Quantifying simulator discrepancy in discrete-time dynamical simulators

Richard D. Wilkinson[1, *], Michail Vrettas[1], Dan Cornford[2], and Jeremy E. Oakley[3]

[1]*School of Mathematical Sciences, University of Nottingham,University Park Nottingham NG7 2RD, United Kingdom. Email: r.d.wilkinson@nottingham.ac.uk. Telephone: 00441158467413.*

[*]*Corresponding author*

[2]*School of Engineering and Applied Science, Aston University, Birmingham B4 7ET, United Kingdom*

[3]*School of Mathematics and Statistics, The University Of Sheffield, Sheffield S3 7RH, United Kingdom*

August 3, 2011

Abstract

When making predictions with complex simulators it can be important to quantify the various sources of uncertainty. Errors in the structural specification of the simulator, for example due to missing processes or incorrect mathematical specification, can be a major source of uncertainty, but are often ignored. We introduce methodology for inferring the discrepancy between the simulator and the system in discrete time dynamical simulators. We assume a structural form linear in unknown parameters for the discrepancy function, and show how to infer the maximum likelihood estimates using a particle filter embedded within a Monte Carlo expectation maximization (MCEM) algorithm. We illustrate the effectiveness of this method on two case-studies. The first is a simulation study in which the motion of an object in freefall is simulated with no air resistance. We use noisy observations of the object's location to infer the error in the dynamics

of the simulator. The second case-study looks at a conceptual rainfall runoff simulator (logSPM) used to model the Abercrombie catchment in Australia. We also include a discussion of how to assess the predictive power of dynamic simulators.

1 Introduction

The increasing usage of computer simulators in science and decision making (such as climate science) raises many interesting statistical challenges. Because there is no natural variability in a simulator experiment, quantifying the degree of confidence in predictions is a task that needs to be explicitly undertaken by the modellers. For a given phenomenon and simulator of it, there are several sources of uncertainty: parametric uncertainty from not knowing the 'true' parameters values; initial condition uncertainty; uncertainty in measurements of the system (which is relevant if observations are used to improve forecast performance in a data assimilation scheme, or if forcing functions are imperfectly observed); numerical solver error; and finally, uncertainty from errors in the specification of the structural form of the simulator. Ideally, predictions should account for uncertainty, giving a forecast distribution that incorporates and combines uncertainty from all of these sources.

In this paper we focus on quantifying the simulator structural error in dynamical systems. There are a large variety of reasons why simulators are nearly always imperfect representations of the physical system they were designed to predict. For example, modellers' understanding of the system may be flawed, or perhaps not all physical processes were included in the analysis, and so on. This discrepancy has variously been called model error, model discrepancy, model structural error, and the term we use, *simulator discrepancy*. Once we accept the existence of simulator discrepancy, it is natural to ask whether we can either improve the simulator or quantify the error. The modeller might seek to improve their simulator through more accurate theory. Instead, we ask what can be learnt empirically about the simulator discrepancy, using past predictions and subsequent system observations.

While many methods have been proposed for dealing with parametric and initial condition uncertainty (Saltelli et al., 2000; Oakley and O'Hagan, 2002) and controlling numerical errors (Oberkampf and Trucano, 2008), methodology for quantifying simulator discrepancy is less well developed. The methods that have been proposed broadly classify into subjective methods that rely on expert knowledge (Goldstein and Rougier, 2009; Vernon et al., 2010; Strong et al., 2011), metric based methods to quantify the degree of error in past

performance (Beven, 2006), turning deterministic dynamics into stochastic dynamics (see for example, Crucifix and Rougier (2009)), allowing stochastic parameters to vary through time (Kuczera et al., 2006; Reichert and Mieleitner, 2009), using ensembles of predictions from different simulators (Smith et al., 2009; House et al., 2011), data assimilation based methods (Griffith and Nichols, 1996, 2000), and direct statistical modeling of the simulator discrepancy (Kennedy and O'Hagan, 2001; Higdon et al., 2008; Goldstein and Rougier, 2009).

The method that is developed here is most closely related to the methodology proposed in Kennedy and O'Hagan (2001). They modelled the simulator discrepancy as a state dependent random function using a Gaussian process model. Their approach was for a static experimental situation in which observations were made for different values of the input conditions. The approach does not easily extend to the analysis of dynamical systems. To see why, suppose the output of the simulator is the prediction of a time-series of observations, $y_1, \ldots, y_n$. Under the approach in Kennedy and O'Hagan (2001), the discrepancy would be a function from the initial conditions to a time-series of length n. When n is moderate to large in size, unless a suitably large number of independent trials (time-series) are available then we are unlikely to be able to successfully model the discrepancy. By considering the simulator discrepancy on the level of the dynamics (rather than the static form used in Kennedy and O'Hagan (2001)), we reduce the dimension of the input and output space of the discrepancy function. Their approach is also not suitable in situations where we want to combine the simulator predictions with past observations in a data assimilation scheme in order to improve performance, which is usual in many fields.

In this paper we focus solely on dynamical systems, where we assume there is a state variable $\mathbf{x}$ evolving through time which is noisily observed at discrete points, giving equally spaced observations $\mathbf{y}_1, \ldots, \mathbf{y}_T$. We aim to quantify errors in the prescribed dynamics of the simulator, and try to learn the simulator discrepancy as a function of the current state vector. Quantifying the simulator discrepancy can be thought of as two separate issues: estimating the direction and magnitude of the bias; and quantifying the remaining uncertainty. We aim to do both, modelling the bias using a simple linear regression and quantifying the remaining uncertainty using an additive Gaussian white noise term. Although this is a simple model for the discrepancy, it should be contrasted with the usual approach in data assimilation schemes, which is to use just an additive Gaussian white noise term.

Learning the discrepancy on the dynamics is inferentially difficult, as the true state $\mathbf{x}_t$ is never observed. The simulator dynamics are a map from the

state vector $\mathbf{x}$, to another state at a later time, and it is here where we seek to train the discrepancy, but using only noisy observations $\mathbf{y}_1, \ldots, \mathbf{y}_T$.

The focus of our approach is on improving the predictive power of the simulator. We aim to give probabilistic predictions of future observations that adequately represent the uncertainty in our predictions. Given observations up to time t, $\mathbf{y}_1, \ldots, \mathbf{y}_t$, we aim to provide forecasts $\pi(\mathbf{y}_{t+1}|\mathbf{y}_1, \ldots, \mathbf{y}_t)$ of future events so that the future holds fewer surprises, in the sense that the tails of our distribution are neither too light nor too heavy. This approach is in contrast to focussing on the explanatory power of the simulator, where we would instead focus on achieving a good fit in the simulations to previously observed data (Shmueli, 2011). We do not address the issue of calibrating unknown simulator parameters here, but instead assume that we are provided with a precalibrated simulator in order to quantify its prediction error.

The structure of the paper is as follows. In the next section we describe the framework used to quantify the discrepancy, the methodology to learn the discrepancy, and comment on how to assess probabilistic forecasts made by dynamical systems using scoring rules. In Section 3 we illustrate the methodology on two case-studies. The first is a simulated physics experiment of an object in freefall where we suppose the modellers failed to include air resistance; we show how the missing dynamics can be learnt. The second case-study is a conceptual rainfall-runoff simulator of the Abercrombie water-basin in Australia that has been the focus of several previous uncertainty quantification studies in hydrology. Section 4 offers discussion, and the Appendix contains some of the technical details of the algorithm. The code used in the analysis is available upon request from the authors.

2 Theory

2.1 Statistical forecasting framework

We consider simulators of dynamical systems in which a state vector evolves in time and is noisily observed at regular intervals. Let $\mathbf{x}_t \in \mathbb{R}^d$ denote the value of the state vector at time t, and let $\mathbf{x}_{0:T} = \{\mathbf{x}_0, \ldots, \mathbf{x}_T\}$. We assume we are given an imperfect simulator of the system dynamics, $\mathbf{f}$, that is used to predict one time-step ahead

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t). \tag{1}$$

For example, $\mathbf{f}$ could be a simulator that numerically integrates a system of differential equations $\frac{\mathrm{dx}}{\mathrm{dt}} = \mathbf{h}(\mathbf{x}, \mathbf{u}, t)$ with $\mathbf{x}_t$ as the initial condition. The vector $\mathbf{u}_t$ contains the forcing functions required by the simulator for the

time period in question, and is included in the notation to emphasise that the simulator is a fixed function, not varying through time. Note that we assume the simulator has been calibrated previously, so that there are no unknown simulator parameters that need to be estimated.

We now impose a statistical framework that allows us to relate the simulator to the observations. This consists of two parts; the first relates the observations to the system (the measurement process), and the second relates the simulator prediction to the system (the simulator discrepancy). Let $\mathbf{y}_{0:T} = \{\mathbf{y}_0, \ldots, \mathbf{y}_T\}$ denote a sequence of observations of the state that are conditionally independent given $\mathbf{x}_1, \ldots, \mathbf{x}_T$ and assume that

$$\mathbf{y}_t = g(\mathbf{x}_t) \in \mathbb{R}^p \tag{2}$$

where $g(\cdot)$ is a stochastic mapping. For example, a common choice is to assume that we observe a simple function of $\mathbf{x}_t$ plus zero-mean Gaussian error, such as $\mathbf{y}_t = A\mathbf{x}_t + \mathbf{e}_t$, where $\mathbf{e}_t \sim N_p(\mathbf{0}, \Sigma_{obs})$ and is independent of $\mathbf{e}_s$ for $s \neq t$. Here, $N_p(\boldsymbol{\mu}, \Sigma)$ denotes a p-dimensional multivariate Gaussian distribution with mean vector $\boldsymbol{\mu} \in \mathbb{R}^p$ and $p \times p$ covariance matrix Σ. It is common to only partially observe the state vector so that $\dim(\mathbf{y}_t) = p \leq d = \dim(\mathbf{x}_t)$. We assume that the observation likelihood, $\pi(\mathbf{y}_t|\mathbf{x}_t)$, is known and can be evaluated point-wise.

