Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

A Linear Regression Framework for the Verification of Bayesian Model Calibration Algorithms

Jerry A. McMahan Jr.
Research Associate
Digimarc Corporation
Beaverton, Oregon 97008
Email: jerry.mcmahan@digimarc.com

Brian J. Williams
Scientist
Statistical Sciences Group (CCS-6)
Los Alamos National Laboratory
Los Alamos, New Mexico 87545
Email: brianw@lanl.gov

Ralph C. Smith
Distinguished Professor, Member of ASME
Department of Mathematics
North Carolina State University
Raleigh, North Carolina 27695
Email: rsmith@ncsu.edu

Nicholas Malaya
Doctoral Candidate, Student Member of ASME
Department of Mechanical Engineering
The University of Texas at Austin
Austin, Texas 78712
Email: nick@ices.utexas.edu

ABSTRACT

We describe a framework for the verification of Bayesian model calibration routines. The framework is based on linear regression and can be configured to verify calibration to data with a range of observation error characteristics. The framework is designed for efficient implementation and is suitable for verifying code intended for large-scale problems. We propose an approach for using the framework to verify Markov chain Monte Carlo (MCMC) software by combining it with a non-parametric test for distribution equality based on the energy statistic. Our MATLAB-based reference implementation of the framework is shown to correctly distinguish between output obtained from correctly and incorrectly implemented MCMC routines. Since correctness of output from an MCMC software depends on choosing settings appropriate for the problem-of-interest, our framework can potentially be used for verifying such settings.

1 Introduction

The growth of interest in uncertainty quantification research is motivated by the potential to improve the practical relevance of computational mathematics to applications in engineering, science, and industry. The breadth of applications

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

found as examples in a small sample of the literature supports this notion. Applications exist for optimal control theory [1,2], acoustic and electromagnetic scattering [3–5], smart material systems [6,7], fluid dynamics [8–12], aircraft design [13–15], physiology and disease modeling [16–19], geology [20–22], and nondestructive evaluation [23–25]. Broadly speaking, uncertainty quantification is concerned with characterizing variability in the outputs of mathematical models arising from uncertainties in the inputs, typically through statistical approaches which facilitate comparison to measurement. Improved understanding of model variability and its relationship to measurement uncertainty translates to improved predictability when applying a computational model to a real-world phenemenon.

Although the concern for understanding variability in mathematical models is a unifying theme, the field of uncertainty quantification encompasses a diverse range of topics, as the recent collection of papers in [26] demonstrates. For broad introductions to the field as a whole we refer the reader to [27] and [28]. In this paper, our focus is on Bayesian formulations for model calibration. A general introduction to Bayesian methods in scientific computing can be found in [29]. A useful introduction to Bayesian methods in the context of model calibration is available in [30]. For a treatment of the mathematical foundations underlying these methods, see [31]. Accepting the premise that interest in uncertainty quantification is largely motivated by its benefits to practical applications of computational models, then the sub-field of model calibration is particularly important. Model calibration quantifies the relationship between observations of a phenomenon and the parameters of the chosen mathematical model. Bayesian techniques provide a rigorous, data-driven approach to parameter inference. They are a natural and robust methodology for uncertainty quantification, as they express calibration results as a probability distribution instead of a point-estimate. Bayesian inference therefore forms a key component of the link between measurement and abstract mathematical model. The reliability of these computational techniques is critical to ensuring the methods of uncertainty quantification fulfill their potential to bolster the contributions of computational engineering in practice.

However useful an algorithm may be, it is only as reliable as its implementation. This naturally leads to the computational engineering research topic of verification. For a general introduction to verification, see [32]. A review of verification research for general computational simulation is available in [33] and [34] reviews verification in computational fluid dynamics context. Verification is a necessary first step for uncertainty quantification. Whereas uncertainty quantification generally deals with the identification of variabilities in models and how they propagate, verification involves approaches to testing the implementation of a computational algorithm to ensure the implementation correctly performs the intended computations. In particular, verification is designed to ensure that the outputs of a computational model accurately reflect the solution of the underlying mathematical model. Tests are designed to estimate the errors from the numerical discretization scheme, as well as to check that the convergence rate (as a function of mesh size or timestep) observed in the software output matches that of the implemented numerical scheme.

One of the simplest methods of code verification is to test the numerical output of a code against an exact, analytical solution. This approach is of limited utility, as typically analytical solutions do not exist for practical problems. Another method is to compare simulation output from the implementation being verified against simulation output from a reference solution. This method is also of limited use, as it is requires availability of an already verified implementation.

A particularly flexible approach to verification is the method of manufactured solutions (MMS) which applies to numerical algorithms for solving nearly all deterministic mathematical equations. MMS assumes a parameterized functional form for the solution that leaves an analytically computable residual when inserted into the equation to be solved. The residual is used to derive inputs for which the functional form does solve the equation of interest. This is a powerful approach because it provides the same level of convergence testing available from an analytical solution, while simultaneously not requiring an analytical solution to exist. MMS typically requires the support of computer algebra systems, due to the complexity of evaluating the residual terms by hand. For a thorough treatment of MMS, including a study of the types of errors that MMS can and cannot find, see [35]. The verification library Manufactured Analytical Solution Abstraction (MASA) library is based on MMS [36], and acts as a repository of solutions generated by this approach.

One more verification approach is the method of nearby problems described in [37]. Here analytical curves are fit to numerical solutions of an equation of interest. These curves are used with the original equation to derive analytical input terms, similar to the approach of MMS. These input terms are used to define a problem that is "nearby" the original equation but whose discretization error can be calculated precisely due to the analytical input terms. The advantage of this over MMS is that it provides a means of defining physically realistic verification routines.

Regretably, all of the approaches outlined are formulated for deterministic problems. Stochastic algorithms present several additional difficulties. Most significant is the fact that the error introduced from random inputs in the system no longer guarantees monotonic convergence under refinement. Furthermore, the non-reproducability of results across different system architectures and random number generators adds additional complexity to the verification process. For these reasons, there are very few existing frameworks for the robust verification of statistical software. Simultaneously, due to inherent randomness in the output of stochastic algorithms, verification is arguably even more important than for deterministic systems. For example, Bayesian model calibration involves computing the density weighting the parameterization of a large ensemble of solutions to an equation. Such complicated results are difficult to assess by inspection.

Currently, very few algorithms presently exist for the verification of stochastic algorithms in general, and the authors are not aware of any designed for use in a Bayesian calibration framework. The objective of this paper is therefore to present a

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

step towards the verification of Bayesian model calibration algorithms. We outline a verification framework based on linear regression, a version of which was first described in [38]. This framework provides analytically derived formulas for the parameter densities obtained via Bayes rule that can be compared to numerical results. Although we employ a linear model which we describe a matrix with a specific class of components for concreteness, the results do not depend on this specific form so the regression matrix can be designed to approximate some types of non-linear functions. The framework is based on widely known statistical concepts, relatively simple to implement, and computationally efficient but has flexibility in the choice of error statistics to ensure the calibration algorithm is exercised over a range of possible inputs.

A major motivation for the framework is the verification of Markov chain Monte Carlo (MCMC) routines for Bayesian inference. A brief introduction to MCMC methods, the mathematical theory underlying them, and their use in a variety of scientific fields is [39]. Rather than directly computing posterior distributions for the calibration parameters, MCMC works by iteratively computing random values in a way that guarantees the limiting distribution of the collective samples will converge to that of the posterior. Properties of the posterior can be inferred from the resulting sample. Several types of MCMC algorithms exist. See [40] for an overview of the classic Metropolis-Hastings algorithm. Variants of the Metropolis-Hastings algorithm designed to improve the sampling efficiency include the Delayed-Rejection Adaptive Metropolis (DRAM) algorithm [41], which automatically tunes the way the random values of the Markov chain are chosen, and the DiffeRential Evolution Adaptive Metropolis (DREAM) algorithm [42], which is designed for parallelization. Hybrid Monte Carlo or Hamiltonian Monte Carlo methods are another variety of algorithms that employ concepts inspired by Hamiltonian dynamics to improve sampling efficiency [43].

When the posterior is a known parametric distribution, there are standard hypothesis tests which can be used to verify the expected distribution from data up to a given confidence level. Although for many configurations of our framework the posteriors are well-known distributions, some are not standard and must be evaluated numerically. For this reason, we employ a non-parametric hypothesis test for equal distributions based on the energy statistic, developed in [44]. This method provides a way to verify any random sample (in our case generated by the MCMC algorithm), provided that sample satisfies certain reasonable assumptions and a separate sample drawn from the true distribution is available (in our case drawn from the known posterior distribution using either standard routines or with the inverse cumulative distribution map).

An alternative approach to verifying that the MCMC output matches a known distribution would be to employ hypothesis tests for comparing moments. One might consider using the classical Hotelling T^2 test to compare the means of two sample populations, one drawn from the MCMC algorithm and the other from the known distribution. Comparison tests designed to be highly scalable like [45] for testing the equality of means for a large number of regression parameters or [46] for comparing covariance matrices in high-dimensions could also be considered. These would apply to a subset of the problems defined by our framework which are normal or are approximately normal due to the central limit theorem. These approaches might require or benefit from transformations designed to better approximate normality (e.g., taking the logarithm of the precision parameter λ defined in the model descriptions below). Another alternative would be to employ kernel density estimate methods to estimate the distribution of the MCMC output and compare this to the known distribution with some metric (e.g., the norm of the difference). Such an approach would not require normally distributed or approximately normally distributed variables, but would require a way of quantifying the errors in the density estimate and incorporating them into thresholds for the comparison metric used to assess the similarity in distribution.

The rest of the paper is organized as follows. In Section 2 we describe the linear model calibration verification framework, including a description of the model, its parameters, and its statistical properties. In Section 3 we discuss the numerical implementation of the framework, describing some computationally efficient approaches to calculating the posterior densities and generating samples from the error processes used. Section 4 provides formulas for the exact posterior densities which are the heart of the verification framework. Derivations for the simplest case are included to make the paper somewhat more self-contained. Section 5 describes the energy statistic, the hypothesis test for equal distributions based on it, and details of how to use it with the posterior densities provided by the framework to verify MCMC output. In Section 6 we apply our MATLAB-based reference implementation of the framework to the verification of MCMC output computed using the Delayed Rejection Adaptive Metropolis (DRAM) MATLAB package in three separate studies. The final section concludes the paper with some additional suggestions of how to apply the framework.

2 Linear Statistical Model Calibration Verification Framework

At the heart of the verification framework is the linear regression model, a ubiquitous tool in applied statistics. We summarize the model here and refer the reader to [27,47–49] for more details. The equation for the model in our framework is

$$y = G\beta + \varepsilon(\lambda, \phi). \tag{1}$$

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

Calibration Case	Calibrated	Known
1	$\hat{\beta}$	λ, ϕ
2	$\hat{\beta}, \hat{\lambda}$	ϕ
3	$\hat{\beta}, \hat{\lambda}, \hat{\phi}$	-

Table 1. The three calibration cases for the verification framework.

Here $y = \begin{bmatrix} y_1 \cdots y_N \end{bmatrix}^T$ is the N-dimensional vector of noisy observations of the model response. The $N \times N_\beta$ regression matrix

$$G = \begin{bmatrix} 1 & x_{1,1} & \cdots & x_{1,N_\beta-1} \\ 1 & x_{2,1} & \cdots & x_{2,N_\beta-1} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{N,1} & \cdots & x_{N,N_\beta-1} \end{bmatrix}$$

is known and the N_β components of the vector β are unknown parameters to be estimated. Each $(N_\beta - 1)$-dimensional covariate vector $x_j = \begin{bmatrix} x_{j,1}\ x_{j,2} \cdots x_{j,N_\beta-1} \end{bmatrix}^T$ is a sample from a zero-mean multivariate normal distribution with known $(N_\beta - 1) \times (N_\beta - 1)$ covariance matrix C. The observation error ε is an N-dimensional, mean-zero, normally distributed random variable with $N \times N$ covariance matrix $R(\phi)/\lambda$. The notation $\varepsilon = \varepsilon(\lambda, \phi)$ signifies the dependence of the distribution of the observation error on the hyperparameters λ and ϕ. The error precision parameter λ is a positive scalar and the admissible values of the correlation structural parameter ϕ depend on the observation error correlation type discussed in Section 2.4.

