A comparison of Latin hypercube and grid ensemble designs for the multivariate emulation of an Earth system model

Nathan M. Urban*

Department of Geosciences, The Pennsylvania State University, University Park, PA 16802, USA

Thomas E. Fricker[1]

Department of Probability and Statistics, University of Sheffield, Sheffield S3 7RH, UK

Abstract

A statistical emulator is a fast proxy for a complex computer model which predicts model output at arbitrary parameter settings from a limited ensemble of training data. Regular grid designs for the training set are commonly used for their simplicity. However, Latin hypercube designs have well known theoretical advantages in the design of computer experiments, especially as the dimension of the parameter space grows.

Here we use time series output from a simple Earth system model to compare the influence of these two design choices on the cross-validation prediction skill of a statistical emulator. We find that an emulator trained on a Latin hypercube design shows a small but clear improvement in prediction quality relative to an emulator trained on a grid design. We also find that the Latin hypercube emulator is more accurate than the grid emulator in single-parameter model sensitivity studies. We conclude with a discussion of ensemble design choices for emulator computer experiments.

Key words: climate model, statistical emulator, design of computer experiments, ensemble, Latin hypercube, kriging

1. Introduction

In climate science it is common to make use of highly complex computer simulations, such as coupled atmosphere-ocean general circulation models (GCMs). GCMs have been coupled to biogeochemical or land surface models to produce Earth system models. GCMs require high spatial and temporal resolutions in order to represent important physical climate processes. They are computationally intensive, sometimes requiring months of computer time to simulate centuries of real time.

The output of complex models is dependent on many parametric assumptions, such as the strength of climate system feedbacks, or the rate of heat transport from the atmosphere to the oceans. In order to examine the sensitivity of a model to its input parameters, or to quantify the parametric uncertainty in model projections, it is necessary to evaluate the model at a variety of plausible parameter settings. The formidable computational requirements of GCMs preclude a full exploration of their parameter spaces.

A simpler class of models, the Earth system models of intermediate complexity (EMICs), has been developed in part to address this problem (Claussen et al., 2002). EMICs require only days to weeks of computer time for centennial scale projections. It is possible to evaluate an EMIC with an ensemble of runs comprising a few hundred combinations of a handful of parameters. Scientific uses of EMIC ensembles include, for example, probabilistic projection of future climate and biogeochemical cycles (Knutti et al., 2003; Matthews and Keith, 2007; Sokolov et al., 2009), quantification of past climate feedbacks and biogeochemical cycles (Schneider von Deimling et al., 2006; Ridgwell et al., 2007; Panchuk et al., 2008), and estimation of ocean mixing processes (Schmittner et al., 2009). The number of parameters which can be realistically varied in a finite ensemble is far less than the num-

*Corresponding author. Postal address: 411 Deike Building, Penn State, University Park, PA 16802, USA. Tel.: 1 (814) 863-9903, Fax: 1 (814) 863-7823

Email addresses: nurban@psu.edu (Nathan M. Urban), T.Fricker@sheffield.ac.uk (Thomas E. Fricker)

ber of adjustable parameters in the model. However, the model behavior of interest is frequently dominated by a relatively small subset of key physical parameters.

In a thorough sensitivity or uncertainty analysis, it is often the case that the ensemble size is only barely large enough to span the parameter space of interest. (If larger ensembles are available, a thorough study will typically expand the number of considered parameters until the ensemble size is, again, minimally adequate to span the larger parameter space.) With a limited ability to sample the space of parameters, the design of an ensemble requires careful consideration. The location and spacing of design points must be chosen to provide sufficient information about the model response across the parameter space.

Emulators have been developed to make maximum use of the model information contained in an ensemble. An emulator is a fast surrogate for a computationally expensive computer model, which is trained on an ensemble of model output. It interpolates the ensemble to predict the model output at parameter settings not included in the design. A statistical emulator is an emulator which can estimate the uncertainty in its predictions. Since an emulator is an approximation to the true model, it is important to assess the quality of this approximation. The ability of statistical emulators to produce error bars for their predictions is therefore valuable.