The second part of the statistical framework is to relate the simulator to reality, by specifying a model of the simulator discrepancy. A common approach in data assimilation is to model the discrepancy as a white noise term, so that errors are independent and identically distributed:

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t) + \boldsymbol{\epsilon}_t \text{ where } \boldsymbol{\epsilon}_t \sim N_d(\mathbf{0}, \Sigma_{sim}).$$

This is equivalent to making the assumption that the prediction error of $\mathbf{f}$ is similar in all parts of space. However, in many scenarios the simulator discrepancy is smaller in some regions of space and larger in others. This occurs in Section 3.1 where we consider a simulator of a falling object with the wrong specification of air-resistance. At low velocities the simulator is accurate, but at higher velocities the simulator error is large. Representing simulator discrepancy as a white noise process ignores this subtlety.

To account for varying simulator accuracy in different parts of space, we introduce a state-dependent simulator discrepancy $\boldsymbol{\delta}(\cdot)$, which is a function of the current state and forcings. We assume that the system dynamics are

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t) + \boldsymbol{\delta}(\mathbf{x}_t, \mathbf{u}_t) \tag{3}$$

(contrast these system dynamics with the simulator dynamics in Equation (1)). The aim of this paper is to describe methodology to infer the functional

form of $\boldsymbol{\delta}$, and to show that the effort of moving from a white simulator discrepancy to a state-dependent discrepancy can significantly improve the performance of the forecasting system. We assume a simple parametric form for $\boldsymbol{\delta}$ linear in the parameters and use ordinary least squares regression to estimate the unknown parameters in $\boldsymbol{\delta}$. Extension to general linear models is straight-forward.

Let $\boldsymbol{\delta}(\mathbf{x}, \mathbf{u}) = (\delta_1(\mathbf{x}, \mathbf{u}), \ldots, \delta_d(\mathbf{x}, \mathbf{u}))^T$. For ease of exposition, we focus on the special case where the stochastic components of the discrepancy are independent for the different dimensions of $\mathbf{x}$, but this is not a necessary assumption. This is equivalent to assuming that $\delta_i(\mathbf{x}, \mathbf{u})$ and $\delta_j(\mathbf{x}, \mathbf{u})$ are conditionally independent given $\mathbf{x}$ (note they are dependent if we don't condition on $\mathbf{x}$). In this special case, we can consider the simulator discrepancy in each of the d dimensions of $\mathbf{x}$ separately. We assume that

$$\delta_j(\mathbf{x}) = \mathbf{p}_j(\mathbf{x}, \mathbf{u})\boldsymbol{\beta}_j + \epsilon_j \tag{4}$$

where $\boldsymbol{\beta}_j$ is a vector of J unknown parameters, $\mathbf{p}_j = (p_j^{(1)}, \ldots, p_j^{(J)})^T$ is a row vector of J specified functions of $\mathbf{x}$ and $\mathbf{u}$, and $\epsilon_j \sim \mathcal{N}(0, \tau_j)$ independently sampled at every occurrence. Examples of regressors we may choose to include are polynomials in each of the dimensions of $\mathbf{x}$ and $\mathbf{u}$, as well as cross terms between these components. We let $\boldsymbol{\theta}$ denote the collection of the $d(J+1)$ unknown parameters in $\boldsymbol{\delta}$. For a deterministic simulator, the probability density function for the system dynamics of $\mathbf{x}$ assumed by our statistical framework is

$$\pi(\mathbf{x}_{t+1}|\mathbf{x}_t, \boldsymbol{\theta}) = \prod_{j=1}^{d} \frac{1}{\sqrt{2\pi\tau_j}} \exp\left[-\frac{1}{2\tau_j}\left(x_{j,t+1} - f_j(\mathbf{x}_t, \mathbf{u}_t) - \mathbf{p}_j(\mathbf{x}_t, \mathbf{u}_t)\boldsymbol{\beta}_j\right)^2\right] \tag{5}$$

where f_j is the jth dimension of the simulator output, and $x_{j,t+1}$ is the jth component of $\mathbf{x}_{t+1}$. We do not explicitly include the forcings $\mathbf{u}$ in the density notation, as we assume they are observed without error.

Estimation of the simulator discrepancy for the dynamics of $\mathbf{x}$, can raise philosophical difficulties. As discussed in Dawid (1986), unobservables are problematic as they are in some sense merely labels. It can be unclear what, if any, physical reality they represent. In conceptual models, $\mathbf{x}$ is often viewed only as a useful tool for modelling and forecasting purposes, but not necessarily as having an operationally defined physical meaning. In this case, it is problematic to discuss the simulator error in $\mathbf{x}$ when $\mathbf{x}$ is not clearly defined. We can avoid this problem by thinking of $\boldsymbol{\delta}$ as a way of decreasing/quantifying errors in forecasts of the observables $\mathbf{y}$, and choosing not to focus on a direct interpretation of $\boldsymbol{\delta}$. In the next section, we introduce a

method for estimating parameter $\boldsymbol{\theta}$. We drop the use of bold notation for vector quantities and stop explicitly including the forcings in the notation.

2.2 Inference for $\delta(\cdot)$

Inferring the shape of the simulator discrepancy is difficult, as it acts on the dynamics of the unobserved state vector. Inference for δ can only be based on observations $y_{0:T}$, and for the parametric model described by Equation (4), the likelihood function $L(\theta) = \pi(y_{1:T}|\theta)$ is unknown. By introducing the hidden state trajectory $x_{0:T}$ into the calculation, the conditional independence structure of the statistical framework can be used. The likelihood of θ given $x_{0:T}$ and $y_{0:T}$ is

$$\begin{aligned}\pi(x_{0:T}, y_{0:T}|\theta) &= \pi(y_{0:T}|\theta, x_{0:T})\pi(x_{0:T}|\theta) \\ &= \left(\prod_{t=0}^{T} \pi(y_t|x_t)\right)\left(\prod_{t=0}^{T-1} \pi(x_{t+1}|x_t, \theta)\right)\pi(x_0). \end{aligned} \tag{6}$$

and could in principle be calculated. The EM algorithm (Dempster et al., 1977) can be used to find the maximum likelihood estimate of θ given $y_{0:T}$, with $x_{0:T}$ acting as the missing data. This is an iterative algorithm which generates a sequence $\theta^{(1)}, \theta^{(2)}, \ldots$ which converge to the MLE $\hat{\theta} = \arg\max_\theta \pi(y_{0:T}|\theta)$, by maximizing

$$Q(\theta, \theta^{(n)}) = \mathbb{E}_X\left[\log \pi(X_{0:T}, y_{0:T}|\theta)|y_{0:T}, \theta^{(n)}\right] \tag{7}$$

with respect to θ (setting the maximizing value equal to $\theta^{(n+1)}$). This algorithm satisfies the likelihood-ascent property, with each subsequent θ increasing the likelihood $L(\theta) = \pi(y_{0:T}|\theta)$. Maximum a posteriori (MAP) estimates can also be found if we specify a prior distribution for θ.

The expectation in Equation (7) is taken with respect to the smoothing distribution $\pi(x_{0:T}|y_{0:T}, \theta^{(n)})$, which is unknown in general and cannot be computed analytically. However, we can sample from $\pi(x_{0:T}|y_{0:T}, \theta^{(n)})$ using sequential Monte Carlo methods. If $\{x_{0:T}^{(i)}\}_{i=1,\ldots,M}$ are samples from $\pi(x_{0:T}|y_{0:T}, \theta^{(n)})$, we can approximate $Q(\theta, \theta^{(n)})$ by

$$\tilde{Q}(\theta, \theta^{(n)}) = \frac{1}{M}\sum_{i=1}^{M} \log \pi(x_{0:T}^{(i)}, y_{0:T}|\theta), \tag{8}$$

and then seek to maximize $\tilde{Q}$, allowing us to bypass the computationally intractable expectation in Equation (7). A consequence of using the Monte Carlo EM algorithm, is that we lose the likelihood-ascent property of the

standard EM algorithm, and so we cannot guarantee convergence (Wei and Tanner, 1990). The number of Monte Carlo samples, M, can be increased for each iteration of the EM algorithm, so that the Monte Carlo error in the estimation of the expectation decreases as we converge on the maximum likelihood estimate $\hat{\theta}$ (Caffo et al., 2005).

Substituting Equation (6) into Equation (8) reduces the problem to maximising

$$\sum_{i=1}^{M}\sum_{t=0}^{T-1}\log\pi(x_{t+1}^{(i)}|x_t^{(i)},\theta) \tag{9}$$

with respect to θ, where we have used the assumption that the prior distribution for x_0 and the observation process do not depend on θ. For various choices of parametric family for δ, Equation (9) can be maximized analytically. In particular, if δ is a linear model with Gaussian noise, such as in Equation (4), then when we substitute Equation (5) for $\pi(x_{t+1}|x_t,\theta)$, and recall that we are assuming conditional independence between the components of δ, the maximization problem in Equation (9) separates into d minimization problems: for $j = 1,\ldots,d$ minimize

$$\frac{1}{2\tau_j}\sum_{i=1}^{M}\sum_{t=0}^{T-1}\left(x_{j,t+1}^{(i)} - f_j(x_t^{(i)},u_t) - p_j(x_t^{(i)},u_t)\beta_j\right)^2 + \frac{1}{2}MT\log\tau_j. \tag{10}$$

These optimization problems can be seen to be equivalent to the classical least squares optimization. Let v_j be the response vector for optimization j, found by stacking elements $x_{j,t+1}^{(i)} - f_j(x_t^{(i)},u_t)$ for $i = 1,\ldots,M$ and $t = 0,\ldots,T-1$, and let Z_j denote the corresponding design matrix found by stacking the rows $p_j(x_t^{(i)},u_t)$ in the same order as for v_j. Maximizing Equation (10) then gives

$$\begin{aligned}\hat{\beta}_j &= (Z_j^T Z_j)^{-1} Z_j^T v_j \\ \hat{\tau}_j &= \frac{1}{MT}(v_j - Z_j\hat{\beta}_j)^T(v_j - Z_j\hat{\beta}_j),\end{aligned}$$

which are the usual maximum-likelihood estimates found in linear regression problems.