The choice of covariates in G is an example that admits a well-known formula for calculating a predictive distribution for new inputs (see, e.g., the review of regression in [50] for a derivation), but is not at all a constraint the framework relies on as any correctly sized matrix will do. It is possible to replace the rows of G with regression functions $g(x)$ which map a p-dimensional vector of inputs $x \in \mathbb{R}^p$ to a N_β-dimensional row vector. For example, $g(x)$ could be a polynomial, a sinusoidal function, a spline function, or any other basis function commonly used in numerical methods. This would enable the approximation of more complicated functions, including nonlinear functions, with no change to the the remainder of the framework.

2.1 Calibration Cases

To generate data to be used for calibration, β, λ, and ϕ are set to some true values and N samples of data are generated from the model defined by the equation in (1). The associated calibration problem is to treat some choice of parameters as unknown and determine the distribution of the parameter values from this generated data using Bayesian inference. Three cases of unknown variables are considered in this framework, as summarized in Table 1. Each of these cases are very common statistical inference problems covered in standard texts on regression such as [51] and very generally applicable. They are ordered by the difficulty of the inference task, with subsequent cases inferring more properties of the model from the observation data. Case 1 estimates the linear parameters assuming the statistics of the observation error are perfectly characterized. This is the simplest case of linear regression, applicable to situations where the observation or measurement error is very well-characterized both in terms of its structure and magnitude. Case 2 removes explicit knowledge of the amount of variance in the observation error samples. This is a more complex model where the structure of the observation error is known ahead of time but its magnitude is inferred from data. In the case of normally distributed observation error which we use, this assumption results in posterior distributions that are no longer Gaussian. Case 3 assumes in addition that a model of the correlation structure in the observation error is known but a parameter characterizing the amount of correlation is unknown.

2.2 Likelihood Function

The model equation for generating observations in the absence of observation error is

$$\hat{y} = G\hat{\beta},$$

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

Calibration Case	Non-informative	Gaussian
1	$\pi_0(\hat{\beta}) \propto 1$	$\pi_0(\hat{\beta}) \propto \exp\left[-\tilde{\beta}^T (\Sigma_0/\lambda)^{-1} \tilde{\beta}/2\right]$
2	$\pi_0(\hat{\beta},\hat{\lambda}) \propto 1/\hat{\lambda}$	$\pi_0(\hat{\beta},\hat{\lambda}) \propto \hat{\lambda}^{N_\beta/2-1} \exp\left[-\tilde{\beta}^T \left(\Sigma_0/\hat{\lambda}\right)^{-1} \tilde{\beta}/2\right]$
3	$\pi_0(\hat{\beta},\hat{\lambda},\hat{\phi}) \propto 1/\hat{\lambda}$	$\pi_0(\hat{\beta},\hat{\lambda},\hat{\phi}) \propto \hat{\lambda}^{N_\beta/2-1} \exp\left[-\tilde{\beta}^T \left(\Sigma_0/\hat{\lambda}\right)^{-1} \tilde{\beta}/2\right]$

Table 2. The priors for each calibration case in the verification framework where $\tilde{\beta} = \hat{\beta} - \beta_0$. Note the explicit appearance of $\hat{\lambda}$ when λ is unknown.

where the hats indicate estimates of the corresponding variables. The residual between the model and the data is denoted $\tilde{y} = y - \hat{y}$ and the likelihood function used in calibration is proportional to

$$\frac{\hat{\lambda}^{N/2}}{\left|R(\hat{\phi})\right|^{1/2}} \exp\left[-\frac{1}{2}\tilde{y}^T \left(\frac{R(\hat{\phi})}{\hat{\lambda}}\right)^{-1} \tilde{y}\right]$$

which is a consequence of the assumption of normally distributed observation error. For cases where the true value of λ is known, $\hat{\lambda} = \lambda$ is used for computing the likelihood and similarly for ϕ.

2.3 Prior Cases

Two cases of prior distribution for β are considered in the verification framework: the non-informative prior case and the Gaussian prior case. A non-informative prior case indicates a belief that all the unknown parameters are equally likely to lie anywhere in their respective parameter domains. This is a reasonable assumption to use for models of poorly understood problems where little is known about the expected character of the solution. A Gaussian prior case indicates a belief that the true parameters for β are likely to be near some expected value in the domain with the likelihood decaying exponentially as a function of the distance from this expected value. This is a reasonable prior to use on problems where some information is known about the solution ahead of time, for example, from historical data. Note for calibration cases where λ and ϕ are unknown, their prior distributions are treated as non-informative, although as will be clear from the expressions for the exact posterior, more general choices are possible.

The Gaussian prior for β has N_β-dimensional mean vector β_0 and diagonal $N_\beta \times N_\beta$ covariance matrix

$$\frac{1}{\lambda}\Sigma_0 = \frac{1}{\lambda}\begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_{N_\beta}^2 \end{bmatrix}, \tag{2}$$

where $\sigma_1, \ldots, \sigma_{N_\beta}$ are positive parameters. Note that when λ is unknown and calibrated, the λ in (2) is replaced with the calibrated variable $\hat{\lambda}$ rather than the true (and unknown) value λ. The non-informative prior for β is simply a uniform density $\pi(\beta) \propto 1$. The use of a diagonal Σ_0 is for convenience as it requires the selection of fewer parameters. Any symmetric positive definite matrix will work just as well.

The non-informative priors for λ and ϕ are, respectively, the Jeffrey's non-informative prior $\pi(\lambda) \propto 1/\lambda$ and the uniform prior $\pi(\phi) \propto 1$. When $\hat{\phi}$ is calibrated, it is restricted to a closed interval that is a proper subset of its domain. The priors for each calibration case are listed in Table 2.

2.4 Correlation Functions

The covariance matrices $R(\phi)/\lambda$ for the observation error are determined from a choice of correlation functions $R(\phi)$. The three correlation functions considered and the corresponding domains of ϕ are:

1. No correlation, no ϕ dependence.

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

2. Equicorrelation, $0 < \phi < 1$.
3. AR(1), i.e., order 1 autoregressive correlation, $-1 < \phi < 1$.

No correlation represents the basic assumption of the simplest form of ordinary least squares in which the phenomenon is assumed to follow a linear trend with independent, identically distributed errors. AR(1) models are utilized in time series analysis that is broadly applicable and common in, e.g., econometrics and climate models [52]. Equicorrelation is another common and general model with applications in finance and econometrics, e.g., [53, 54].

The correlation matrices for each of these cases are

$$R_1(\phi) = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix},$$

$$R_2(\phi) = \begin{bmatrix} 1 & \phi & \cdots & \phi \\ \phi & 1 & \cdots & \phi \\ \vdots & \vdots & \ddots & \vdots \\ \phi & \phi & \cdots & 1 \end{bmatrix},$$

$$R_3(\phi) = \begin{bmatrix} 1 & \phi & \phi^2 & \cdots & \phi^{N-1} \\ \phi & 1 & \phi & \cdots & \phi^{N-2} \\ \phi^2 & \phi & 1 & \cdots & \phi^{N-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi^{N-1} & \phi^{N-2} & \phi^{N-3} & \cdots & 1 \end{bmatrix},$$

where R_1 is uncorrelated error, R_2 is equally correlated error, and R_3 is AR(1) correlated error. The uncorrelated case has no correlation between output observations and no dependence on ϕ. Equicorrelation makes the correlation between all observations equal with ϕ determining the amount of correlation. For AR(1) correlation, any two output observations y_i, y_j become less correlated the further apart they are in index; i.e., as $|i - j|$ increases. From a numerical perspective, the role of the equicorrelation and AR(1) correlation functions in this framework is to make the inference task more challenging as values of ϕ nearer to 1 are more poorly conditioned.

3 Numerical Implementation

While it is possible to use any correlation function for the observation errors, a major advantage of using the correlation functions described in Section 2.4 is that they allow the calibration and exact posteriors to be computed more efficiently. The discussion in Section 3.1 through Section 3.3 summarizes these benefits.

3.1 Realizations of observation error

To generate a realization of an N-dimensional mean-zero normally distributed random variable with covariance matrix C, the Cholesky factorization $C = LL^T$ of C is computed and the resulting Cholesky factor is applied to a vector x of N samples from a standard normal distribution. The vector $y = Lx$ is then the desired realization. For a general covariance matrix C, the Cholesky factorization requires $O(N^2)$ arithmetic operations. Storing C in memory also requires $O(N^2)$ units of memory. These constraints are prohibitive for large sample sizes, N. For the covariance matrices corresponding to the correlation functions in Section 2.4, these constraints can both be reduced to $O(N)$. To simplify notation, the algorithms below do not include the multiplicative factor $1/\sqrt{\lambda}$ which must be applied to the output.

For uncorrelated error, the covariance matrix is simply a scalar multiple of the identity matrix $R(\phi)/\lambda = I/\lambda$. For equicorrelated error, it can be verified that the Cholesky factor for $R_2(\phi)$ is given by

$$L_2 = \begin{bmatrix} d_1 & 0 & 0 & \cdots & 0 \\ o_1 & d_2 & 0 & \cdots & 0 \\ o_1 & o_2 & d_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ o_1 & o_2 & o_3 & \cdots & d_N \end{bmatrix},$$

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

where the entries o_i, d_i are given by the recursion

$$d_1 = 1, \qquad o_1 = \phi$$
$$d_{n+1} = \sqrt{d_n^2 - o_n^2},\ o_{n+1} = \frac{(d_n - o_n)\, o_n}{d_{n+1}}.$$

Storage of this matrix in memory only requires the $2N-1$ values $\{d_1, \ldots, d_N, o_1, \ldots, o_{N-1}\}$. Furthermore, a recursion for computing a realization $y = \begin{bmatrix} y_1 & \cdots & y_N \end{bmatrix}^T$ can be derived from multiplying L_2 by an N-vector $x = \begin{bmatrix} x_1 & \cdots & x_N \end{bmatrix}$ of samples from a standard normal distribution, giving

$$\begin{aligned} s_1 &= 0 \\ y_n &= d_n x_n + s_n \\ s_{n+1} &= s_n + o_n x_n \end{aligned}$$

for $n = 1, \ldots, N$. Similarly, for AR(1) correlation, the Cholesky factor is

$$L_3 = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ \phi & \sqrt{1-\phi^2} & 0 & \cdots & 0 \\ \phi^2 & \phi\sqrt{1-\phi^2} & \sqrt{1-\phi^2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi^{N-1} & \left(\phi^{N-2}\sqrt{1-\phi^2}\right) & \left(\phi^{N-3}\sqrt{1-\phi^2}\right) & \cdots & \sqrt{1-\phi^2} \end{bmatrix}.$$

Storage of this matrix only requires the $2N-1$ values

$$1, \phi, \phi^2, \ldots, \phi^{N-1}$$

and

$$\sqrt{1-\phi^2}, \phi\sqrt{1-\phi^2}, \ldots, \phi^{N-2}\sqrt{1-\phi^2}.$$

A recursive algorithm for computing $y = L_3 x$ is

$$\begin{aligned} y_1 &= x_1 \\ y_{n+1} &= \phi y_n + \sqrt{1-\phi^2} x_{n+1}. \end{aligned}$$

Similar recursions can be found for the inverse matrices of L_2, L_3 which suggests an approach for computing the term $\tilde{y}^T R^{-1}(\phi)\tilde{y}$ needed in the likelihood. Since this term is computed repeatedly in the calibration routines, there is potential to save a great deal of computational effort. Caution should be exercised in implementing these strategies, however, to avoid loss of numerical precision due to cancellation and similar effects.