Examples of emulators (both statistical and non-statistical) include spline interpolation, parametric regression, artificial neural networks, and nonparametric Gaussian process regression. Our discussion of emulators focuses exclusively on the last type of emulator, and we use the term "emulator" to refer specifically to this method of statistical emulation. Gaussian process (GP) emulators (Kennedy and O'Hagan, 2001) are a generalization of kriging, a geostatistical interpolation technique (Cressie, 1993; Wackernagel, 2003; Banerjee et al., 2004). Kriging is used to interpolate point measurements over physical space. A GP emulator interpolates model output over parameter space. In geostatistical terms, a multivariate GP emulator is analogous to a form of Bayesian universal cokriging.

Uses of ensembles to explore climate model parameters range from very large ($\sim 10^5$) ensembles of relatively simple models (Tomassini et al., 2007) to small (~ 50) ensembles of more complex models (Ridgwell et al., 2007). Applications of non-statistical climate model emulators have recently emerged (e.g., Knutti et al., 2003). The use of GP statistical emulators in climate science is relatively new (*cf.* Challenor et al., 2006; Sansó et al., 2008).

Two ensemble types which have been extensively used in the design of computer experiments are grids (also known as Cartesian products) and Latin hypercubes (LHs) (McKay et al., 1979). Latin hypercube designs (LHDs) have good theoretical properties when compared to grids, as discussed in the next section. However, the practical effect of different designs on the emulation of a climate or Earth system model is not well studied. Here we compare the predictive skill of a multivariate climate model emulator trained on grid and Latin hypercube ensemble designs.

2. Design choices

A common ensemble design is a regular grid (Figure 1, left panel). A few equally spaced values are selected for each parameter. Every combination of values for each parameter is included in the grid design. Grid designs are popular for several reasons. First, they are the simplest possible designs, and require no expert judgment other than the parameter ranges and ensemble size. Second, they facilitate single-parameter sensitivity studies. To determine the response of a model to changes in a single parameter, it is possible to move from point to point in a grid along a single dimension, holding all other parameters fixed. Third, it is easy to integrate quantities over a regular grid by Riemann summation. This is important in the Bayesian calibration of computer models (Kennedy and O'Hagan, 2001), where marginalization integrals are necessary to calculate the observationally calibrated probability distributions of individual parameters (Oakley and O'Hagan, 2002).

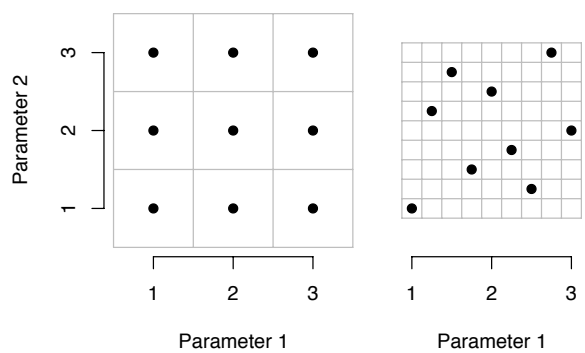

Figure 1: Regular grid (*left*) and Latin square (*right*) designs for a $p = 2$ dimensional, $N = 9$ member ensemble.

Given these advantages, simple grids are popular design choices. However, they have several drawbacks (Santner et al., 2003; Husslage et al., 2008). A key limitation of grids is their "collapsing" property: multiple grid points can have the same coordinate value

when projected onto a parameter axis. This is illustrated in Figure 1 (left panel), where three different grid points share each value of either parameter. In a two-dimensional grid, the collapsing property means that there is more than one design point in each row and column.