We use the bootstrap particle filter (Gordon et al., 1993; Doucet et al., 2001) to generate a single trajectory $x_{0:T}$ from $\pi(x_{0:T}|y_{0:T},\theta)$. This is a sequential importance resampling algorithm, which approximates the filtering distributions $\pi(x_{1:t}|y_{1:t})$ by a weighted sample of N particles $\{(x_{1:t}^{(j)}, w_t^{(j)})\}_{j=1}^N$ with $\sum_{j=1}^N w_t^{(j)} = 1$, so that any expectation can be approximated by a weighted sum:

$$\mathbb{E}(h(x_{0:t})|y_{0:t}) = \int h(x_{0:t})\pi(x_{0:t}|y_{0:t})\mathrm{d}x_{0:t} \approx \sum_{j=1}^{N} w_t^{(j)} h(x_{0:t}^{(j)}).$$

Details of the algorithm are given in the Appendix. In theory the filter generates N smoothed trajectories, although in practice the marginal distribution of x_0 will be degenerate, with typically the same value of x_0 being observed in all N trajectories. To avoid the problem of degeneracy we can either use a particle smoother, such as that suggested by Godsill et al. (2004), or instead run the filter multiple times. We have tried both methods, but found it to be easier to simply run multiple independent particle filters allowing for easier parallelization. To generate M smoothed trajectories, we implement M independent filters, and randomly pick a single smoothed trajectory $x_{0:T}^{(J)}$ from the final filtering distribution $\{(x_{1:T}^{(j)}, w_T^{(j)})\}_{j=1}^N$ with probability $w_T^{(J)}$ in each filter. Because each filter is independent, we avoid the problem of degeneracy for x values towards the start of the time-series.

Because we are using the MCEM algorithm with finite sample size in the smoother and filter, the parameter estimates will continue to fluctuate even after having essentially converged. A stopping rule can be used to decide when to terminate the iterations in the EM algorithm, such as requiring a minimum percentage change in the MLE estimates over consecutive iterations. The stringency of the stopping criterion applied varies in our two examples, due to the differing computational burdens of the two simulators (affecting the size of N and M we can use) and the identifiability of the discrepancy parameters.

A drawback of using the EM algorithm to estimate the MLEs is that error estimation is far from straightforward as the marginal likelihood is not directly available. Standard error estimates are usually found by estimating the Hessian matrix using numerical differentiation, which can then be inverted to estimate the asymptotic variance of the MLE. For example, the supplemented EM algorithm (Meng and Rubin, 1991) uses an identity relating the Hessian matrix to the second derivative of Q and the first derivative of the EM operator (i.e., the derivative of $M(\theta^{(n)}) = \arg\max_\theta Q(\theta, \theta^{(n)})$). These approaches are unlikely to work for the MCEM algorithm. Because we approximate Q by a Monte Carlo sum in the MCEM algorithm, numerical differentiation of Q and of $M(\theta^{(n)})$ is likely to be both prohibitively expensive (computationally) and unstable in most cases. As the focus of our paper is on improving the predictive power of simulators, rather than on the value of the estimated discrepancy, we do not focus on the uncertainty of the parameter estimates here. If uncertainty estimates of the parameters are required, then a Markov chain Monte Carlo (MCMC) approach is likely to be a simpler way to access the uncertainty distributions than the EM algorithm, although this will require considerably more computation.

2.3 Assessing forecasting systems

Our motivation for quantifying simulator error is to improve forecasting power, both in terms of reducing absolute error and quantifying uncertainty. As the majority of statistical diagnostic tools are designed to assess explanatory power rather than predictive power (Shmueli (2011)), we now make clear how we will judge the success or otherwise of a forecast.

We base assessment on the ability to predict future observations given past observations, via the use of the k-step-ahead forecast distributions $\pi(y_{t+k}|y_{1:t})$. Typically, we focus on $k = 1$, the one-step-ahead forecasts, but there may be benefits from looking further ahead in some situations. We use a training sequence of data $y^{(1)}_{1:T_1}$ to train the model, and then use an independent validation data set $y^{(2)}_{1:T_2}$ in the testing. To find $\pi(y_{t+k}|y_{1:t})$ we use a data assimilation scheme to obtain the filtering distributions $\pi(x_t|y_{1:t})$, before propagating these through Equation (3) to find $\pi(x_{t+k}|y_{1:t})$ and then through the observation process (Equation (2)) to find $\pi(y_{t+k}|y_{1:t})$. In all but the simplest of simulators, it is not possible to analytically calculate these distributions and so all calculations are done using a weighted ensembles of particles obtained from the particle filter (see Section 2.2).

We wish to assess both the bias and the uncertainty quantification of the forecasts. To assess the bias, we only need the means of the forecasts. Let $m_t(k) = \mathbb{E}(y_{t+k}|y_{1:t})$ be the mean k-step-ahead forecast at time t. We use the mean-square-error (MSE) and the Nash-Sutcliffe (NS) statistic (Nash and Sutcliffe (1970)) applied to the mean forecast

$$\text{MSE} = \frac{1}{T-k}\sum_{t=1}^{T-k}(y_{t+k} - m_t(k))^2, \qquad \text{NS} = 1 - \frac{\sum_{t=1}^{T-k}(y_{t+k} - m_t(k))^2}{\sum_{t=k+1}^{T}(y_t - \bar{y})^2}$$

to assess the accuracy of the mean forecast. The Nash-Sutcliffe statistic is an analogue of the coefficient of determination, R^2, and is commonly used in hydrology to assess simulator accuracy. It compares the mean forecast performance with the performance of the climatological forecast $\bar{y} = \frac{1}{T}\sum y_t$. The values are often converted to percentages, so that 100% indicates perfection. Any score greater than 0% indicates superior performance to the climatological forecast.

Although the mean-square-error and Nash-Sutcliffe statistics are useful for quantifying the bias of forecast systems, they ignore any quantification of uncertainty. Scoring rules can be used to assess probabilistic forecasts, as they judge forecasts not only on their mean prediction, but also on the accuracy of the uncertainty quantification (see Jolliffe and Stephenson (2003) for an introduction). A score is said to be proper if it is optimized for

well-calibrated probability assessments (Gneiting and Raftery, 2007), and propriety is considered an essential attribute in scientific forecast evaluation. We focus on two specific scores, the continuously ranked probability score (CRPS) and the Dawid score (DS), both of which are proper. If $F(y)$ is the distribution function of the forecast, and if $\tilde{y}$ is the observation, then the CRPS is defined to be

$$\text{crps}(F, \tilde{y}) = \int_{-\infty}^{\infty} (F(x) - \mathbb{I}_{y \geq \tilde{y}})^2 \text{dy}. \tag{11}$$

If $\pi(\cdot)$ is the density function of the forecast, then it can be shown that

$$\text{crps}(\pi, \tilde{y}) = \mathbb{E}_\pi ||Y - \tilde{y}|| - \frac{1}{2}\mathbb{E}_\pi ||Y - Y'|| \tag{12}$$

where Y and Y' are independent copies of a random variable with probability density function $\pi(\cdot)$. This representation allows the CRPS to be estimated by a Monte Carlo estimate using an ensemble of forecasts. If $\{y^{(j)}, w^{(j)}\}_{j=1}^M$ is a weighted approximation to $\pi(y_{t+k}|y_{1:t})$ then we use the approximation

$$\text{crps}(\pi, \tilde{y}) = \sum_{j=1}^{M} w^{(j)} ||y^{(j)} - \tilde{y}|| - \frac{1}{2} \sum_{i,j=1}^{M} w^{(i)} w^{(j)} ||y^{(i)} - y^{(j)}||$$

to estimate the CRPS. Note that if the forecast is deterministic, then Equation (12) reduces to the absolute error

$$\text{crps}(Y, y) = |Y - \tilde{y}|.$$

Hence, the CRPS generalises the absolute error, allowing us to compare probabilistic and deterministic forecasts.

Dawid and Sebastiani (1999) proposed a score that depends only on the first and second moments of the prediction. Suppose we have a forecasting system that makes predictions $\pi(\cdot)$, with expected value m and variance s^2, and that we observe $\tilde{y}$. Then

$$S(\pi, \tilde{y}) = \left(\frac{\tilde{y} - m}{s}\right)^2 + \log s^2,$$

which we will call the Dawid score, is a proper scoring rule which is closely related to several commonly used measures (Gneiting and Raftery, 2007). Note that if the forecast is Gaussian, then the Dawid score is just a linear transformation of the logarithmic score.

We compare forecasting systems by calculating the average score across a sequence of observations

$$\text{CRPS} = \frac{1}{T-k}\sum_{t=1}^{T-k} \text{crps}(F_{t,k}, y_{t+k}), \quad \text{DS} = \frac{1}{T-k}\sum_{t=1}^{T-k}\left[\left(\frac{y_{t+k}-m_t(k)}{s_t(k)}\right)^2 + \log s_t(k)^2\right],$$

where $F_{t,k}$ is the distribution function of the k-step-ahead forecast. Both scores are written in their negative orientation, so that the forecast system with the smallest value is preferred. We can convert the raw CRPS value into a skill score by comparing it to the score attained by a reference forecast (such as climatology) in the same way the Nash-Sutcliffe statistic converts raw mean-square-error values into a percentage by comparing the forecast with climatology $\bar{y}$. We define the continuously ranked probability skill score (CRPSS) to be

$$\text{CRPSS} = 1 - \frac{\text{CRPS}_{\text{forecast}}}{\text{CRPS}_{\text{reference}}},$$

which can also be converted into a percentage. It is not as simple to convert the Dawid score into a skill score because a perfect forecast has a Dawid score of minus infinity. Instead we report the raw value of the Dawid score in each case. Finally, plots of the forecast errors versus the fitted values can also be used to assess the forecasting system.

3 Case studies

We demonstrate the methodology in two case-studies. The first is an artificial simulation study where we simulate noisy data from a model assumed to be the true system, and then use an incorrect version of that model and try to learn the simulator discrepancy. The second example examines the simulator discrepancy in a rainfall-runoff simulator that has been the focus of several previous studies.