3.2 Matrix Inversion

Analytical expressions for the inverses of the correlation matrices are known, saving the expense of solving those expressions numerically. The uncorrelated case is again trivial, as the inverse of the identity matrix is the identity. For the equicorrelated case, the inverse is given by

$$R_2^{-1}(\phi) = \begin{bmatrix} d & o & o & \cdots & o \\ o & d & o & \cdots & o \\ o & o & d & \cdots & o \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ o & o & o & \cdots & d \end{bmatrix},$$

where the diagonal entries d and off-diagonal entries o are given by

$$d = \frac{1+(N-2)\phi}{1+(N-2)\phi-(N-1)\phi^2}$$
$$o = -\frac{\phi}{1+(N-2)\phi-(N-1)\phi^2}.$$

For the AR(1) case, the inverse is given by

$$R_3^{-1}(\phi) = \frac{1}{1-\phi^2}\begin{bmatrix} 1 & -\phi & 0 & \cdots & 0 & 0 \\ -\phi & \left(1+\phi^2\right) & -\phi & \cdots & 0 & 0 \\ 0 & -\phi & \left(1+\phi^2\right) & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \left(1+\phi^2\right) & -\phi \\ 0 & 0 & 0 & \cdots & -\phi & 1 \end{bmatrix}.$$

3.3 Determinants

The determinant of $R(\phi)$ is required in calibration Case 3 when ϕ is treated as an unknown variable. Each of the correlation cases has an efficiently computable analytical expression for the determinant. The uncorrelated case has determinant 1, although this is irrelevant since in this case calibration of ϕ is meaningless because there is no dependence on ϕ. The equicorrelated case has the determinant

$$|R_2(\phi)| = (1-\phi)^{N-1}\left(1+[N-1]\phi\right)$$

and for the AR(1) case

$$|R_3(\phi)| = \left(1-\phi^2\right)^{N-1}.$$

4 Exact Posterior Densities

Bayes formula expresses the posterior densities of the calibrated (unknown) parameters in terms of the prior densities of the parameters, the assumed likelihood, and the observed data via the equation

$$\pi(\theta|y) = \frac{1}{Z(y)}\pi(y|\theta)\,\pi_0(\theta),$$

where θ is the vector of unknown parameters, y is the observed data, $\pi(\theta|y)$ is the probability density of θ given y, $\pi(y|\theta)$ is the likelihood of y given θ, $\pi_0(\theta)$ is the prior probability density of the parameters, and

$$Z(y) = \int_{D(\theta)} \pi(y|\theta)\,\pi_0(\theta)\,d\theta$$

is an integral taken over $D(\theta)$, the domain of θ. Division by $Z(y)$ ensures the posterior integrates to 1, as required for a probability density. The verification framework outlined in Section 2 has three cases, each with different choices of unknown parameters. Throughout the remainder of the paper, the model residual is denoted $\tilde{y}(\hat{\beta}) = y - G\hat{\beta}$. Expressions for the posteriors and likelihoods of each case are:

$$\begin{aligned}
\text{Case 1: } & \pi\left(\hat{\beta}|y\right) \propto \pi\left(y|\hat{\beta}\right)\pi_0(\hat{\beta}) \\
& \pi\left(y|\hat{\beta}\right) \propto \exp\left[-\frac{\lambda}{2}\tilde{y}(\hat{\beta})^T R_j^{-1}(\phi)\tilde{y}(\hat{\beta})\right] \\
\text{Case 2: } & \pi\left(\hat{\beta},\hat{\lambda}|y\right) \propto \pi\left(y|\hat{\beta},\hat{\lambda}\right)\pi_0(\hat{\beta},\hat{\lambda}) \\
& \pi\left(y|\hat{\beta},\hat{\lambda}\right) \propto \hat{\lambda}^{N/2}\exp\left[-\frac{\hat{\lambda}}{2}\tilde{y}(\hat{\beta})^T R_j^{-1}(\phi)\tilde{y}(\hat{\beta})\right] \\
\text{Case 3: } & \pi\left(\hat{\beta},\hat{\lambda},\hat{\phi}|y\right) \propto \pi\left(y|\hat{\beta},\hat{\lambda},\hat{\phi}\right)\pi_0(\hat{\beta},\hat{\lambda},\hat{\phi}) \\
& \pi\left(y|\hat{\beta},\hat{\lambda},\hat{\phi}\right) \propto \left|R_j(\hat{\phi})\right|^{-1/2}\hat{\lambda}^{N/2}\exp\left[-\frac{\hat{\lambda}}{2}\tilde{y}(\hat{\beta})^T R_j^{-1}(\hat{\phi})\tilde{y}(\hat{\beta})\right].
\end{aligned}$$

As more calibrated parameters are added, the form of the likelihood becomes more complicated due to the need to explicitly include the functional dependence on all unknown parameters. The verification framework described in Section 2 provides an exact expression for the posterior densities of the calibrated parameters. These are presented in the following subsections. We provide derivations of these expressions for the first case and omit the similar (but more tedious) details for the others. For the sake of notational clarity, dependence of R on ϕ and choice of correlation type is suppressed unless needed; i.e., $R_j(\phi)$ is written R.

4.1 Case 1

Non-informative prior: The true parameters λ, ϕ are known. The true parameter β is unknown and approximated by the calibrated parameter $\hat{\beta}$ with the improper prior

$$\pi_0(\hat{\beta}) \propto 1,$$

and the domain of $\hat{\beta}$ equal to $\mathbb{R}^{N_\beta}$. Applying Bayes rule gives the posterior density

$$\begin{aligned}
\pi(\hat{\beta}|y) &= \frac{\pi(y|\hat{\beta})\pi_0(\hat{\beta})}{Z(y)} = \frac{\exp\left[-\frac{\lambda}{2}\tilde{y}^T(\hat{\beta})R^{-1}\tilde{y}(\hat{\beta})\right]}{Z(y)} \\
Z(y) &= \int_{\mathbb{R}^{N_\beta}} \pi(y|\hat{\beta})\pi_0(\hat{\beta})\,d\hat{\beta} = \int_{\mathbb{R}^{N_\beta}} \exp\left[-\frac{\lambda}{2}\tilde{y}^T(\hat{\beta})R^{-1}\tilde{y}(\hat{\beta})\right]d\hat{\beta}.
\end{aligned}$$

The numerator term is a Gaussian function in $\tilde{y}(\hat{\beta})$ while the denominator term $Z(y)$ is a normalizing constant which makes the posterior integrate to 1; i.e., dividing it into the Gaussian function results in a Gaussian probability density. Using $\tilde{y}(\hat{\beta}) = y - G\hat{\beta}$, the numerator can be rewritten as

$$\begin{aligned}
\exp\left[-\frac{\lambda}{2}\tilde{y}^T(\hat{\beta})R^{-1}\tilde{y}^T(\hat{\beta})\right] &= \exp\left[-\frac{\lambda}{2}(y-G\hat{\beta})^T R^{-1}(y-G\hat{\beta})\right] \\
&= \exp\left[-\frac{\lambda}{2}\left(y^T R^{-1}y - 2y^T R^{-1}G\hat{\beta} + \hat{\beta}^T G^T R^{-1}G\hat{\beta}\right)\right] \\
&= \exp\left[-\frac{\lambda}{2}y^T R^{-1}y\right]\exp\left[-\frac{\lambda}{2}\left(\hat{\beta}^T G^T R^{-1}G\hat{\beta} - 2y^T R^{-1}G\hat{\beta}\right)\right],
\end{aligned}$$

where we have used the fact that $y^T R^{-1}G\hat{\beta} = \hat{\beta}^T G^T R^{-1}y$ due to the symmetry of R^{-1}. The factor $\exp\left[-\lambda y^T R^{-1}y/2\right]$ has no dependence on $\hat{\beta}$ and hence can be disregarded for the purposes of determining the density for $\hat{\beta}$ since the integral for

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

$Z(y)$ is taken with respect to $\hat{\beta}$ only. The remaining terms making up the argument of the exponential can be simplified by completing the square to obtain

$$\hat{\beta}^T G^T R^{-1} G\hat{\beta} - 2y^T R^{-1} G\hat{\beta} = (\hat{\beta} - \beta_{mle})^T \left[G^T R^{-1} G\right] (\hat{\beta} - \beta_{mle}) - \beta_{mle}^T G^T R^{-1} y,$$

where

$$\beta_{mle} = \left[G^T R^{-1} G\right]^{-1} G^T R^{-1} y$$

is the maximum likelihood estimate of β. Notice that the term $\beta_{mle}^T G^T R^{-1} y$ has no dependence on $\hat{\beta}$ so it can be factored out of the exponential like the $y^T R^{-1} y$ term was. Setting $\Sigma_1 = \left(G^T R^{-1} G\right)^{-1}$, this implies that

$$\pi\left(\hat{\beta}|y\right) \propto \exp\left[-\frac{1}{2}\left(\hat{\beta} - \beta_{mle}\right)^T \left(\frac{\Sigma_1}{\lambda}\right)^{-1} \left(\hat{\beta} - \beta_{mle}\right)\right],$$

meaning the posterior is a Gaussian density with covariance matrix

$$\frac{1}{\lambda}\Sigma_1 = \frac{1}{\lambda}\left(G^T R^{-1} G\right)^{-1}$$

and mean vector β_{mle}; i.e., $\pi(\hat{\beta}|y) = N(\beta_{mle}, \Sigma_1/\lambda)$.

Gaussian prior: The Gaussian prior case with prior mean β_0 and prior covariance matrix Σ_0/λ is derived similarly to the non-informative case. Recall that $\tilde{y} = y - G\hat{\beta}$ and $\tilde{\beta} = \hat{\beta} - \beta_0$. The only difference is that instead of being able to ignore the $\hat{\beta}$-independent term for the prior density, the terms from the prior combine with the likelihood as in

$$\exp\left(-\frac{\lambda}{2}\tilde{y}^T R^{-1}\tilde{y}\right)\exp\left(-\frac{\lambda}{2}\tilde{\beta}^T \Sigma_0^{-1}\tilde{\beta}\right) = \exp\left(-\frac{\lambda}{2}\left[\tilde{y}^T R^{-1}\tilde{y} + \tilde{\beta}^T \Sigma_0^{-1}\tilde{\beta}\right]\right).$$

After expanding the $\tilde{y}, \tilde{\beta}$ terms and defining $\Sigma_2 = \left(\Sigma_0^{-1} + G^T R^{-1} G\right)^{-1}$, it is possible to complete the square as before and show the posterior is a Gaussian probability density with covariance matrix and mean vector

$$\frac{1}{\lambda}\Sigma_2 = \frac{1}{\lambda}\left(\Sigma_0^{-1} + G^T R^{-1} G\right)^{-1}$$
$$\beta^* = \Sigma_2\left(G^T R^{-1} y + \Sigma_0^{-1}\beta_0\right),$$

respectively. If we denote the posterior mean in the non-informative case β_{mle}, the first term in β^* can be written

$$G^T R^{-1} y = \left(G^T R^{-1} G\right)\beta_{mle},$$

giving the alternative expression for the mean vector

$$\beta^* = \Sigma_2\left[\left(G^T R^{-1} G\right)\beta_{mle} + \Sigma_0^{-1}\beta_0\right].$$

4.2 Case 2

Non-informative prior: The true parameter ϕ is known and the calibrated variables are $\hat{\beta}, \hat{\lambda}$. A non-informative prior is specified for $\hat{\beta}$ and $\hat{\lambda}$, namely the Jeffreys prior, $\pi_0(\hat{\beta}, \hat{\lambda}) \propto 1/\hat{\lambda}$.