The collapsing property of grids has several consequences. First, the model is evaluated at only a few values for any parameter, since many grid points share the same small set of possible values. A grid design cannot reveal model behavior at any but a limited number of parameter settings, which can be problematic for emulation if the model output varies rapidly between design points. Second, the ensemble size grows exponentially with the number of parameters. For example, a four-parameter grid of size ≈ 100 can sample no more than three values for each parameter $3^4 = 81 \approx 100$. As in the first point, this severely limits the ability of a grid to sample intermediate parameter values. Conversely, for a limited ensemble size, a grid design can only explore low dimensional parameter spaces. Third, models are often more sensitive to some parameters than others. If an insensitive parameter is included in a grid design, many model evaluations are wasted altering a parameter whose value does not affect the model output. Due to the collapsing property, two grid points which differ only by an insensitive parameter are effectively the same point, because no other parameters which can affect the output are varied.

Latin hypercube (LH) designs have long been used as an alternative to grids in the design of computer experiments (for a review of ensemble design methods, see Santner et al., 2003). Like a regular grid, a Latin hypercube partitions each parameter range into equally spaced values. Unlike a grid, the number of partitions is equal to the size of the ensemble. Latin hypercubes are constructed to avoid the collapsing property of grids: no two LH design points share the same value for any parameter. In the case of a two-dimensional design (Latin square), this property is equivalent to each row and each column containing exactly one design point (Figure 1, right panel). Unlike grids, Latin hypercubes ensemble sizes need not grow exponentially with the dimensionality of the parameter space, permitting a LH design to explore more parameters than a grid using the same ensemble size.

Grid and Latin hypercube designs are compared in Figure 1. In both cases, the ensemble consists of $N = 9$ design points, with $p = 2$ parameters varied. Both designs cover the same range for each parameter, the interval [1,3]. The grid is evaluated at only three values of each of the two parameters. The Latin square is evaluated at nine different values of each parameter, the maximum number possible in an ensemble of size 9.

There are many different LH designs for a given parameter space, corresponding to different permutations of design points. (In Figure 1, right panel, there are many ways of placing exactly one point in each row and column.) A degenerate Latin square design puts all the points on the diagonal, which has obvious difficulty covering the entire space. We use "maximin" LH designs (Morris and Mitchell, 1995), generated using the GEM-SA software package,[1] which maximize the minimum distance between neighboring points. Intuitively, a maximin design spreads points as far away from each other as possible, to maximize their coverage of parameter space. This prevents design points from clustering too close together and over-representing some regions of parameter space. The GEM-SA software uses a stochastic simulated annealing algorithm (Morris and Mitchell, 1995) to generate a large number of candidate LH designs and chooses the one which best satisfies the maximin distance criterion.

It is important to note the differences between how Latin hypercube designs are used to train emulators, and how they traditionally have been used in model uncertainty analysis. Historically, Latin hypercubes have been commonly used as an efficient method to randomly sample the model output from some prior probability distribution on parameter space. The marginal probability distribution for each parameter is "stratified" into equal probability bins, from which a Latin hypercube is constructed. Stratified LH sampling is a form of importance sampling and concentrates samples in high-probability regions of parameter space.

By contrast, an emulator simply needs to be trained to reproduce model output based on the model behavior at known inputs. Because the input parameters are not treated as uncertain, there is no need to sample randomly from parameter space or to specify any kind of prior probability distribution on model parameters. The Latin hypercubes discussed here are equivalent to stratified LH sampling from a bounded uniform distribution. However, there is no need for them to honor this particular distribution or any distribution. To train an accurate model emulator it may be preferable to concentrate design points in regions of parameter space where the model output is most variable, or the output covariance structure is most uncertain. These regions of parameter

[1] Kennedy, M.C., 2004. Description of the Gaussian process model used in GEM-SA. GEM-SA help documentation, Department of Probability and Statistics, University of Sheffield, Sheffield, 3 pp. `http://www.ctcd.group.shef.ac.uk/gem.html`.

space may have nothing to do with the parameter settings which the user may believe to be most probable.