3.1 Case study 1: Free fall

We begin with an idealised example where we assume we know the true behaviour of the system and use an incorrect model as a simulator, and then infer the simulator discrepancy using noisy observations. For the true model, consider an object in free fall near the surface of the earth. This is a two-dimensional system described by a displacement velocity state-vector (x, v). Assuming the object has constant acceleration g and is subject to Stokes'

drag with coefficient k, then the differential equations determining the true system behaviour are

$$\frac{\mathrm{dv}}{\mathrm{dt}} = g - kv, \qquad \frac{\mathrm{dx}}{\mathrm{dt}} = v. \tag{13}$$

We assume x is observed at regular points in time ($t = 0, \Delta t, 2\Delta t, \ldots$) with zero mean Gaussian measurement error. We let x_n and v_n denote the position and velocity when the n^{th} observation, y_n, is made

$$y_n \sim \mathcal{N}(x_n, \sigma^2_{obs}).$$

Equations (13) imply that the one-step-ahead dynamic updates for the system are

$$\begin{aligned} x_{n+1} &= x_n + \frac{1}{k}(\frac{g}{k} - v_n)(\mathrm{e}^{-\mathrm{k}\Delta\mathrm{t}} - 1) + \frac{\mathrm{g}\Delta\mathrm{t}}{\mathrm{k}}, \\ v_{n+1} &= (v_n - \frac{g}{k})\mathrm{e}^{-\mathrm{k}\Delta\mathrm{t}} + \frac{\mathrm{g}}{\mathrm{k}}. \end{aligned} \tag{14}$$

For the (incorrect) simulator, we suppose that the modellers neglected to include air resistance ($k = 0$ in Equation (13)), so that the simulator has one-step-ahead dynamics

$$\begin{aligned} x_{n+1} &= x_n + v_n\Delta t + \frac{1}{2}g(\Delta t)^2 \\ v_{n+1} &= v_n + g\Delta t. \end{aligned} \tag{15}$$

The discrepancy function $\boldsymbol{\delta}(x, v)$ for the difference between the system dynamics and the simulator dynamics can be calculated explicitly in this case, giving

$$\boldsymbol{\delta}(x,v) = \frac{g}{k}\begin{pmatrix} \frac{1}{k}(\mathrm{e}^{-\mathrm{k}\Delta\mathrm{t}} - 1) + \Delta\mathrm{t} - \frac{1}{2}\mathrm{k}(\Delta\mathrm{t})^2 \\ 1 - k\Delta t - \mathrm{e}^{-\mathrm{k}\Delta\mathrm{t}} \end{pmatrix} - v\begin{pmatrix} \frac{1}{k}(\mathrm{e}^{-\mathrm{k}\Delta\mathrm{t}} - 1) + \Delta\mathrm{t} \\ 1 - \mathrm{e}^{-\mathrm{k}\Delta\mathrm{t}} \end{pmatrix}, \tag{16}$$

which is a linear function in v, with no dependence on x. The discrepancy illustrates the difficulty faced. Equation (16) depends solely on the velocity, not the displacement, yet we only observe the displacement. Hence, learning $\boldsymbol{\delta}$ relies upon our ability to infer the velocity trajectory, which can then be used to train the discrepancy model.

The discrepancy $\boldsymbol{\delta}$ is a map from $\mathbb{R}^2$ to $\mathbb{R}^2$, which we denote as

$$\boldsymbol{\delta}(x,v) = \begin{pmatrix} \delta_x(x,v) \\ \delta_v(x,v) \end{pmatrix}.$$

Note that although the true discrepancy is deterministic, we need to use a stochastic model for $\boldsymbol{\delta}$, because measurement error and the finite number of observations make the estimate of $\boldsymbol{\delta}$ uncertain. A statistical model is used in order to describe this uncertainty. In more complex situations, we may not expect a deterministic discrepancy function to exist (for example the state vector may not contain enough information to fully model the system dynamics, or if the system is stochastic), and so a statistical model will be vital.

Expert opinion can be useful when deciding which family of parametric models we should use for δ. In this case, universality allows us to argue that the discrepancy will not depend on x so long as we are near the surface of the earth. So in this case, we could have proposed a model for $\boldsymbol{\delta}$ that depended only on v. Furthermore, someone experienced with both the system and the simulator may be able to make rudimentary comments about the shape of $\boldsymbol{\delta}$, although expert elicitation of simulator error can be challenging (Goldstein and Rougier, 2009; Vernon et al., 2010).

We begin by fitting a model of the correct parametric form, namely

$$\boldsymbol{\delta}(x, v) = \begin{pmatrix} a_x + b_x v \\ a_v + b_v v \end{pmatrix} + \begin{pmatrix} e_x \\ e_v \end{pmatrix} \tag{17}$$

where $e_x \sim \mathcal{N}(0, \tau_x)$ and $e_v \sim \mathcal{N}(0, \tau_v)$, and estimate the six unknown parameters $(a_x, b_x, \tau_x, a_v, b_v, \tau_v)$. We used a time series of 100 noisy observations of the true system generated with air-resistance fixed at $k = 0.1$ and a time step between observations of $\Delta t = 0.5$. We allowed measurement error to vary to show the effect on our ability to accurately estimate $\boldsymbol{\delta}$. We used 1000 filtering particles and five smoothed trajectories in the inference scheme. The stopping rule used to decide when to terminate the iterations in the EM algorithm, was to look at the last five estimates for a particular parameter and to test whether all five estimates are within 0.001 of each other. We stopped the EM algorithm when this condition was met simultaneously for all six parameters.

The results are shown in Table 1. The first thing to note is that the estimated coefficient of v in the δ_v discrepancy, b_v, always matches the true value to two significant figures. The other parameter estimates in δ_x and δ_v are of the right magnitude and sign, but are further from the true values. We believe parameter b_v is well estimated because it is the key determining factor for the simulator discrepancy and if we estimate v incorrectly at stage n, the subsequent estimate of x will be wrong at time $n+1$. The parameters estimates for b_x and a_v appear to be biased, but this appears to be a quirk of the time-period and number of observations used. The two variance param-

σ_{obs}	#iters.	δ_x estimate	δ_v estimate	τ_x	τ_v
0.01	17	$-0.019 - 0.0062v$	$-0.18 - 0.049v$	3.5×10^{-5}	2.0×10^{-5}
0.1	30	$-0.014 - 0.0062v$	$-0.18 - 0.049v$	2.7×10^{-4}	9.9×10^{-5}
0.5	173	$-0.019 - 0.0062v$	$-0.18 - 0.049v$	1.6×10^{-4}	9.5×10^{-6}
1	215	$-0.066 - 0.0056v$	$-0.17 - 0.049v$	2.9×10^{-4}	7.3×10^{-6}
5	267	$-0.037 - 0.0058v$	$-0.15 - 0.049v$	1.3×10^{-4}	1.3×10^{-4}
10	519	$-0.050 - 0.0059v$	$-0.19 - 0.049v$	8.7×10^{-6}	2.2×10^{-4}

Table 1: Parameter estimates found using Equation (17) as the discrepancy function. σ_{obs} is the standard deviation of the Gaussian measurement error used, #iters. is the number of iterations required by the EM algorithm in order to reach convergence, and τ_x and τ_v are the estimates of the variances of the Gaussian white noise part of the discrepancies for x and v. All values are reported to two significant figures. The true values of the two discrepancy functions are $\delta_x(x, v) = -0.020 - 0.012v$ and $\delta_v(x, v) = -0.12 - 0.049v$, with $\tau_x = \tau_v = 0$ in both cases.

eters, τ_x and τ_v, are unable to reach zero, because the other parameters are not accurately estimated and so the variance is inflated to account for this.

More precise estimates of the parameters could be found by replacing the crude stopping rule used to terminate the EM algorithm, and using a larger number of filtering and smoothing particles. However the aim of this paper is to show the improvement that can be made to the forecasting system, and the estimates above are more than sufficient to do this. If we take these discrepancy estimates and use them in a forecasting system, the improvements in predictive power immediately become clear. Table 2 shows the performance of various different forecasting systems. The test data used was a sequence of 100 new observations (with measurement error $\sigma^2_{obs} = 0.1$) generated from different starting values of x and v to those used in the training data (we used an object fired upwards which then decelerates before falling back to the ground). The first row of the table contains the forecast error from simply running the deterministic simulator from the initial conditions and adding measurement error. The second row was found by using the true value of the state vector with the deterministic simulator to predict the next observation (note this would not be possible in a real situation). The third row is the result of using the simulator plus a white noise simulator discrepancy, with variances estimated from the data to be $\tau_x = 3084$ and $\tau_v = 27.4$. The fourth row is the simulator plus the discrepancy estimated in Table 1 when $\sigma_{obs} = 0.1$ (row 2). The results show the vast improvement in predictive power that can be achieved in this case by training a discrepancy term.

Forecasting framework	MSE	NS (%)	DS	CRPSS (%)
Simulator only	9.57×10^6	-585.2	9.567×10^8	-124.86
Simulator only, corrected	1708	99.9	170800	1.94
Simulator plus white noise	11900	99.1	11.9	95.03
Sim. + discrep. (line 2 Table 1)	0.0296	100.0	-1.877	99.99

Table 2: Predictive performance of various different forecasting systems, as measured by the mean-square-error (MSE), the Nash-Sutcliffe statistic (NS), the Dawid score (DS), and the continuously ranked probability skill score (CRPSS). Low values of MSE and DS indicate good predictive performance, as do NS and CRPSS values close to 100%. The reference forecast used for the NS statistic and the CRPSS was a Gaussian distribution with mean and variance estimated from the observations. Values of NS and CRPSS greater than 0 indicate the predictions are superior to the reference forecast.

These results also highlight the danger of relying on a single diagnostic measure to judge predictive performance. The Nash-Sutcliffe statistic indicates very similar (and excellent) performance for the simulator plus white noise, and the simulator plus discrepancy frameworks because it only judges the mean prediction of the simulator. The DS and the CRPSS also take into account the uncertainty quantification of the forecasts, and show that superior predictions are made by the full simulator plus discrepancy framework.

Figure 1 shows the forecast errors $y_{t+1} - \mathbb{E}(y_{t+1}^{\text{rep}}|y_{1:t})$ plotted against the fitted values $\mathbb{E}(y_{t+1}^{\text{rep}}|y_{1:t})$, where y_{t+1}^{rep} denotes theoretical repetitions of the t+1th observation assuming the model is true. These plots are the analogue of the residual plots used as diagnostic tools in linear regression. If the framework is correct, we should see the residuals form an uncorrelated band distributed symmetrically about $y = 0$. Only the residual plot for the full discrepancy model looks like this. The tail near 2000 appears because these observations are made at low velocities where the discrepancy correction is less effective. The other plots all show a high degree of correlation between the errors, indicating problems with the forecasting system.