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

Following similar steps to the derivation in Case 1, the posterior of the $\hat{\lambda}$ parameter can be shown to be the gamma distribution $\mathrm{Gamma}(a,b)$, having probability density

$$\mathrm{Gamma}(\hat{\lambda}\,|a,b\,) = \frac{b^a}{\Gamma(a)}\lambda^{a-1}\exp\left(-\lambda b\right),$$

where Γ is the Gamma function, the shape parameter a is

$$a = \frac{N - N_\beta}{2}$$

and the rate parameter b is

$$b = \frac{\left(y - G\beta_{mle}\right)^T R^{-1}\left(y - G\beta_{mle}\right)}{2}.$$

Gaussian prior: The posterior distribution is similar in the Gaussian prior case. In fact, the posterior densities are the same but with different parameters. That is, the posterior density for $\hat{\lambda}$ is $\mathrm{Gamma}(\hat{\lambda}\,|a,b\,)$ but with shape parameter

$$a = \frac{N}{2}$$

and rate parameter

$$b = \frac{\left(y - G\beta_{mle}\right)^T R^{-1}\left(y - G\beta_{mle}\right) + \left(\beta_{mle} - \beta_0\right)^T \Sigma_3^{-1}\left(\beta_{mle} - \beta_0\right)}{2},$$

where

$$\begin{aligned}\Sigma_3 &= \Sigma_0 + \Sigma_1\\ &= \Sigma_0 + \left(G^T R^{-1} G\right)^{-1}.\end{aligned}$$

4.3 Case 3

Non-informative prior: In the final case, the correlation parameter $\hat{\phi}$ is calibrated. The posterior density $\pi(\hat{\phi}|y)$ is given by

$$\pi(\hat{\phi}|y) = \frac{1}{c(y)}\left(\frac{A}{b(\hat{\phi})^a|R(\hat{\phi})|^{1/2}|G^T R^{-1}(\hat{\phi})G|^{1/2}}\right),$$

where

$$A = \frac{1}{\hat{\phi}_{max} - \hat{\phi}_{min}}$$

and

$$c(y) = A\int_{\hat{\phi}_{min}}^{\hat{\phi}_{max}} \frac{d\hat{\phi}}{b(\hat{\phi})^a|R(\hat{\phi})|^{1/2}|G^T R^{-1}(\hat{\phi})G|^{1/2}}.$$

Note that $\pi_0(\hat{\phi}) = A = 1/(\hat{\phi}_{max} - \hat{\phi}_{min})$ is the prior of $\hat{\phi}$, where $(\hat{\phi}_{min}, \hat{\phi}_{max})$ is the domain of $\hat{\phi}$. These integrals should be evaluated numerically and the analytical formulas for the determinant of $R(\hat{\phi})$ should be used for computational efficiency.

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

The denominator of $\pi(\hat{\phi}|y)$ goes to zero as a monomial term with exponent N, the number of observations used for calibration, so care must be taken in the evaluation.

Gaussian prior: The posterior density for $\hat{\phi}$ is

$$\pi(\hat{\phi}|y) = \frac{1}{c(y)} \left(\frac{A}{b(\hat{\phi})^a |R(\hat{\phi})|^{1/2} |G^T R^{-1}(\hat{\phi}) G|^{1/2} |\Sigma_3(\hat{\phi})|^{1/2}} \right),$$

where $\Sigma_3(\hat{\phi})$ is as previously described in Case 2,

$$A = \frac{1}{\hat{\phi}_{max} - \hat{\phi}_{min}},$$

and

$$c(y) = A \int_{\hat{\phi}_{min}}^{\hat{\phi}_{max}} \frac{d\hat{\phi}}{b(\hat{\phi})^a |R(\hat{\phi})|^{1/2} |G^T R^{-1}(\hat{\phi}) G|^{1/2} |\Sigma_3(\hat{\phi})|^{1/2}}.$$

5 Energy Statistic Test for Equal Distributions

The previous section detailed exact expressions which can be used to compute the posterior probability densities of the calibrated parameters for each case of the regression framework. We are interested in applying these expressions to the verification of output from Markov chain Monte Carlo (MCMC) algorithms for Bayesian inversion. These methods do not directly compute the posterior density and hence can not be directly compared with the expressions described above. While density estimation methods can be used to approximate a posterior density from the output sample for comparison with the true posterior, it is difficult to use this approach for anything other than qualitative verification due to uncertainties introduced by the density estimation algorithms. Our proposed method for obtaining quantitative results which are better suited for rigorous verification is to apply a statistical test for comparing the distribution of two random samples. This test is based on the energy distance, a class of statistics described in [55]. A test for distribution equality is developed in [44], which we outline here and employ in our framework. The test provides a rigorous way of evaluating whether the differences between a sample drawn from the software being verified and a sample drawn from the true posterior are statistically significant, up to a given level of confidence in the test results. The confidence metrics involved in this approach are readily interpretable statistically, making them well-suited as a means of verifying software whose outputs are inherently random.

5.1 Hypothesis Test for Distribution Equality

Let $X_1, \ldots, X_{n_X}$ and $Y_1, \ldots, Y_{n_Y}$ be independent realizations drawn from two random vectors in $\mathbb{R}^d$. The energy test statistic is defined as

$$\mathcal{E}_{n_X,n_Y} = \frac{n_X n_Y}{n_X + n_Y} \left(\frac{2}{n_X n_Y} \sum_{i,m} \|X_i - Y_m\| - \frac{1}{n_X^2} \sum_{i,j} \|X_i - X_j\| - \frac{1}{n_Y^2} \sum_{l,m} \|Y_l - Y_m\| \right),$$

where the norm $\|\cdot\|$ is the Euclidean norm on $\mathbb{R}^d$, the indices i, j range between 1 and n_X, and the indices l, m range between 1 and n_Y. Note that Σ's in the equation are double summations over each of the indices. The name of the statistic is motivated by its analogy to physical formulas for energy or distance [55].

Let $M_0 = [X_1, \ldots, X_{n_X}, Y_1, \ldots, Y_{n_Y}]$ denote the $d \times (n_X + n_Y)$ matrix of samples, where the first n_X columns designate the first sample and the last n_Y columns the second sample. Roughly speaking, $\mathcal{E}_{n_X,n_Y}$ is large when $\{X_i\}$ and $\{Y_l\}$ are drawn from different distributions and small when they are not. For the equal distribution case, one would expect $\mathcal{E}_{n_X,n_Y}$ to be approximately the same value if computed after the columns of M_0 are randomly permuted and the resulting first n_X columns are taken as the first sample and the last n_Y columns as the second sample.

If $\{X_i\}$ and $\{Y_l\}$ are drawn from different distributions, then the same permutation would result in sets that are more similarly distributed so that the resulting $\mathcal{E}_{n_X,n_Y}$ would be smaller than that obtained from the original two data sets. The test for distribution equality based on the energy statistic is a formalization of this intuitive idea. It is a permutation test with the following steps:

1. Choose a significance level α and initialize the sample counter b to 1.
2. Apply a random permutation to the columns of M_0 and take the first n_X and last n_Y columns of the resulting matrix to be the first and second samples, respectively.
3. Evaluate the energy test statistic for this permutation and label it $\mathcal{E}^b$.
4. Increment b and repeat the previous two steps until a set $\mathcal{E}^1, \ldots, \mathcal{E}^B$ of B energy test statistic values is obtained.
5. Compute $\mathcal{E}$ on the original, unpermuted sets $\{X_i\}$ and $\{Y_i\}$.
6. Compute the empirical p-value (p) as the fraction of $\mathcal{E}^b$ values at least as large as $\mathcal{E}$. If $p < \alpha$, the data $\{X_i\}$ and $\{Y_l\}$ are statistically unlikely to have been drawn from the same distribution.

This test is shown to be statistically consistent in [44]. The paper also reports several empirical results which suggest this test may perform better for small sample sizes than some alternative methods for distribution comparison. These properties make it a good choice as an approach for the verification tasks we are interested in.

5.2 Random Sampling from Posterior Density Approximation

The hypothesis test outlined in Section 5.1 can be used for the verification of MCMC output by comparing a sample drawn from the MCMC algorithm with a sample drawn from the exact posterior. This requires a way of drawing samples from the posteriors. Conditioned on knowledge of ϕ, the parameters $\hat{\beta}$ and $\hat{\lambda}$ are normal and Gamma distributions, respectively. These are well-known parametric distributions which can be sampled from with standard routines. When $\hat{\phi}$ is calibrated, its posterior is not contained in a widely-used class of parametric distributions. Instead, its density is expressed as an integral which is approximated numerically. To use the energy statistic test above for the third calibration case, it is necessary to draw a random sample from this approximation.

Our approach is to use the well-known property that if u is distributed as a uniform variable on $[0,1]$ and ϕ is a random variable with cumulative distribution function $F(\phi)$, then the transformed variable $F^{-1}(u)$ has the same distribution as ϕ. For a small sample, one can simply draw a uniform sample $u_1, \ldots, u_n$ and solve the equation $F(\phi_i) - u_i = 0$ for each ϕ_i using a numerical routine for solving nonlinear equations. The resulting sample $\{\phi_i\}$ will have the desired distribution. If a large sample is desired, it may be more efficient to evaluate a function to directly approximate $F^{-1}(u)$ using interpolation in order to avoid repeated solution of nonlinear equations. We use this approach in our reference implementation. The equation $F(\phi_i) - u_i = 0$ is still solved for ϕ_i, but the u_i are an evenly spaced grid on $[0,1]$ rather than draws from a uniform distribution. The resulting solutions are used as interpolation nodes to directly approximate $F^{-1}(u)$ for arbitrary u. The specific algorithm used in our code involves two steps:

1. Divide the interval $[0,1]$ into n uniformly spaced subintervals with grid-points $F_0 = 0, F_1 = 1/n, \ldots, F_{n-1} = 1 - 1/n, F_n = 1$, and obtain values $\phi_1, \ldots, \phi_{n-1}$ by solving the equation $F(\phi_i) - F_i = 0$ for $i = 1, \ldots, n-1$ on the interior points of the grid.
2. Interpolate a uniformly distributed random sample using F_i as grid points and ϕ_i as function values.

The final result uses linear interpolation over the resulting function-value pairs, where again, the interpolation grid values are F_i and the function values are ϕ_i. Given a sample u from the uniform distribution on $[0,1]$, determine the index i for which $u \in [F_{i-1}, F_i)$. Approximate the associated sample ϕ as follows: $\phi \approx \phi_{i-1} + n(u - F_{i-1})(\phi_i - \phi_{i-1})$. This results in better approximation to the desired random sample and more uniform decrease in its error as n is increased.

5.3 Differences from Verification of Deterministic Output

There is a distinction in the meaning of "verification" using the proposed approach as compared with its use for deterministic problems. Algorithms for deterministic problems will produce the same results for the same inputs no matter how many times they are run. Approaches like the method of manufactured solutions provide a means of verifying a software routine produces correct output, up to some level of error tolerance. A single test result can determine whether the output satisfies that error tolerance constraint.

