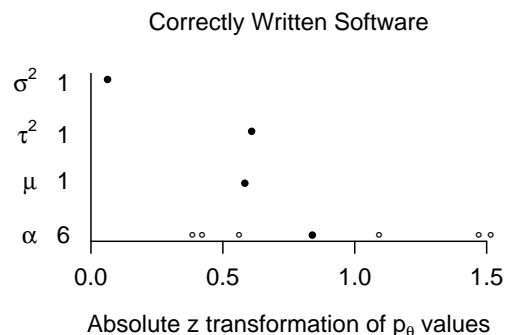


Figure 1:  $z_\theta$  statistics: Correctly written software. Each row represents a scalar parameter or batch of parameters; the circles in each row represent the  $z_\theta$  statistics associated with that parameter or batch of parameters. Solid circles represent the  $z_\theta$  statistics associated with the mean of that batch of parameters. The numbers on the y axis indicate the number of parameters in the batch. The  $z_\theta$  statistics are all within the expected range for standard normal random variables.

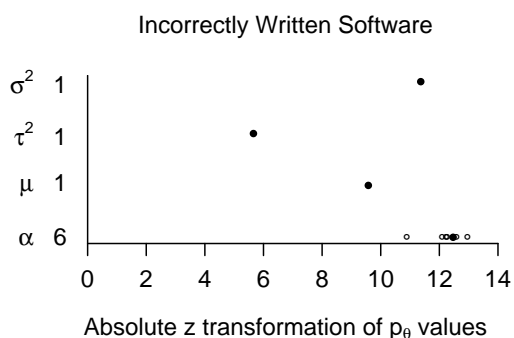


Figure 2:  $z_\theta$  statistics: *Incorrectly written software* (error sampling the parameter  $\alpha$ ). Each row represents a scalar parameter or batch of parameters; the circles in each row represent the  $z_\theta$  statistics associated with that parameter or batch of parameters. Solid circles represent the  $z_\theta$  statistics associated with the mean of that batch of parameters. The numbers on the y axis indicate the number of parameters in the batch. Values of  $z_\theta$  larger than 2 indicate a potential problem with the software; this plot provides convincing evidence that the software has an error.

## Bibliography

- S. Cook, A. Gelman, and D. B. Rubin. Validation of software for Bayesian models using posterior quantiles. *Journal of Computational and Graphical Statistics*, 2006. To appear.
- J. Geweke. Getting It Right: Joint Distribution Tests of Posterior Simulators. *Journal of the American Statistical Association*, 99:799–804, 2004.