This case-study has shown the ability of a discrepancy function to improve forecast accuracy and we have demonstrated that the methodology works in this case. In the next section we examine a real situation where we do not expect to be able to find such a neat solution. Although this was a simple example, it has demonstrated the difficulty involved in making such inferences. Here we successfully inferred the structure of a discrepancy which depends on a variable (v) that is never observed. Finally, note that the dynamics and observation process are linear and Gaussian, and so in

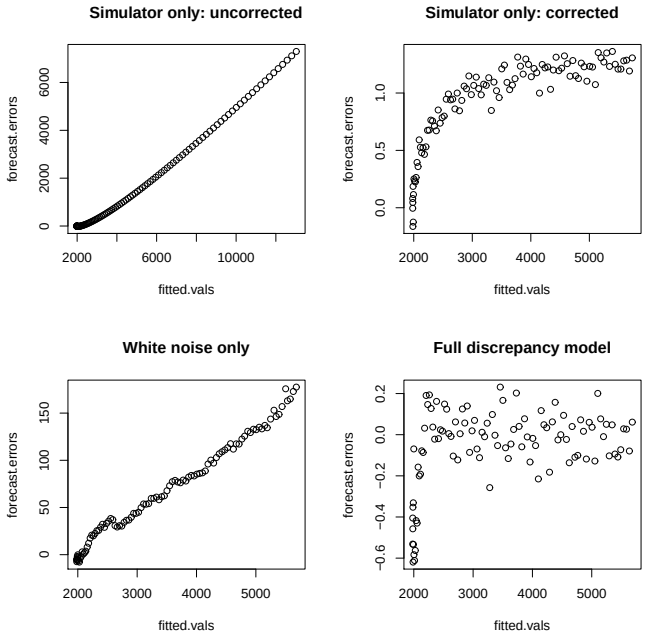

Figure 1: A residual plot showing the one-step-ahead forecast errors versus the fitted values. The four plots correspond to the first four cases described in Table 2.

this case the Kalman smoother could be used rather than the particle filter. This would be much more efficient and would significantly decrease the computation time.

3.2 Case study 2: Rainfall-runoff simulator of the Abercrombie watershed

We now turn our attention to a simulator from hydrology that has been the subject of several previous analyses in the literature on uncertainty quantification in computer experiments (Kuczera et al., 2006; Reichert and Mieleitner, 2009; Conti et al., 2009). The logSPM simulator is a conceptual rainfall-runoff model from the saturated path modelling (SPM) family (Kavetski et al., 2003) used to model the conversion of rainfall into runoff. The model can be considered as three linked conceptual stores (representing soil, ground, and river water stores) with flow between, in, and out of the compartments at different rates. Each store can be thought of as a box, with a base area equal to the area of the catchment, containing a varying depth of water (see Figure 2). Water enters the catchment area as rain and leaves either through river discharge, evaporation, or percolation to deep aquifers. We model the system by a three dimensional temporally varying state vector, denoted $\mathbf{h}(t) = (h_{\text{soil}}(t), h_{\text{gw}}(t), h_{\text{river}}(t))$, which represents the spatially averaged depth of water in each store (measured in mm) at time t. The mathematical specification of the simulator is given by mass balance equations for each of the three conceptual stores.

1. The depth of water in the soil store is denoted $h_{\text{soil}}(t)$ (mm), and increases at rate $(1 - f_{\text{sat}}(t))R(t)$, due to mass flux from rain, $R(t)$ (mm/day), minus surface runoff, $R(t)\ f_{\text{sat}}(t)$. The proportion of rain diverted to overland flow depends on the saturation of the soil which is modelled by a modified logisitic function

$$f_{\text{sat}}(t) = \frac{1}{1 + \phi_{\text{F}} \exp(-\phi_{\text{s}} h_{\text{soil}}(t))} - \frac{1}{\phi_{\text{F}} + 1}.$$

   Water in the soil store decreases due to lateral subsurface flow to the river store at rate $\phi_{\text{lat}}\ f_{\text{sat}}(t)$, percolation to the ground water store at rate $\phi_{\text{gw}}\ f_{\text{sat}}(t)$, and evapotranspiration at rate $f_{\text{et}}(t)P(t)$, where $P(t)$ is the potential evapotranspiration (mm/day), and the ratio of actual to potential evapotranspiration is related to the soil saturation by the model

$$f_{\text{et}}(t) = 1 - \exp(-\phi_{\text{et}} h_{\text{soil}}(t)).$$

Mathematically,

$$\frac{\mathrm{d}h_{\text{soil}}}{\mathrm{d}t} = (1 - f_{\text{sat}}(t))R(t) - \phi_{\text{lat}}\ f_{\text{sat}}(t) - \phi_{\text{gw}}\ f_{\text{sat}}(t) - f_{\text{et}}(t)P(t).$$

2. The ground water store (deep aquifers) is a linear reservoir with depth $h_{\text{gw}}(t)$ (mm). The depth increases due to percolation from the soil at rate $\phi_{\text{gw}}\ f_{\text{sat}}(t)$, and decreases due to base flow to the river store at rate $\phi_{\text{bf}}h_{\text{gw}}(t)$, and percolation to deep aquifers at rate $\phi_{\text{dp}}h_{\text{gw}}(t)$:

$$\frac{\mathrm{d}h_{\text{gw}}}{\mathrm{d}t} = \phi_{\text{gw}}\ f_{\text{sat}}(t) - (\phi_{\text{bf}} + \phi_{\text{dp}})h_{\text{gw}}(t).$$

3. The river water store temporarily delays the water flow in the river, and is modelled as a linear reservoir of depth $h_{\text{river}}(t)$ (mm). The depth increases due to surface runoff at rate $R(t)f_{\text{sat}}(t)$, lateral subsurface flow at rate $\phi_{\text{lat}}f_{\text{sat}}(t)$, and base flow from groundwater at rate $\phi_{\text{bf}}h_{\text{gw}}(t)$. It decreases due to river flow out of the watershed at rate $\phi_{\text{r}}h_{\text{river}}(t)$:

$$\frac{\mathrm{d}h_{\text{river}}}{\mathrm{d}t} = R(t)f_{\text{sat}}(t) + \phi_{\text{lat}}f_{\text{sat}}(t) + \phi_{\text{bf}}h_{\text{gw}}(t) - \phi_{\text{r}}h_{\text{river}}(t).$$

The final output of the simulator is the river flow, $Q_r(t)$, and is the product of the watershed area A_w and the river runoff flux $\phi_{\text{r}}h_{\text{river}}(t)$:

$$Q_r(t) = A_w\phi_{\text{r}}h_{\text{river}}(t).$$

See Figure 2 for a visual representation of the simulator. The two external forcing functions relate to weather conditions for the day in question. They are the rain, $R(t)$, and the potential evapotranspiration, $P(t)$. Only the daily sums of the two variables were available and so we represent daily intensity of rainfall and potential evapotranspiration as step functions in the numerical solver applied to the differential equations. There are eight simulator parameters, denoted $\phi_\cdot$, which we fixed at values estimated (using data from the Abercrombie catchment) in Reichert and Mieleitner (2009), with $\phi_{\text{s}} = 0.02$, $\phi_{\text{F}} = 125$, $\phi_{\text{et}} = 0.016$, $\phi_{\text{lat}} = 1.5$, $\phi_{\text{gw}} = 4.9$, $\phi_{\text{bf}} = 0.0002$, $\phi_{\text{r}} = 0.6$, and $\phi_{\text{dp}} = 0.02$. In a more comprehensive analysis of this simulator, we may wish to let these parameters vary and estimate them along with the discrepancy function. However, for the purposes of this paper, we suppose we are given a calibrated simulator which we treat as a black-box, that takes the current state vector, rain, and potential evapotranspiration as inputs, and outputs a prediction of the state vector for the following day. We then attempt to characterize and quantify the discrepancy for the black-box simulator.

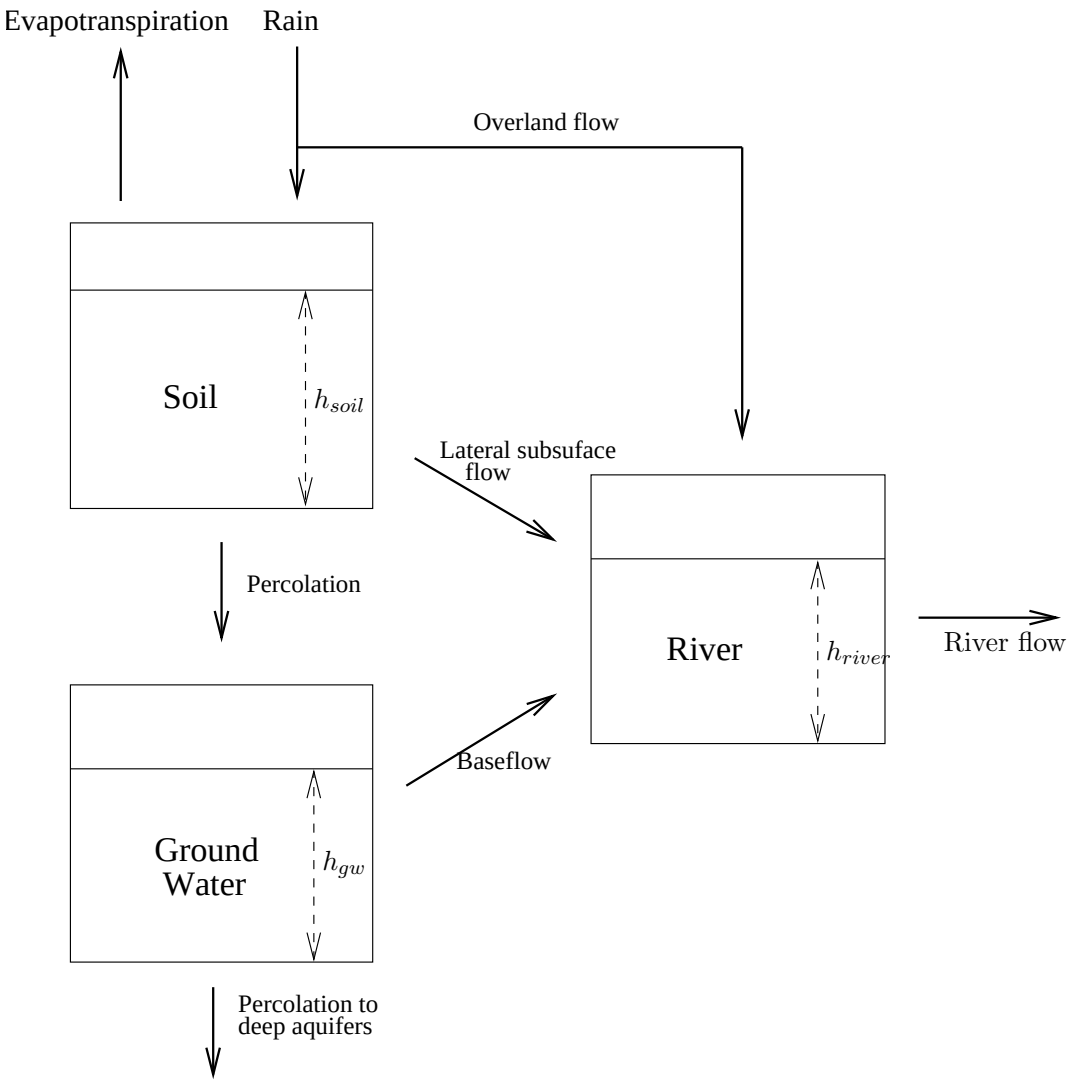

Figure 2: A visual representation of the logSPM simulator.