The situation is somewhat different for the proposed approach. The result of a single energy statistic test depends on the particular random realization of the output. Whether a test fails or not can change from run-to-run due to random variations. When a test does not fail, it does not imply the software has no errors or produces the correct output, up to some tolerance. Rather it implies there is no "statistically significant" difference between the computed output and the expected output, where "statistically significant" is defined by the user through the α parameter. Whereas the tolerance in deterministic verification quantifies an acceptable level of error in the output, tolerance with the proposed method for stochastic verification quantifies an acceptable level of risk that the test result produces a false positive by chance. This distinction should be kept in mind when interpreting results and incorporating such tests into procedures for ensuring software correctness.

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

6 Numerical Studies

We developed a reference MATLAB implementation of the framework described in this paper called LinVer, which is available on GitHub [56]. This code was developed as a reference for those interested in implementing the framework in their own software and was used to produce the results in this section. It makes use of routines in the MATLAB Statistics Toolbox for convenience. The tests in this section require the Delayed Rejection Adaptive Metropolis (DRAM) package for MATLAB [41], but the core functionality of the framework implementation is independent of this package.

The remainder of this section reports the results of three numerical studies designed to test the framework and examine some of its capabilities and limitations. The first study introduces a single error into the likelihood function used by the MCMC algorithm and compares the approximate probability of the energy test reporting an error from the resulting output to that from an error-free implementation. The second study varies the size of an error in the likelihood function used by the MCMC algorithm as well as the sample size used by the energy test and reports the ratio of reported failures in a fixed number of repeated tests. While the first two studies involve errors in the likelihood function used by the MCMC algorithm, the final study introduces two separate errors directly into the MCMC implementation itself and applies the energy test a fixed number of times to samples drawn from the outputs of each of these erroneous implementations.

6.1 Multiple Configurations

Several examples were run using MCMC output generated for a range of parameters to exercise various configurations of the framework. The MCMC output was generated with code employing the DRAM MATLAB package [41]. The code was written so that all calibrated parameters were sampled from a Gaussian proposal distribution in DRAM. Output was generated from two sets of code. The first ("Good") was implemented correctly while the second ("Bad") used an incorrectly implemented Gaussian log-likelihood which left out the factor of $1/2$; i.e., $-\lambda\tilde{y}^T R^{-1}\tilde{y}$ was used rather than the correct $-(1/2)\lambda\tilde{y}^T R^{-1}\tilde{y}$.

6.2 Simulation Parameters

For all cases, the number of observations used for calibration was $N = 100$, the number of regression parameters was $N_\beta = 2$, and the true regression parameters and true error precision parameters were set to

$$\beta = \begin{bmatrix}1.5 & 3.5\end{bmatrix}^T$$
$$\lambda = 10.$$

For the non-informative prior cases, the true correlation structure parameter was $\phi = 0.5$. For the Gaussian prior cases, the true correlation structure parameter was $\phi = 0.2$. When the Gaussian prior for the regression parameters was used, the mean vector and unscaled covariance matrix were

$$\mu_0 = \begin{bmatrix}2 & 3\end{bmatrix}^T$$
$$\Sigma_0 = \begin{bmatrix}1/10 & 0 \\ 0 & 1/10\end{bmatrix}.$$

For all tests, the MCMC output generated a chain of 100,000 iterates using both the delayed rejection and proposal adaptation features of DRAM. The sample to be used in the energy statistic tests was gathered from the the chain by choosing the 20,001st iterate of the chain, then the 20,501st iterate, and subsequent iterates in increments of 500. This resulted in a sample of 160 values. This was partially done to reduce the computational effort required by the unoptimized implementation of the energy statistic test that our reference code uses, as this increases with the size of the sample. This also helped satisfy the assumption that samples used in the energy test are comprised of independent realizations from their respective distributions, as neighboring iterates in an MCMC chain are more correlated. We note that Section 7.4 of [55] mentions the independence assumption is stronger than necessary so this consideration may not be critical for some applications. Starting from the 20,001st iterate allows time for "burn-in" of the chain.

All energy statistic tests were performed with significance level $\alpha = 0.01$. This means that for output from a correct implementation, we would expect no more than 1% of the tests to fail. Each sample drawn from the MCMC code was computed once and tested 500 times against a sample drawn from the exact posterior (using the method described in Section 5.2 whenever $\hat{\phi}$ was calibrated). For each of the 500 tests, the same MCMC sample was used and a new sample from the exact posterior was drawn. A ratio of failed tests was computed from these. Using the binomial distribution, a p-value was computed for these ratios, calculating the probability that 500 random tests with a 1% probability of failing would result in a ratio higher than what was observed. We expect the p-value to be very small when the experiment is run on the incorrectly implemented code and relatively large otherwise.

Calibrated	β Prior	Correlation	Ratio (Good)	p-Value (Good)	Ratio (Bad)	p-Value (Bad)
$\hat{\beta}$	Non-informative	None	0.012	0.2371	0.418	0.0000
$\hat{\beta}$	Non-informative	Equal	0.002	0.9602	0.090	0.0000
$\hat{\beta}, \hat{\lambda}$	Non-informative	Equal	0.000	0.9934	0.430	0.0000
$\hat{\beta}, \hat{\lambda}, \hat{\phi}$	Non-informative	AR(1)	0.002	0.9602	0.362	0.0000
$\hat{\beta}$	Gaussian	Equal	0.000	0.9934	1.000	0.0000
$\hat{\beta}, \hat{\lambda}$	Gaussian	Equal	0.004	0.8766	1.000	0.0000
$\hat{\beta}, \hat{\lambda}, \hat{\phi}$	Gaussian	AR(1)	0.004	0.8766	1.000	0.0000

Table 3. Energy statistic test results for all examined cases. The significance level for all tests was $\alpha = 0.01$. In each case, the test was performed 500 times with each test drawing a new random sample from the exact posterior. The ratio of failed tests to total tests is reported along with the p-value indicating the likelihood that an energy test result would yield a higher ratio by chance. The (Good) label refers to the correctly implemented code and the (Bad) label refers to the code with a scaling error introduced in the likelihood calculation.

Sampling from the exact joint posterior distribution of ($\hat{\beta}$, $\hat{\lambda}$, $\hat{\phi}$) for Case 3 is accomplished as follows:

1. Sample $\hat{\phi}$ from the relevant marginal posterior distribution given in Section 4.3.
2. Sample $\hat{\lambda}$ from the relevant conditional posterior distribution given in Section 4.2, where ϕ is set equal to the $\hat{\phi}$ sampled in the first step.
3. Sample $\hat{\beta}$ from the relevant conditional posterior distribution given in Section 4.1, where ϕ is set equal to the $\hat{\phi}$ sampled in the first step, and λ is set equal to the $\hat{\lambda}$ sampled in the second step.
4. Repeat the first three steps to obtain a set of samples from the desired exact joint posterior distribution.

For Case 2, only the second and third steps are executed with ϕ set to its fixed value. Similarly, for Case 1, the third step is executed with ϕ and λ set to their fixed values.

One test was performed calibrating only $\hat{\beta}$ using a non-informative prior and no correlation in the observation error. Tests calibrating $\hat{\beta}$ alone and $\hat{\beta}, \hat{\lambda}$ for both non-informative and Gaussian priors were performed with equicorrelated observation error. Finally, calibration of $\hat{\beta}, \hat{\lambda}$, and $\hat{\phi}$ was performed with AR(1) observation error for both non-informative and Gaussian priors.

The results of the energy statistic tests for each examined configuration are reported in Table 3. The results of the test are as expected, with the ratio of the number of failures for the correct code generally falling below the $\alpha = 0.01$ significance level and the ratio of number of failures for the incorrect code lying well above it. Out of the correct code tests, only one resulted in a ratio greater than 0.01. Even though the ratio of failures for this case fell above the significance threshold, the p-value for the result is 0.2371, meaning the probability of observing a test with a higher ratio than this is about 23%. This is a plausible figure for correct code, in contrast to the p-values for all of the tests involving the incorrect code, which were 0 up to at least four figures. A second run of this particular test case resulted in a failure ratio of 0.004, which supports the conclusion that this result is simply a chance occurrence.

In addition to the table of energy test results, density estimates were computed from the parameter chains using the MATLAB command `ksdensity`. All points of the chain were used past the initial burn-in iterate (i.e., past the 20,001st chain value). These densities were compared to those computed using the expressions for the exact posteriors. The results of calibrating $\hat{\beta}, \hat{\lambda}$, and $\hat{\phi}$ with the non-informative prior are shown in Figure 1 and Figure 2. The same calibration case with Gaussian prior is shown in Figure 3 and Figure 4. These plots demonstrate the close agreement between the correct MCMC code and the posterior contrasted with the results from the incorrect MCMC code. Plots for the other test cases listed in the table can be found in the supplementary material and show similar agreement.

6.3 Error Sensitivity Study

The proposed verification framework can be viewed as two fairly independent subsystems. The first subsystem is the family of linear regression problems that can be configured to provide a diverse collection of baseline challenges to the

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

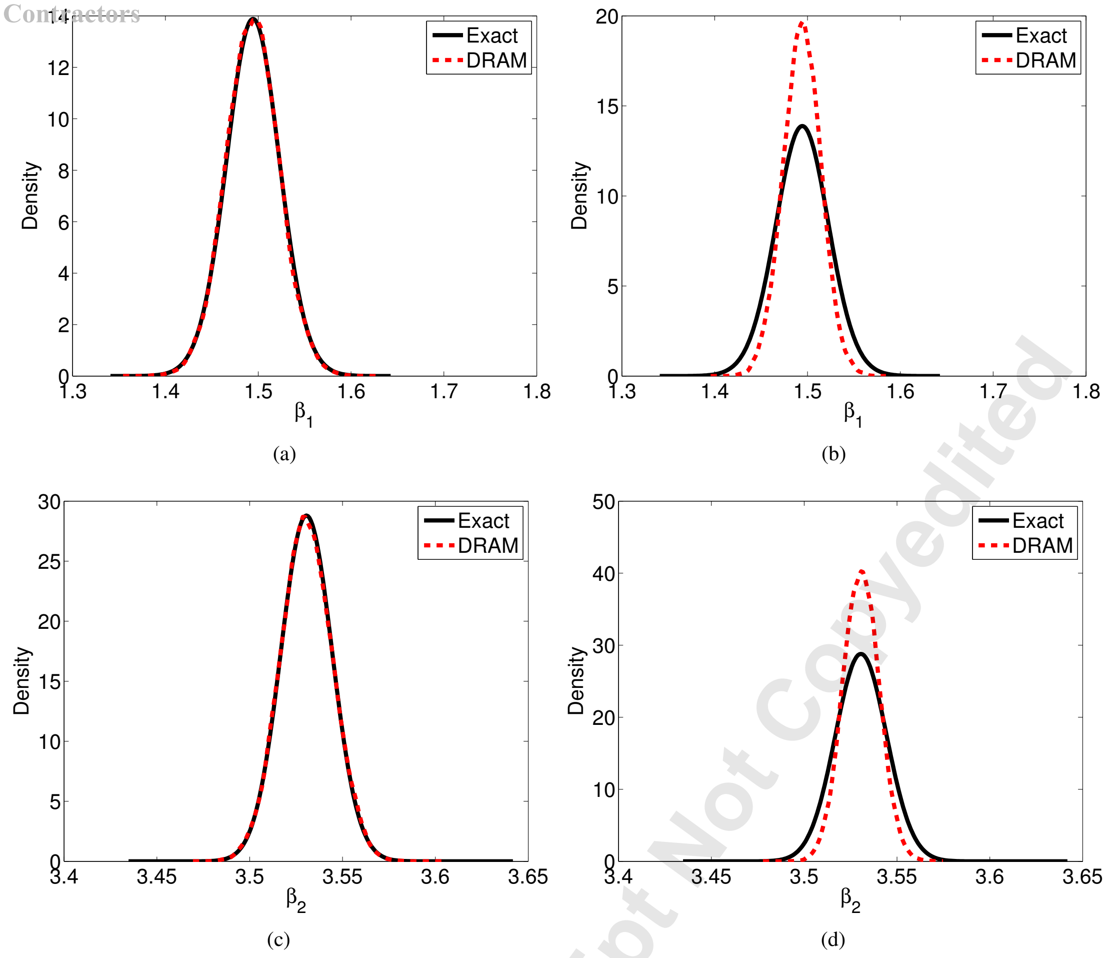

Fig. 1. Comparison of DRAM-computed densities with the analytically computed densities for Case 3 ($\hat{\beta}, \hat{\lambda}, \hat{\phi}$ calibrated) with AR(1) correlated observation error and non-informative prior. (a) $\hat{\beta}_1$ correct code (b) $\hat{\beta}_1$ incorrect code (c) $\hat{\beta}_2$ correct code (d) $\hat{\beta}_2$ incorrect code.

MCMC routine under test that all have analytical or semi-analytical exact solutions to compare the results against. The second subsystem is the energy statistic test that assesses the statistical significance of any difference in distribution between a sample drawn from the MCMC code under test and a sample drawn from the known analytical solution the MCMC code should draw from. The previous numerical experiment tested for proper behavior of the first subsystem by checking that deliberately introduced errors were likely to be distinguished from those of a correct implementation as the configuration was varied over a subset of the available problem space. The experiment in this section examines the second subsystem by fixing the problem configuration and varying the size of the error introduced into the implementation and the number of samples used in the energy statistic test.

Whether or not a given verification methodology reports an issue for a particular error is a function of how sensitive the output of the algorithm under test is to the error as well as how sensitive the error-detecting metric used is to the output changes resulting from the error. This experiment examines the latter for a family of output errors obtained by scaling the log-likelihood supplied to the MCMC algorithm. This is a generalization of the implementation error used in the previous experiment, as leaving off a factor of $\frac{1}{2}$ is equivalent to scaling by 2. This is applied to the Case 1 problem of inferring β with other parameters known, making the scales interpretable as errors in the known observation error variance used in the problem. A non-informative prior is assumed and the observation error is uncorrelated. The remaining calibration parameters used are the same as in the previous experiment.

A collection of samples of size 20, 40, 60, 80, and 100 is drawn from each of the erroneous implementations and tested against a sample of the same size drawn from the true distribution using the energy statistic test. The confidence level of the energy statistic test is set to $\alpha = 0.05$ and 200 tests are performed for each combination of sample size and error effect size. Unlike the previous experiment, each replication of the energy statistic test resamples from the MCMC code as well as the true posterior. Table 4 demonstrates how the reliability of error detection generally increases with sample size and error effect size. The scales used were 1.2, 1.3225, 1.440, 1.5625, 1.69, 1.8225, 2.0 and 3.0. The table reports the KL-divergence of the posterior distribution due to applying these scales and the true posterior.

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

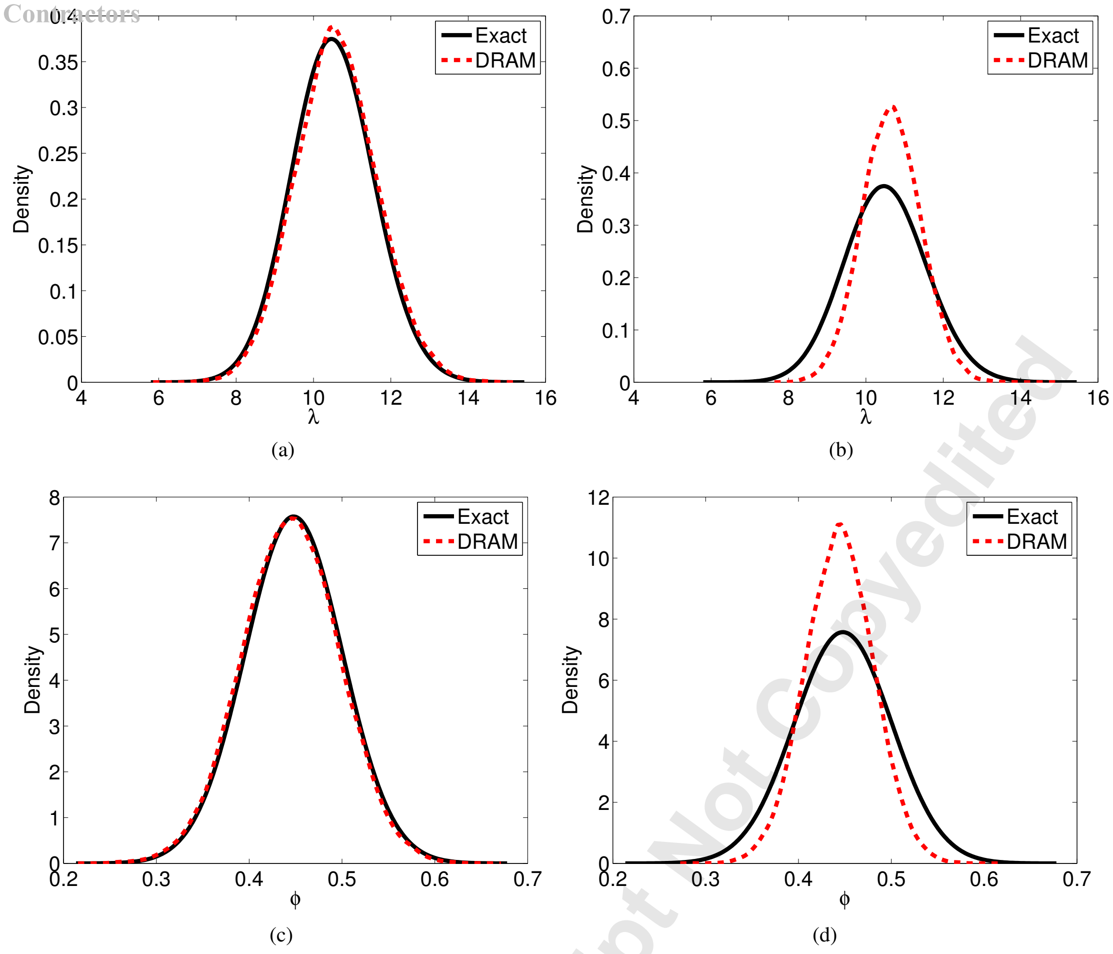

Fig. 2. Comparison of DRAM-computed densities with the analytically computed densities for Case 3 ($\hat{\beta}, \hat{\lambda}, \hat{\phi}$ calibrated) with AR(1) correlated observation error and non-informative prior. (a) $\hat{\lambda}$ correct code (b) $\hat{\lambda}$ incorrect code (c) $\hat{\phi}$ correct code (d) $\hat{\phi}$ incorrect code.

It is noted that this error model is a convenient way of continuously varying the error effect size and we do not claim that it directly corresponds to a realistic set of code errors. This experiment can be considered an examination of the statistical power of our proposed methodology for a set of error effects which are contrived but easily interpretable. This provides some intuition for the relationship between the severity of a hypothetical error and the quantity of sample data needed to reliably detect it.

6.4 Direct Algorithm Errors

The previous two experiments examined the performance of the framework using errors in the log-likelihood function supplied to the MCMC algorithm. Although some MCMC packages implement log-likelihood routines in which these types of errors could conceivably arise, these are actually errors in the MCMC algorithm inputs. In this section, two separate errors are introduced directly into the MCMC algorithm itself and the verification framework is applied. The first error is in the accept / reject stage of the MCMC algorithm, where a uniformly distributed random variable is drawn to determine whether to accept a proposed new iterate in the chain. The error draws a normally distributed random variable in its place. This is a plausible mistake in a MATLAB implementation, for instance, that could occur by accidentally using the `randn` function rather than `rand`. The second error is in the recursive formula used to update the proposal covariance in the adaptive portion of the DRAM algorithm specifically. The formula from [41] is

$$C_{t+1} = \frac{t-1}{t}C_t + \frac{s_d}{t}\left(t\overline{X}_{t-1}\overline{X}_{t-1}^T - (t+1)\overline{X}_t\overline{X}_t^T + X_tX_t^T + \varepsilon_d I\right).$$

Briefly, the subscripts t index the iteration of the MCMC chain, X_t are vectors representing the d-dimensional chain value at iterate t, $\overline{X}_t$ represent the means of X_t at iterate t, C_t is the empirical covariance of the chain at iterate t taken over its history, and s_d is a scaling parameter. The error changes $t-1$ and $t+1$ to t, which is plausible as a transcription error when

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

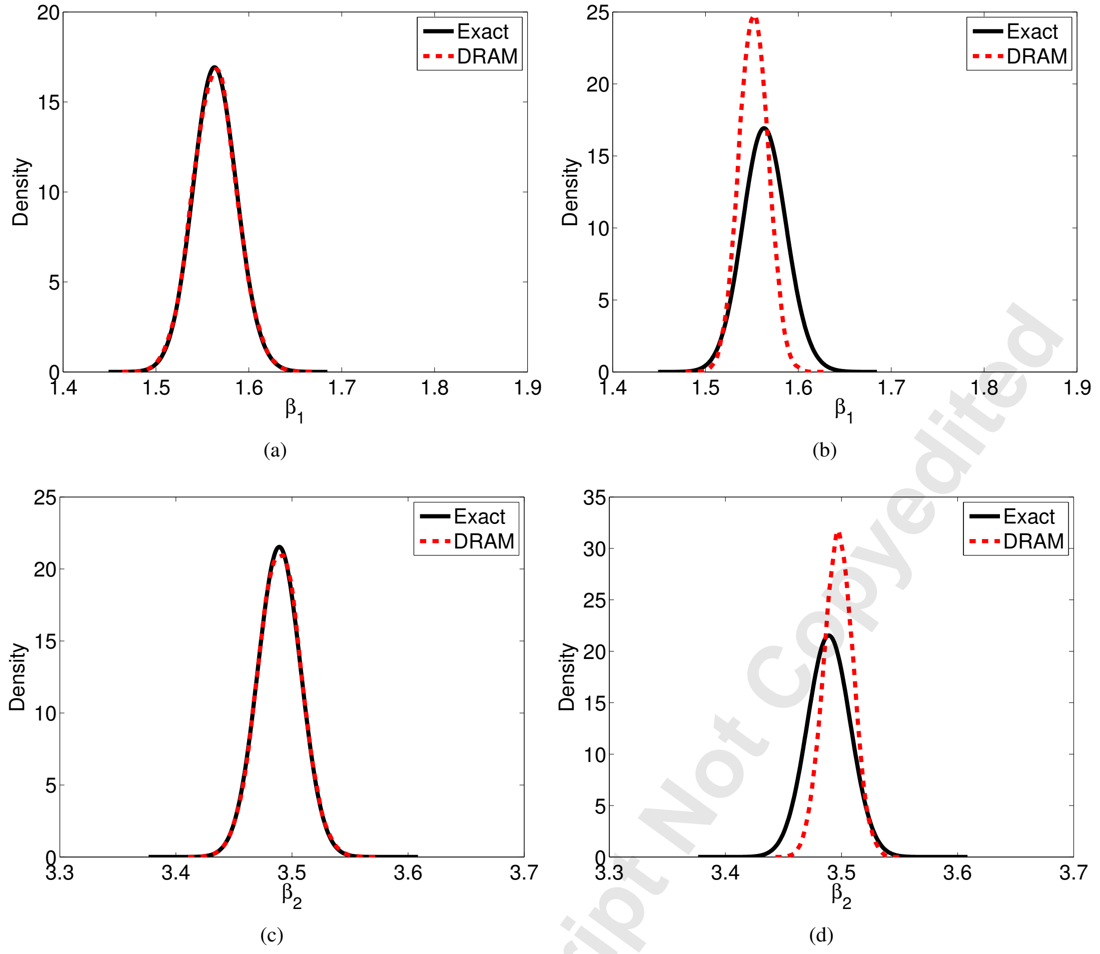

Fig. 3. Comparison of DRAM-computed densities with the analytically computed densities for Case 3 ($\hat{\beta}, \hat{\lambda}, \hat{\phi}$ calibrated) with AR(1) correlated observation error and Gaussian prior. (a) $\hat{\beta}_1$ correct code (b) $\hat{\beta}_1$ incorrect code (c) $\hat{\beta}_2$ correct code (d) $\hat{\beta}_2$ incorrect code.