Data are available from the Abercrombie watershed in New South Wales, Australia, from the year 1972 to 1976. Of the three state variables, only a function of the river flow $h_{\text{river}}(t)$ is observed, which again highlights the difficulty faced when quantifying model error: noisy observations of one of the three state vectors are used to estimate the uncertainty in the dynamics of all three quantities. Reichert and Mieleitner (2009) and Kuczera et al. (2006) examined the logSPM simulator for the Abercrombie watershed using the same data as we use below. Their approach to dealing with simulator structural error differs from the approach we take here. Both approaches focused on allowing the simulator parameter values (ϕ) to change through time: Kuczera et al. (2006) looked for storm dependence in the parameter values; Reichert and Mieleitner (2009) used stochastic model parameters and introducted multipliers onto the forcing terms to correct for input errors, and then inferred the implied dynamics of the parameters through time. We prefer to take a different approach and use constant (calibrated) simulator parameters, and instead look to learn a functional form for the simulator discrepancy for this choice.

Our statistical framework for relating the simulator to the observations, can be broken down into two parts. We start by relating the simulator dynamics to the system, before then describing a model relating the system to the observations. For the discrepancy model we used a linear combination of the three state variables and the two forcing functions, a constant bias term, plus white noise Gaussian residuals for each of the three dimensions in the dynamics:

$$\begin{aligned}\boldsymbol{\delta}(\mathbf{h},\mathbf{u}) &= \begin{pmatrix} \delta_s(\mathbf{h},\mathbf{u}) \\ \delta_{gw}(\mathbf{h},\mathbf{u}) \\ \delta_r(\mathbf{h},\mathbf{u}) \end{pmatrix} + \boldsymbol{\epsilon} \\ &= \begin{pmatrix} a_s + \mathbf{b}_s^\top \mathbf{h} + \mathbf{c}_s^\top \mathbf{u} \\ a_{gw} + \mathbf{b}_{gw}^\top \mathbf{h} + \mathbf{c}_{gw}^\top \mathbf{u} \\ a_r + \mathbf{b}_r^\top \mathbf{h} + \mathbf{c}_r^\top \mathbf{u} \end{pmatrix} + \boldsymbol{\epsilon}\,, \end{aligned} \tag{18}$$

where $\mathbf{h} = [h_{\text{s}}\ h_{\text{gw}}\ h_{\text{river}}]^\top \in \mathbb{R}^3$ is the state vector and $\mathbf{u} = [R\ P]^\top \in \mathbb{R}^2$ represents the input forcing functions (rain and potential evapotranspiration). The linear parameters for the soil dynamics discrepancy are grouped in the vectors $\mathbf{b}_s = [b_{s,1}\ b_{s,2}\ b_{s,3}]^\top \in \mathbb{R}^3$ and $\mathbf{c}_s = [c_{s,1}\ c_{s,2}]^\top \in \mathbb{R}^2$, whilst the constant bias is given by the scalar a_s. Similarly, the other vectors $\mathbf{b}_{gw}$, $\mathbf{b}_r$, $\mathbf{c}_{gw}$, $\mathbf{c}_r$ represent the same coefficients for the ground water and river state dynamics with a_{gw} and a_r denoting the constant biases. The remaining uncertainty is captured through Gaussian white noise, with

$\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ where

$$\boldsymbol{\mu} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \qquad \text{and} \qquad \Sigma = \begin{pmatrix} \sigma_s^2 & 0 & 0 \\ 0 & \sigma_{gw}^2 & 0 \\ 0 & 0 & \sigma_r^2 \end{pmatrix} . \tag{19}$$

More complex choices, such as non-diagonal choices for Σ, heteroscedastic variances, and more complex structural forms in Equation (18) can be considered, and will be discussed later.

We follow Kuczera et al. (2006) when modelling the measurement process for the catchment, and assume $Q_r = A_w \phi_{\mathrm{r}} h_{\mathrm{river}}(t)$, where $A_w = 2770\mathrm{km}^2$ is the area of the Abercrombie catchment. To reduce the heteroscedasity of the residuals, we follow Reichert and Mieleitner (2009) and apply a transformation (Box and Cox, 1964) to the observations and predicted system value. We assume independent identically distributed Gaussian measurement error on the transformed river flow, $\log(Q_r + \lambda)$, so that

$$\log(Q_r + \lambda) \sim \mathcal{N}(\log(A_w \phi_{\mathrm{r}} h_{\mathrm{river}}(t) + \lambda), s^2), \tag{20}$$

where we take the measurement variance to be $s^2 = 0.1$. The effect of applying the logarithmic transformation to the data is to induce a heteroscedastic variance on the measurement process, so that on days with small average river flows the measurements are assumed to have a smaller variance than on days for which the average river flow was large.

To train the discrepancy model $\boldsymbol{\delta}(\mathbf{h}, \mathbf{u})$, we used a half year period (180 days) of contiguous observations from the Abercrombie dataset (observations from 16 June 1975 till 11 December 1975). We used $N = 2000$ filtering particles and $M = 50$ smoothed trajectories in the algorithm described in Section 2.2 and the appendix. We tested various starting points for the parameters in the MCEM algorithm, and the converged estimates were similar in each case. Some variation in the estimated values is observed due to the Monte Carlo nature of the approximation used in Equation (8) (this could be reduced by increasing N and M), but we found that this variation did not have a large effect on the predictive power of the forecasting system. The estimated maximum-likelihood values are given in Table 3. Notice that the estimated variance term for the river discrepancy function is several orders of magnitude smaller than for the soil or ground water discrepancy. This should not be surprising, as we observe the river flow, but not the other two water stores. For a similar reason, it appears that the most important aspect of the discrepancy term are the coefficients of the river flow $b_{\cdot,3}$, and the forcing function coefficients. Inferring relationships involving observed quantities (rain, potential evapotranspiration, and river flow) is easier than inferring relationships involving the unobserved soil and ground water stores.

Dimension	Bias	Linear parameters			Forcing coeffs.		Variance
	$a_\cdot$	$b_{\cdot,1}$	$b_{\cdot,2}$	$b_{\cdot,3}$	$c_{\cdot,1}$	$c_{\cdot,2}$	$\sigma^2_\cdot$
Soil δ_s	12.7803	-0.0662	0.0740	0.8091	-0.6254	-2.0863	29.7519
Ground water δ_{gw}	6.7218	-0.0205	0.0362	-0.8516	-0.0766	-1.5297	2.7294
River δ_r	-0.2111	0.0022	-0.0019	-0.0487	0.0034	0.0384	0.0005

Table 3: Estimated maximum likelihood parameters for the discrepancy function described by Equation (18). Each row describes the parameter values for the discrepancy function in the dynamics of one of the three state variables representing the three conceptual water stores in the logSPM simulator.

These parameter estimates are not particularly informative by themselves. To assess the impact of our efforts we need to examine the predictive performance of the forecasting system. We do this by reporting the mean square error (MSE), the Nash-Sutcliffe statistic (NS), the Dawid score (DS) and the continuously ranked probability skill score (CRPSS) as in the previous section. We use the bootstrap particle filter (see the Appendix) to find $\pi(\mathbf{h}_t|Q_{1:t})$ and then run the system forwards in time to find the one- and five-step-ahead predictions, which can then be compared with the observations. The particle filter represents $\pi(\mathbf{h}_t|Q_{1:t})$ as a weighted sample of particles $\{W_t^{(i)}, \mathbf{h}_t^{(i)}\}$. We propagate each particle $\mathbf{h}_t^{(i)}$ through the system dynamics (Equation (23)) k times to get a weighted sample of particles $\{W_t^{(i)}, \mathbf{h}_{t+k}^{(i)}\}$ which approximate the density $\pi(\mathbf{h}_{t+k}|Q_{1:t})$. The predictive densities used in the evaluation are the distribution of the observations, and so the particles need to be propagated through the measurement process. These are found by taking just the river state-vector, $h_{\text{river},t+k}$, applying the Box-Cox transformation described by Equation (20), adding Gaussian noise, and applying the inverse Box-Cox transformation. Let Q_{t+k}^{rep} denote theoretical replications of the $(t+k)^{th}$ observation, assuming the statistical framework is true, so that

$$Q_{t+k}^{rep,(i)} = \exp\left[\log(A_w \phi_{\text{r}} h_{\text{river},t+k} + \lambda) + u_i\right] - \lambda, \tag{21}$$

where $u_i \sim \mathcal{N}(0, s^2)$ independently for each i. This gives a weighted sample of points $\{W_t^{(i)}, Q_{t+k}^{rep,(i)}\}$ that approximates the predictive distribution $\pi(Q_{t+k}^{rep}|Q_{1:t})$, which can then be compared to the observed value Q_{t+k}. The scores require the predictive mean and variance for the k step ahead forecast

made at time t, which are given by

$$m_t(k) = \sum_{i=1}^{N} W_t^{(i)} Q_{t+k}^{rep,(i)}$$

$$s_t(k) = \frac{1}{1-V} \sum_{i=1}^{N} W_t^{(i)} (Q_{t+k}^{rep,(i)} - m_t(k))^2,$$

where the second equation is the unbiased estimator of the weighted variance, where $V = \sum_{i=1}^{N} (W_t^{(i)})^2$ and $\sum_{i=1}^{N} W_t^{(i)} = 1$. These terms can then be used to calculate the four scores discussed in Section 2.3.

We compare the performance of three different forecasting systems:

(ODE) logSPM with measurement process only (no simulator error). A common assumption made when using complex simulators is to assume that the observations arise from the simulator prediction plus measurement error, ignoring any simulator discrepancy. We use this forecasting system as the benchmark against which we measure any improvements made by quantification of the simulator discrepancy. We generate predictions of this system by running the deterministic logSPM simulator and propagating the prediction through the observation process described by Equation (21). The observation process is applied N times to get an ensemble comparable with that generated by the other forecasting systems.

(VAR) logSPM plus a white noise simulator discrepancy and measurement process. We assume no deterministic bias in the model discrepancy (setting $\mathbf{a}_\cdot = \mathbf{b}_\cdot = \mathbf{c}_\cdot = 0$ in Equation (18)) and use system dynamics

$$\mathbf{h}_{t+1} = \mathbf{f}(\mathbf{h}_t, \mathbf{u}_t) + \boldsymbol{\epsilon}_t \ , \quad \text{with} \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, D) \ . \tag{22}$$

where D is a diagonal matrix. We estimated the variances to be $\sigma_s^2 = 97.6929$, $\sigma_{gw}^2 = 4.4354$ and $\sigma_r^2 = 0.0004$ using the same MCEM algorithm. These values are larger than the estimated variances for the linear discrepancy model (Table 3), as expected.

(FULL) logSPM plus full discrepancy model and measurement process. We assume the system dynamics are described by

$$\mathbf{h}_{t+1} = \mathbf{f}(\mathbf{h}_t, \mathbf{u}_t) + \boldsymbol{\delta}(\mathbf{h}_t, \mathbf{u}_t). \tag{23}$$

During the assessment phase, the parameter estimates for $\boldsymbol{\delta}$ remain fixed at the values shown in Table 3.

Tables 4 and 5 show the results from assessing the three forecasting system performances on the training data (data from 16 June 1975 to 11 December 1975), for the one- and five-step-ahead predictions. We can see that the system that uses the full discrepancy model (Equation (23)) out performs the other two systems on all four measures. Recall that smaller values of MSE and DS are desirable, and that NS and CRPSS vary between minus infinity and 100%, with a perfect simulator achieving a score of 100%. The inclusion of any simulator discrepancy, VAR or FULL, leads to superior predictions over the simulator only system (ODE). The use of the full discrepancy model (FULL) does bring improvement over the variance only model (VAR), but not by a great amount. Figure 3 shows the fitted residuals for the ODE and FULL forecast systems. Both plots show evidence of correlated residuals showing that further modelling improvements could still be made, although the correlation is less extreme when using the full discrepancy. The simulator only residuals are not centred around zero showing a systematic departure from the modelling assumptions, whereas the residuals using the discrepancy model are centred around the line $y = 0$, as would be expected if the model were true. Also plotted are dashed lines showing two standard deviations either side of $y = 0$, at $y = \pm 2s$ where s is the standard deviation of the measurement process. If the assumed level of measurement error is accurate, then we would expect approximately 95% of the 180 observations to lie within these two dashed lines if the simulator was perfect. This occurs for the full discrepancy forecasting system, but is clearly not the case for the simulator only system (ODE).

If we test the forecasting systems on an independent data set, i.e., on data that was not used in the training procedure, then the results are not always so positive and it is possible to make worse predictions using the full discrepancy model than when simply using the simulator only. For example, testing the forecasting systems on the same period, but from the year 1976, yields a CRPSS of 60% for the ODE system, but a value of only 21% for the FULL system (VAR scores best with 81.4%), which is superior to climatology, but poorer than the deterministic ODE model. There are a few reasons why we believe we see this drastic drop off in performance. The first is that the results here were obtained after fitting the model to a short period of only 180 days. As found in Kuczera et al. (2006), the simulator discrepancy is largest during periods of high rainfall (storms). For the training data used there was essentially only a single large storm during this time, and so it seems likely that we have over-fit the model. By using a longer training period of data collected during more representative conditions, we hope to be able to solve the problem of overfitting. We also found evidence of seasonal dependence, with the simulator discrepancy taking a different form in summer months to

One step ahead predictions ($k = 1$)				
	MSE	NS (%)	DS	CRPSS (%)
ODE	0.2764	74.6	0.4619	73.2
VAR	0.1547	85.8	-0.7851	81.6
FULL	0.0988	90.9	-1.1955	85.0

Table 4: Validation results for the one-step-ahead forecasts for the three forecasting systems described in the text. ODE is the deterministic logSPM simulator, VAR is the simulator plus a white noise discrepancy, and FULL is the simulator plus the estimated discrepancy function. The four measures used are the mean square error (MSE), the Nash-Sutcliffe statistic (NS), the Dawid score (DS) and the continuously ranked probability skill score (CRPSS). The data used in the validation was a 180 day period (16 June 1975 till the 11 December 1975). The reference forecast used for the NS statistic and the CRPSS was a Gaussian distribution with mean and variance estimated from the observations (i.e., the climatological forecast).

Five step ahead predictions ($k = 5$)				
	MSE	NS (%)	DS	CRPSS (%)
ODE	0.2764	74.6	0.4619	73.2
VAR	0.1944	81.0	-0.5978	79.5
FULL	0.1035	89.9	-1.1559	84.5

Table 5: Validation results for the five-step-ahead forecasts for the three forecasting systems described in the text. The scores for the ODE system are the same as in Table 4.

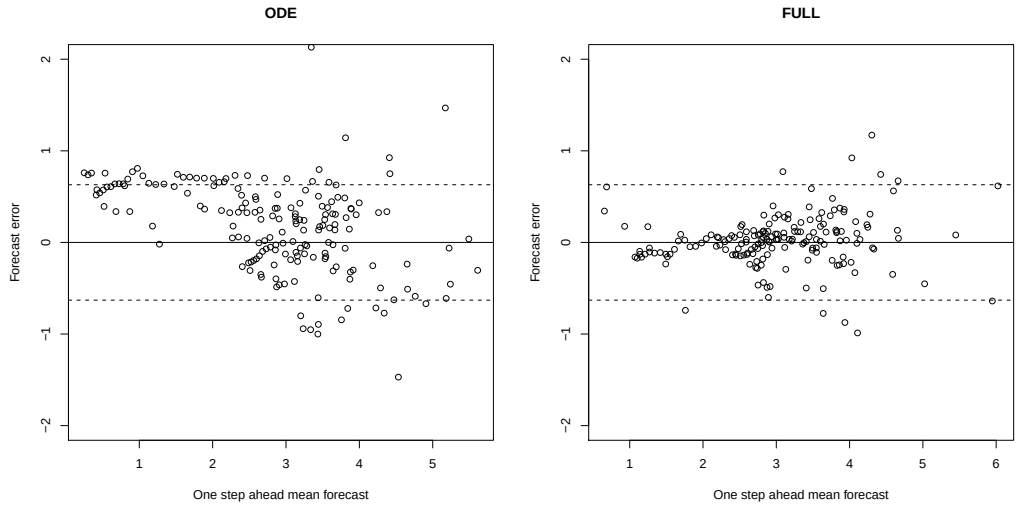

Figure 3: A residual plot showing the one-step-ahead transformed forecast errors, $\log(Q_{t+1} + \lambda) - \tilde{m}_t(1)$, versus the fitted values, $\tilde{m}_t(1) = \log(A_w k_r h_{\text{river},t+k} + \lambda)$). The plot on the left is for the ODE forecasting system with no simulator discrepancy term, and the plot on the right is for the full discrepancy model. A forecasting system which had no simulator discrepancy would have a residual plot that looked like an uncorrelated band of residuals distributed about the line $y = 0$. The dashed lines are two standard deviations (of measurement error) either side of $y = 0$, giving bounds within which we would expect to see approximately 95% of the 180 points if the simulator were perfect.

that found in the winter months. We can attempt to correct this by either fitting separate discrepancy functions during the different seasons (assuming we have enough data to do this), or by including an element of seasonal dependence into the structural form of the discrepancy.

Finally, it should be noted that the discrepancy model used is extremely simple. Extending the model to allow heteroscedastic variances in the discrepancy model (i.e., making $\mathbb{V}\mathrm{ar}(\epsilon)$ state dependent in Equation (19)) either through the use of generalised linear models, or through another normalising transformation, may lead to an improvement in the quantification of uncertainty. As mentioned above, the discrepancy is largest during storms, and relatively small during periods of minimal rain. The model we have fit here only allows for a single variance for the discrepancy, regardless of the weather, and so is a compromise between the two different situations. By allowing the variance to increase during periods of heavy rain, we would hopefully get more accurate and less variables predictions during dry periods, and more variable predictions that better quantified the uncertainty during wet periods. Another improvement in the modelling would be to allow for dependencies between the dimensions of the discrepancy function (i.e., using a non-diagonal Σ in Equation (19)) and using multivariate regression techniques. Finally, using a more complex, or non-parametric mean function (such as a Gaussian process) for the discrepancy in Equation (18) would allow us greater flexibility to capture any signal about the shape of the discrepancy function.

4 Discussion

A simulator can only ever approximate reality, so that there will always be a discrepancy between the system and the simulator prediction. If we wish to make predictions that take uncertainty into account then we must include some description of simulator discrepancy, as otherwise we will have an unrealistic level of confidence in our predictions. In this paper, we specified a statistical model of the discrepancy function and have then shown how to use a training period of simulator predictions and subsequent observations to calibrate the statistical model. The aim is to combine the scientific knowledge built into the simulator by expert modellers, with empirical knowledge learnt from past performance. This is a hard problem, as typically we are trying to infer errors in the dynamics of variables that are never observed. For complex simulators with a state vector of higher dimension than the observation, we may find that it is hard to improve on a simple white noise model unless high quality data are available.

We intend this paper to be the starting point in the development of methods for learning the simulator discrepancy in discrete time dynamical simulators. The focus here was on simple linear models for δ with homoscedastic error. Several immediate extensions are possible within this framework, such as the use of general linear models to allow heteroscedastic errors with state dependent variance, as well as allowing for correlation between different dimensions of the discrepancy function. The approach can incorporate situations in which there are a limited number of missing observations, but does require there to be a common interval between most observations, and so would not be suitable for situations in which observations occur irregularly in time.