	KL-Div 0.018	KL-Div 0.043	KL-Div 0.075	KL-Div 0.116	KL-Div 0.165	KL-Div 0.222	KL-Div 0.307	KL-Div 0.901
	Ratio (p-value)	Ratio (p-value)	Ratio (p-value)	Ratio (p-value)	Ratio (p-value)	Ratio (p-value)	Ratio (p-value)	Ratio (p-value)
20 Samples	0.055 (0.300)	0.090 (0.006)	0.080 (0.024)	0.050 (0.417)	0.100 (0.001)	0.095 (0.003)	0.115 (0.000)	0.230 (0.000)
40 Samples	0.065 (0.130)	0.040 (0.673)	0.080 (0.024)	0.085 (0.012)	0.110 (0.000)	0.100 (0.001)	0.225 (0.000)	0.550 (0.000)
60 Samples	0.060 (0.204)	0.070 (0.078)	0.070 (0.078)	0.140 (0.000)	0.145 (0.000)	0.185 (0.000)	0.290 (0.000)	0.810 (0.000)
80 Samples	0.045 (0.545)	0.075 (0.044)	0.125 (0.000)	0.125 (0.000)	0.250 (0.000)	0.250 (0.000)	0.460 (0.000)	0.960 (0.000)
100 Samples	0.060 (0.204)	0.110 (0.000)	0.140 (0.000)	0.235 (0.000)	0.315 (0.000)	0.480 (0.000)	0.555 (0.000)	0.990 (0.000)

Table 4. Energy statistic test results for a range of errors in the supplied observation error likelihood and sample size supplied to the test. The Kullback-Leibler divergence of the resulting distributions is used as a measure of the error effect size and is reported in the top row. The ratio reported is the fraction of failed tests out of 200 with each test drawing a new sample both from the MCMC routine and the exact posterior. The p-value indicating the probability a higher ratio could occur by chance is reported in parentheses. The significance level for all tests was $\alpha = 0.05$.

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

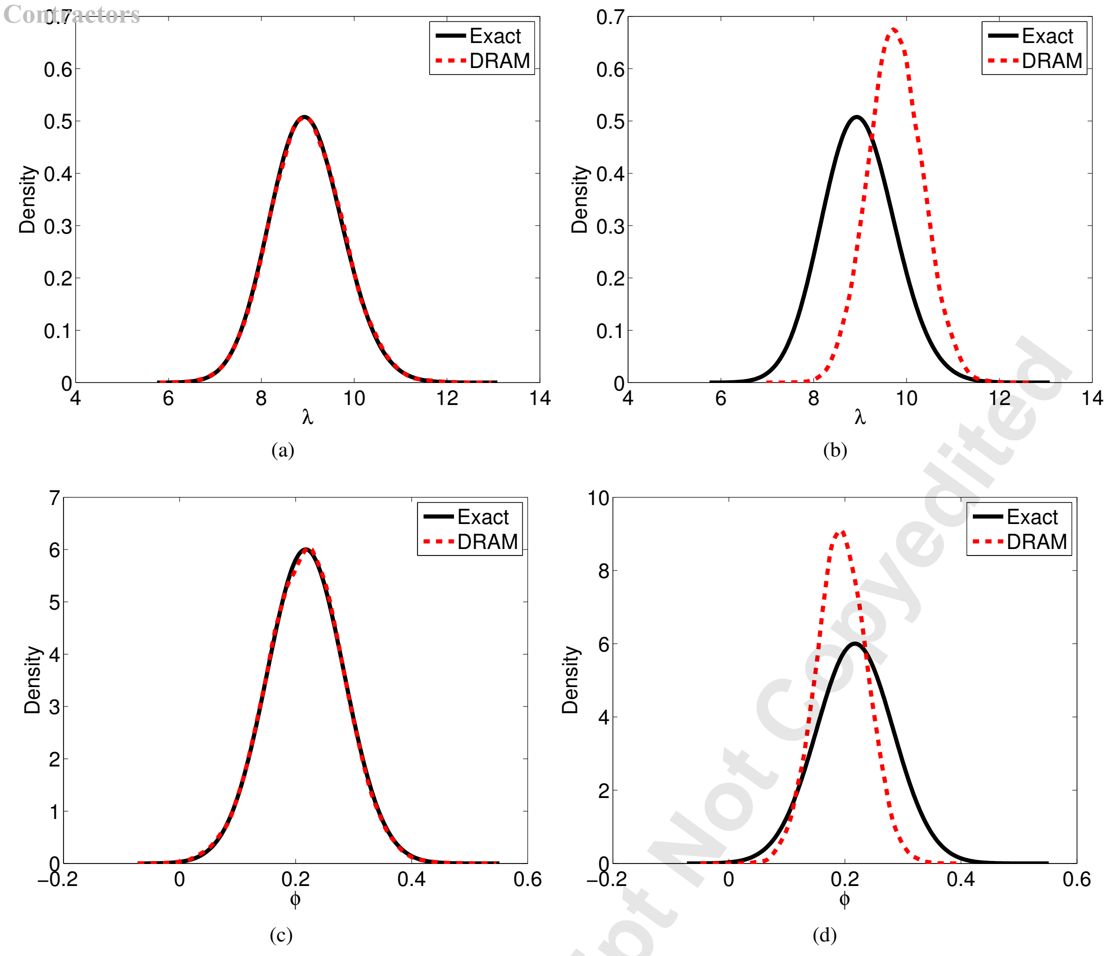

Fig. 4. Comparison of DRAM-computed densities with the analytically computed densities for Case 3 ($\hat{\beta}, \hat{\lambda}, \hat{\phi}$ calibrated) with AR(1) correlated observation error and Gaussian prior. (a) $\hat{\lambda}$ correct code (b) $\hat{\lambda}$ incorrect code (c) $\hat{\phi}$ correct code (d) $\hat{\phi}$ incorrect code.

translating the formula to code.

The results in Table 5 show the energy statistic test fails reliably for the first error but is fairly insensitive to the second. This observation is consistent with the fact that the first error leads to an incorrect MCMC algorithm, while the second error leads to a slightly different adapting proposal covariance matrix at each iteration but with ultimate convergence to the same matrix as obtained from the correct recursion. These errors are similar in the sense that each of them could arise from a subtle textual error in the algorithm source code. While this is certainly a limitation of the approach, it is not unique to it. Furthermore, if the resulting change in output due to an error is too small to be considered statistically significant, if the problem configuration used for verification is close enough to that of a target application, then it may not matter if that error is present or not.

We note that while this experiment indicates some potential of the proposed framework as a verification tool, it is not feasible to assess the types of errors that may be reliably detected with it from this limited study. One approach to gain an understanding of the capabilities of the framework in practice would require the design of a large-scale study of implementation errors, similar to what was done in [35] for the method of manufactured solutions. Unfortunately, as the size of this report shows, such a study is a large-scale effort that is beyond the scope of our current investigation but is an interesting possibility for future research.

7 Conclusion

We have presented a linear-regression-based framework for the verification of Bayesian model calibration routines. The framework provides a flexible model problem with formulas for the exact densities which can be compared with the results of the calibration code applied to the model problem. As highlighted above, the choice of framework has several advantages from the point-of-view of computational efficiency. We demonstrated the use of the framework by comparing an MCMC-based Bayesian parameter estimation code we developed using the DRAM MATLAB package with the results computed from the exact formulas and found good agreement. The energy statistic test was shown to distinguish between a sample

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

	Accept/Reject Bug	Covariance Update Bug
10 Samples	0.985 (0.000)	0.050 (0.417)
30 Samples	1.000 (0.000)	0.080 (0.024)
50 Samples	1.000 (0.000)	0.065 (0.130)
70 Samples	1.000 (0.000)	0.070 (0.078)
90 Samples	1.000 (0.000)	0.070 (0.078)

Table 5. Energy statistic test results for two separate errors introduced in the MCMC algorithm. The first uses an incorrectly distributed random variable when determining whether to accept a new value for the chain drawn from the proposal distribution. The second is an off-by-one error in a recursive formula used to update the proposal covariance to the recent history of the MCMC chain. The sample size supplied to the energy test was also varied. The ratio reported is the fraction of failed tests out of 200, with each test drawing a new sample both from the MCMC routine and the exact posterior. The p-value indicating the probability a higher ratio could occur by chance is reported in parentheses. The significance level for all tests was $\alpha = 0.05$.

drawn from code with a deliberate mistake added to its computation of the posterior and a sample drawn from correctly implemented code. Although our test examples focused on DRAM, the framework is applicable to other MCMC methods such as DREAM or Hybrid Monte Carlo, etc. We also note that while the emphasis of the paper was verification of MCMC routines for Bayesian calibration, the framework itself is also applicable to other approaches.

The framework was useful even while developing the code for these tests. Early results produced a significant ratio of failed tests for the "correct" code whenever variables other than $\hat{\beta}$ were calibrated (although the ratio was still noticeably lower than that of the deliberately incorrect code). This turned out to be due to a bug in the function for calculating the prior, where a factor of $1/\hat{\lambda}$ was left out. Correcting this yielded the results reported here.

Another observation from testing is that the test results can depend on the configuration of the MCMC algorithm. For instance, earlier testing used a smaller MCMC chain, which mostly worked well but resulted in a statistically significant amount of failures for the more difficult inference cases involving $\hat{\phi}$ (although again, the ratio was much smaller than for the deliberately incorrect code). Using a longer chain to give the routine more time to reach convergence resulted in consistently reasonable test ratios. This suggests the framework may be useful not only for detecting errors in software, but for determining MCMC configurations which are suitable for a given inference problem. For example, if an MCMC algorithm were to be used with a nonlinear model, one might solve for the maximum likelihood estimate, linearize the problem, then use the framework to verify the particular MCMC configuration used converges. This would provide at least some level of confidence in obtaining meaningful results.

While the framework is limited to verifying the correct calibration of a set of linear problems, this is an important and general class of models whose verification provides some confidence that the algorithms perform as expected. Its ease of implementation makes it a good choice when limited resources are available for verification development. The models employed are flexible enough that they can be adapted to many application scenarios typical in uncertainty quantification, e.g., approximating nonlinear input-output relationships via the linear model, even though this may be somewhat inefficient and require some application-specific tuning. Furthermore, many components of the framework we have outlined are independent of each other and can be easily substituted with alternatives. For example, sampling from the distribution of a known analytical solution framework could be substituted with sampling from an assumed-correct reference implementation. We reiterate that our results demonstrated code errors that have a reasonable chance of going unobserved by the framework. Practical application of our approach to provide high confidence in an implementation will require additional investigation into the scope of errors that the framework is likely to find. Additionally, we acknowledge that our approach will not scale well for the non-Gaussian cases of our framework or for other non-Gaussian cases in general, due to the need to sample from a true posterior which can be determined analytically or numerically. Hence verification of samplers for high-dimensional, strongly non-Gaussian samplers is not feasible in our current framework and remains a topic of future research.

8 Acknowledgment

This research was supported by the Consortium for Advanced Simulation of Light Water Reactors (http://www.casl.gov), an Energy Innovation Hub (http://www.energy. gov/hubs) for Modeling and Simulation of Nuclear Reactors under U.S.

Department of Energy Contract No. DE-AC05-00OR22725.

References

[1] Tiesler, H., Kirby, R., Xiu, D., and Preusser, T., 2012. "Stochastic collocation for optimal control problems with stochastic PDE constraints". *SIAM Journal on Control and Optimization,* **50**(5), pp. 2659–2682.

[2] Hays, J., Sandu, A., Sandu, C., and Hong, D., 2011. "Motion planning of uncertain fully-actuated dynamical systems: A forward dynamics formulation". *ASME Conference Proceedings,* **2011**(54853), pp. 895–901.

[3] Chauvieère, C., Hesthaven, J. S., and Lurati, L., 2006. "Computational modeling of uncertainty in time-domain electromagnetics". *SIAM Journal on Scientific Computing,* **28**(2), pp. 751–775.

[4] Xiu, D., and Shen, J., 2007. "An efficient spectral method for acoustic scattering from rough surfaces". *Commun. Comput. Phys.,* **2**(1), pp. 54–72.

[5] Bui-Thanh, T., and Ghattas, O., 2014. "An analysis of infinite dimensional Bayesian inverse shape acoustic scattering and its numerical approximation". *SIAM Journal on Uncertainty Quantification,* **2**(1), pp. 203–222.

[6] Hu, Z., Smith, R. C., Burch, N., Hays, M., and Oates, W. S., 2014. "A modeling and uncertainty quantification framework for a flexible structure with macrofiber composite actuators operating in hysteretic regimes". *Journal of Intelligent Material Systems and Structures,* **25**(2), pp. 204–228.

[7] Crews, J. H., McMahan, J. A., Smith, R. C., and Hannen, J. C., 2013. "Quantification of parameter uncertainty for robust control of shape memory alloy bending actuators". *Smart Materials and Structures,* **22**(11), p. 115021.

[8] Najm, H. N., 2009. "Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics". *Annual Review of Fluid Mechanics,* **41**(1), pp. 35–52.

[9] Yu, Y., Zhao, M., Lee, T., Pestieau, N., Bo, W., Glimm, J., and Grove, J., 2006. "Uncertainty quantification for chaotic computational fluid dynamics". *Journal of Computational Physics,* **217**(1), pp. 200 – 216. Uncertainty Quantification in Simulation Science.

[10] Knio, O., and Maître, O. L., 2006. "Uncertainty propagation in CFD using polynomial chaos decomposition". *Fluid Dynamics Research,* **38**(9), pp. 616 – 640. Recent Topics in Computational Fluid Dynamics Recent Topics in Computational Fluid Dynamics.

[11] Xiu, D., and Karniadakis, G. E., 2003. "Modeling uncertainty in flow simulations via generalized polynomial chaos". *Journal of Computational Physics,* **187**(1), pp. 137 – 167.

[12] Cui, T., Marzouk, Y. M., and Willcox, K. E., 2014. "Data-driven model reduction for the bayesian solution of inverse problems". *International Journal for Numerical Methods in Engineering*, pp. n/a–n/a.

[13] Allaire, D., and Willcox, K., 2010. "Surrogate modeling for uncertainty assessment with application to aviation environmental system models". *AIAA journal,* **48**(8), pp. 1791–1803.

[14] Pettit, C. L., 2004. "Uncertainty quantification in aeroelasticity: Recent results and research challenges". *Journal of Aircraft,* **41**(5), pp. 1217–1229.

[15] Green, L. L., Lin, H.-Z., and Khalessi, M. R., 2002. "Probabilistic methods for uncertainty propagation applied to aircraft design". *AIAA Paper,* **3140**, p. 2002.

[16] Hollinger, D. Y., and Richardson, A. D., 2005. "Uncertainty in eddy covariance measurements and its application to physiological models". *Tree Physiology,* **25**(7), pp. 873–885.

[17] Chen, P., Quarteroni, A., and Rozza, G., 2013. "Simulation-based uncertainty quantification of human arterial network hemodynamics". *International Journal for Numerical Methods in Biomedical Engineering,* **29**(6), pp. 698–721.

[18] Estep, D., and Neckels, D., 2006. "Fast and reliable methods for determining the evolution of uncertain parameters in differential equations". *Journal of Computational Physics,* **213**(2), pp. 530 – 556.

[19] Banks, H. T., Hu, S., Jang, T., and Kwon, H.-D., 2012. "Modeling and optimal control of immune response of renal transplant recipients". *Journal of biological dynamics,* **6**(2), pp. 539–567.

[20] Floris, F., Bush, M., Cuypers, M., Roggero, F., and Syversveen, A. R., 2001. "Methods for quantifying the uncertainty of production forecasts: a comparative study". *Petroleum Geoscience,* **7**(S), pp. S87–S96.

[21] Kovscek, A., and Wang, Y., 2005. "Geologic storage of carbon dioxide and enhanced oil recovery. I. uncertainty quantification employing a streamline based proxy for reservoir flow simulation". *Energy Conversion and Management,* **46**(11–12), pp. 1920 – 1940.

[22] Bui-Thanh, T., Ghattas, O., Martin, J., and Stadler, G., 2013. "A computational framework for infinite-dimensional Bayesian inverse problems part i: The linearized case, with application to global seismic inversion". *SIAM Journal on Scientific Computing,* **35**(6), pp. A2494–A2523.

[23] Reed, H., and Hoppe, W., 2016. "A model-based, Bayesian characterization of subsurface corrosion parameters in composite multi-layered structures". *AIP Conference Proceedings,* **1706**.

[24] McMahan, J. A., and Criner, A. K., 2016. "Statistical flaw characterization through Bayesian shape inversion from scattered wave observations". *AIP Conference Proceedings,* **1706**.

Journal of Verification, Validation and Uncertainty Quantification. Received November 20, 2016;
Accepted manuscript posted August 22, 2017. doi:10.1115/1.4037705
Section 3: U.S. Gov Contractors

[25] McMahan, J. A., Aldrin, J. C., Shell, E., and Oneida, E., 2017. "Bayesian flaw characterization from eddy current measurements with grain noise". *AIP Conference Proceedings, To Appear*.
[26] Ghanem, R., Higdon, D., and Owhadi, H., eds., 2017. *Handbook of Uncertainty Quantification*. Springer International Publishing.
[27] Smith, R. C., 2013. *Uncertainty Quantification: Theory, Implementation, and Applications*, Vol. 12. SIAM.
[28] Sullivan, T. J., 2015. *Introduction to Uncertainty Quantification*, 1 ed., Vol. 63 of *Texts in Applied Mathematics*. Springer International Publishing.
[29] Calvetti, D., and Somersalo, E., 2007. *An Introduction to Bayesian Scientific Computing: Ten Lectures on Subjective Computing*, Vol. 2. Springer Science & Business Media.
[30] Tarantola, A., 2005. *Inverse problem theory and methods for model parameter estimation*. SIAM.
[31] Stuart, A. M., 2010. "Inverse problems: a Bayesian perspective". *Acta Numerica,* **19**, pp. 451–559.
[32] Roache, P. J., 1998. *Verification and validation in computational science and engineering*, Vol. 895. Hermosa Albuquerque, NM.
[33] Roy, C. J., 2005. "Review of code and solution verification procedures for computational simulation". *Journal of Computational Physics,* **205**(1), pp. 131–156.
[34] Oberkampf, W. L., and Trucano, T. G., 2002. "Verification and validation in computational fluid dynamics". *Progress in Aerospace Sciences,* **38**(3), pp. 209 – 272.
[35] Roache, P. J., 2002. "Code verification by the method of manufactured solutions". *Transactions-American Society of Mechanical Engineers Journal of Fluids Engineering,* **124**(1), pp. 4–10.
[36] Malaya, N., Estacio-Hiroms, K. C., Stogner, R. H., Schulz, K. W., Bauman, P. T., and Carey, G. F., 2012. "MASA: a library for verification using manufactured and analytical solutions". *Engineering with Computers*, pp. 1–10.
[37] Raju, A., Roy, C., and Hopkins, M., 2005. "On the generation of exact solutions using the method of nearby problems". In 43rd AIAA Aerospace Sciences Meeting and Exhibit, American Institute of Aeronautics and Astronautics.
[38] Adams, B., Hooper, R. W., Lewis, A., Jr., J. A. M., Smith, R. C., Swiler, L. P., and Williams, B. J., 2014. User guidelines and best practices for CASL VUQ analysis using DAKOTA. Tech. Rep. Sandia Technical Report 2014-2864, Sandia National Laboratory, Albuquerque, New Mexico, March.
[39] Diaconis, P., 2009. "The markov chain monte carlo revolution". *Bulletin of the American Mathematical Society,* **46**(2), pp. 179–205.
[40] Chib, S., and Greenberg, E., 1995. "Understanding the Metropolis-Hastings algorithm". *The American Statistician,* **49**(4), pp. 327–335.
[41] Haario, H., Laine, M., Mira, A., and Saksman, E., 2006. "DRAM: Efficient adaptive MCMC". *Statistics and Computing,* **16**(4), pp. 339–354.
[42] Solonen, A., Ollinaho, P., Laine, M., Haario, H., Tamminen, J., Järvinen, H., et al., 2012. "Efficient MCMC for climate model parameter estimation: parallel adaptive chains and early rejection". *Bayesian Analysis,* **7**(3), pp. 715–736.
[43] Neal, R. M., et al., 2011. "MCMC using hamiltonian dynamics". *Handbook of Markov Chain Monte Carlo,* **2**(11).
[44] Székely, G. J., and Rizzo, M. L., 2004. "Testing for equal distributions in high dimension". *InterStat,* **5**, pp. 1–6.
[45] Gregory, K. B., Carroll, R. J., Baladandayuthapani, V., and Lahiri, S. N., 2015. "A two-sample test for equality of means in high dimension". *Journal of the American Statistical Association,* **110**(510), 06, pp. 837–849.
[46] Cai, T., Liu, W., and Xia, Y., 2013. "Two-sample covariance matrix testing and support recovery in high-dimensional and sparse settings". *Journal of the American Statistical Association,* **108**(501), pp. 265–277.
[47] DeGroot, M. H., Schervish, M. J., Fang, X., Lu, L., and Li, D., 1986. *Probability and Statistics*, Vol. 2. Addison-Wesley Reading, MA.
[48] Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., and Tibshirani, R., 2009. *The Elements of Statistical Learning*, Vol. 2. Springer.
[49] Kuhn, M., and Johnson, K., 2013. *Applied Predictive Modeling*. Springer.
[50] Rasmussen, C. E., and Williams, C. K., 2006. *Gaussian processes for machine learning*, Vol. 1. MIT press Cambridge.
[51] Seber, G. A., and Lee, A. J., 2012. *Linear regression analysis*, Vol. 936. John Wiley & Sons.
[52] Hamilton, J. D., 1994. *Time series analysis*, Vol. 2. Princeton university press Princeton.
[53] Engle, R., and Kelly, B., 2012. "Dynamic equicorrelation". *Journal of Business & Economic Statistics,* **30**(2), pp. 212–228.
[54] Alexander, C., and Lazar, E., 2006. "Normal mixture garch (1, 1): Applications to exchange rate modelling". *Journal of Applied Econometrics,* **21**(3), pp. 307–336.
[55] Székely, G. J., and Rizzo, M. L., 2013. "Energy statistics: A class of statistics based on distances". *Journal of Statistical Planning and Inference,* **143**(8), pp. 1249–1272.
[56] McMahan, J. A., 2016. LinVer-Matlab. Available at https://github.com/jmcmahan/LinVer-Matlab.