The method proposed here is computationally expensive, as it requires the repeated use of a particle filter embedded within the EM algorithm, which in turn requires repeated draws from the simulator. For expensive dynamical simulators, we could dynamically emulate the simulator as described in Conti et al. (2009), and use the emulator as a cheap statistical surrogate for the simulator to decrease computation time. Because of the desire to avoid running the particle filter an excessive number of times, we used a maximum likelihood approach to estimate the parameters in the discrepancy function. Fixing the parameters at their maximum likelihood values ignores parametric uncertainty in our estimate of the simulator discrepancy. This could be avoided with a Bayesian approach, but at the expense of further computation. This paper focused solely on quantifying simulator discrepancy, not on simulator calibration. In the case where we also wished to estimate uncertain simulator parameters we could either calibrate the simulator before fitting the discrepancy model, as done in this paper, or attempt to jointly infer both sets of parameters. A joint approach is preferable, but raises computational and statistical problems and has not been considered in this paper. We suspect that in most problems a high degree of non-identifiability would exist among the simulator and discrepancy parameters.

Finally, the catchment area simulator studied here has a lumped or box structure and is thus intrinsically non-spatial. However, many simulators of dynamic environmental systems originate from conservation laws in both space and time, and thus intrinsically have spatial characteristics in addition to their dynamic properties. Such spatially extended simulators are typically characterised by high dimensional state spaces arising from the discretization of the partial differential equations at their cores. Developing discrepancy models for such simulators remains very challenging. Most existing attempts assume the discrepancy can be characterised via parameter uncertainty within the simulators; however, for environmental systems it seems likely that any simulator will also omit some potentially relevant

processes and characteristics, meaning that parameter uncertainty alone will not explain the model discrepancy with respect to reality.

Developing discrepancy models for spatially distributed simulators would either require dense (in space and time) observations, or strong prior knowledge of the likely discrepancy function form. Issues of identifiably, particularly where there is also simulator parameter uncertainty, are likely to require careful modelling. Several possible approaches might be considered for addressing spatially distributed discrepancy. Where dense observations are available, for example in a heavily instrumented catchment, or measurement campaign, then the approaches presented in this paper could be applied, replacing the regression functions in the discrepancy term (Equation (18)) with spatially distributed functions, such as radial basis functions, or spatial splines. This would maintain the relative simple parametric form for the discrepancy, but introduces the challenge of locating and setting the number of basis functions/knot points. Alternative representations of the 'spatial' discrepancy process could use spatial Gaussian processes with an appropriate parametrization, for example as in Csató and Opper (2002). Further work is needed to explore whether such methods can realistically be applied to the complicated spatial distributed simulators used widely in addressing some of the key environmental issues currently facing society.

References

Beven, K., 2006. A manifesto for the equifinality thesis. Journal of Hydrology 320, 18–36.

Box, G. E. P., Cox, D. R., 1964. An analysis of transformations. Journal of the Royal Statistical Society. Series B (Methodological) 26, 211–252.

Caffo, B. S., Jank, W., Jones, G. L., 2005. Ascent-based Monte Carlo expectation-maximization. Journal of the Royal Statistical Society, Series B (Methodological) 67 (2), 235–251.

Conti, S., Gosling, J., Oakley, J., O'Hagan, A., 2009. Gaussian process emulation of dynamic computer codes. Biometrika 96 (34), 663–676.

Crucifix, M., Rougier, J., 2009. On the use of simple dynamical systems for climate predictions. The European Physical Journal - Special Topics 174, 11–31.

Csató, L., Opper, M., 2002. Sparse on-line Gaussian processes. Neural Computation 14, 641–668.

Dawid, A. P., 1986. A Bayesian view of statistical modelling. In: Goel, P. K., Zellner, P. (Eds.), Bayesian inference and decision techniques. Elsevier Science, pp. 391–404.

Dawid, A. P., Sebastiani, P., 1999. Coherent disperson criteria for optimal experimental design. The Annals of Statistics 27, 65–81.

Dempster, A. P., Laird, N. M., Rubin, D. B., 1977. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society. Series B (Methodological) 39 (1), 1–38.

Doucet, A., de Freitas, N., Gordon, N., 2001. Sequential Monte Carlo Methods in Practice. Springer.

Gneiting, T., Raftery, A. E., 2007. Strictly proper scoring rules, prediction, and estimation. Journal of the American Statistical Association 102 (477), 359–378.

Godsill, S., Doucet, A., West, M., 2004. Monte Carlo smoothing for nonlinear time series. Journal of the American Statistical Association 99 (465), 156–168.

Goldstein, M., Rougier, J., 2009. Reified Bayesian modelling and inference for physical systems (with discussion). Journal of Statistical Planning and Inference 139, 1221–1239.

Gordon, N. J., Salmond, D. J., Smith, A. F. M., 1993. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. Radar and Signal Processing, IEE Proceedings F 140, 107–113.

Griffith, A. K., Nichols, N. K., 1996. Accounting for model error in data assimilation using adjoint methods. In: Berz, M., Bischof, C., Corliss, G., Greiwank, A. (Eds.), Computational Differentiation: Techniques, Applications and Tools. SIAM, Philadelphia, pp. 195–204.

Griffith, A. K., Nichols, N. K., 2000. Adjoint techniques in data assimilation for estimating model error. Journal of Flow, Turbulence and Combustion 65, 469–488.

Higdon, D., Gattiker, J., Williams, B., Rightley, M., 2008. Computer model calibration using high-dimensional output. Journal of the American Statistical Association 103, 570–583.

House, L., Goldstein, M., Rougier, J., 2011. Assessing model discrepancy using a multi-model ensemble. In submission.

Jolliffe, I. T., Stephenson, D. B., 2003. Forecast Verification: A Practitioner's Guide in Atmospheric Science. Wiley and Sons, Chichester.

Kavetski, D., Kuczera, G., Franks, S. W., 2003. Semi-distributed hydrological modelling: a 'saturation path' perspective on TOPMODEL and VIC. Water Resources Research 39, 1246–1253.

Kennedy, M. C., O'Hagan, A., 2001. A Bayesian calibration of computer models (with discussion). Journal of the Royal Statistical Society. Series B (Methodological) 63, 425–464.

Kitagawa, G., 1996. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. Journal of Computational and Graphical Statistics 5, 1–25.

Kuczera, G., Kavetski, D., Franks, S., Thyer, M., 2006. Towards a Bayesian total error analysis of conceptual rainfall-runoff models: Characterising model error using storm-dependent parameters. Journal of Hydrology 331, 161–177.

Meng, X. L., Rubin, D. B., 1991. Using EM to obtain asymptotic variance-covariance matrices: the SEM algorithm. Journal of the American Statistical Association 86, 899909.

Nash, J. E., Sutcliffe, J. V., 1970. River flow forecasting through conceptual models part I - a discussion of principles. Journal of Hydrology 10, 282–290.

Oakley, J. E., O'Hagan, A., 2002. Bayesian inference for the uncertainty distribution of computer model outputs. Biometrika 89 (4), 769–784.

Oberkampf, W. L., Trucano, T. G., 2008. Verification and validation benchmarks. Nuclear Engineering and Design 238, 716–743.

Reichert, P., Mieleitner, J., 2009. Analyzing input and structural uncertainty of nonlinear dynamic models with stochastic time-dependent parameters. Water Resources Research 45, 1–19.

Saltelli, A., Chan, K., Scott, M. (Eds.), 2000. Sensitivity Analysis. Wiley, New York, USA.

Shmueli, G., 2011. To explain or to predict? Statistical Science.

Smith, R., Tebaldi, C., Nychka, D., Mearns, L., 2009. Bayesian modeling of uncertainty in ensembles of climate models. Journal of the American Statistical Association 104, 97–116.

Strong, M., Oakley, J. E., Chilcott, J., 2011. Managing structural uncertainty in health economic decision models: a discrepancy approach. Journal of the Royal Statistical Society. Series C (Applied Statsitics), In press.

Vernon, I. R., Goldstein, M., Bower, R. G., 2010. Galaxy formation: a Bayesian uncertainty analysis. Bayesian Analysis 5, 619–670.

Wei, G. C. G., Tanner, M. A., 1990. A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. JASA 85 (411), 699–704.

5 Appendix

To sequentially generate observations from $\pi(x_{1:t} \mid y_{1:t})$ we use the bootstrap particle filter (Doucet et al., 2001):

Bootstrap particle filter

t = 1 (i) Initialize: For $i = 1, \ldots, N$

$$x_1^{(i)} \sim \pi(x_1)$$
$$w_1^{(i)} = \pi_1(y_1 \mid x_1^{(i)})$$

(ii) Normalise: set $W_1^{(i)} = \frac{w_1^{(i)}}{\sum_{j=1}^N w_1^{(j)}}$, to obtain a weighted sample $\{W_1^{(i)}, x_1^{(i)}\}$ approximating $\pi(x_1 \mid y_1)$.

(iii) Resample: if ESS$< T$, resample the particles and set $W_1^{(i)} = 1/N$ for all i. Set $t = 2$.

t ≥ 2 (i) Simulate: For $i = 1, \ldots, N$

$$x_t^{(i)} \sim \pi(x_t \mid x_{t-1}^{(i)})$$
$$w_t^{(i)} = W_{t-1}^{(i)} \pi(y_t \mid x_t)$$

(ii) Normalise: set $W_t^{(i)} = \frac{w_t^{(i)}}{\sum_{j=1}^N w_t^{(j)}}$, to obtain a weighted sample $\{W_t^{(i)}, x_{1;t}^{(i)}\}$ approximating $\pi(x_{1:t} \mid y_{1:t})$.

(iii) Resample: if ESS$< T$, resample and set $W_t^{(i)} = 1/N$ for all i. Set $t = t + 1$.

All distributions are conditional on θ. The effective sample size (ESS) of the weighted sample is $\left(\sum_{i=1}^{N}(W_t^{(i)})^2\right)^{-1}$, and T denotes the threshold we use to decide when to resample (typically we take $T = N/2$). We used a systematic resampling scheme (Kitagawa, 1996) throughout. To get a smoothed trajectory from $\pi(x_{1:t} \mid y_{1:T})$ we pick a single trajectory from $\{x_{1:T}^{(i)}\}$ according to the weights $\{W_T^{(i)}\}$.