3. Methods

3.1. Earth system model

Emulators are designed to be used with complex, computationally intensive models which can be evaluated at a limited number of parameter settings. For the purposes of this paper, we instead work with a computationally efficient simple Earth system model (Urban and Keller, 2010, in revision).[2] An emulator is unnecessary for this model, which requires less than a second of CPU time per model evaluation. However, its speed makes it possible to construct and evaluate many different ensemble designs, so for the purposes of this paper we treat its output as representative of a more complex EMIC we might wish to emulate.

We use the DOECLIM climate model, a zero-dimensional energy-balance model coupled to a one-dimensional diffusive ocean (Kriegler, 2005). The climate model is coupled to the NICCS nonlinear impulse response model of the terrestrial/ocean carbon cycle (Hooss et al., 2001). Together these two coupled models comprise our Earth system model. The coupled model is forced in years 1850–2000 with historic CO_2 emissions, non-CO_2 greenhouse gas and aerosol concentrations, solar irradiance, and volcanic forcings; these data are taken from references cited in Kriegler (2005). The model outputs three globally averaged annual time series: surface temperature anomaly, ocean heat anomaly, and atmospheric CO_2 concentration.

3.2. Ensemble design

We vary four parameters in the model: (1) Q_{10}, the temperature sensitivity of CO_2 respiration from microbial decomposition in soil (Davidson and Janssens, 2006); (2) κ, the vertical diffusivity of heat in the ocean (Hansen et al., 1985); (3) S, the equilibrium climate sensitivity (temperature response) to a doubling of atmospheric CO_2 (Knutti and Hegerl, 2008); and (4) α, a multiplicative scale factor on the cooling influence of industrial aerosols, including the indirect effect on clouds (Lohmann and Feichter, 2005). These parameters were chosen for their influence on the present and future responses of the coupled climate-carbon cycle system. We collectively refer to the parameters as $\{p_i\} = \{Q_{10}, \kappa, S, \alpha\}$, $i = 1, \ldots, 4$.

To make the emulation challenging, we train the emulator using ensembles with fewer than 100 members. We construct two training ensemble designs, a grid and a Latin hypercube. To construct the grid we choose three evenly spaced values for each of the four parameters, in the ranges 0.5–4.5 (Q_{10}), 0.5–3.5 cm^2/s (κ), 1–7 °C (S), and 0–2 (α). This gives a $3^4 = 81$ member ensemble. A grid of 3^p members is the coarsest grid one would typically use, corresponding to picking a low, medium, and high value for each parameter. We generate a maximin Latin hypercube design of the same size over the same parameter ranges. These designs are shown in Figure 2.

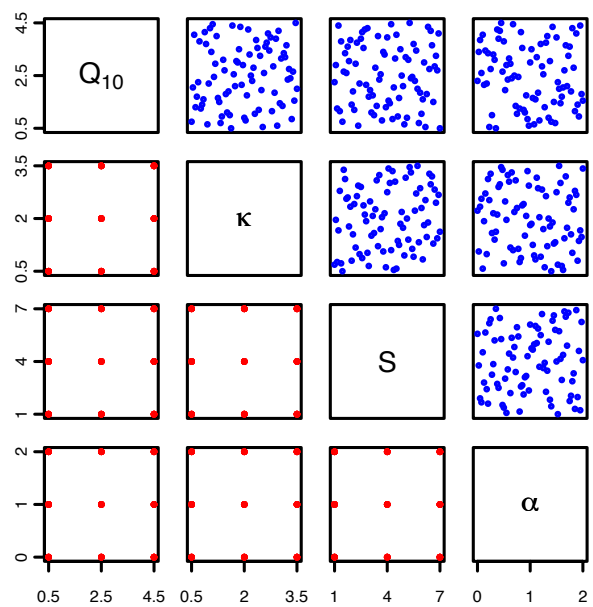

Figure 2: Designs of $p = 4$ dimensional, $N = 81$ member ensembles. Depicted are scatterplots of the designs projected onto two-parameter subspaces. *Lower left:* Grid design. *Upper right:* Latin hypercube design. Note each point in grid scatterplots represents $3^2 = 9$ different ensemble points: due to a grid design's collapsing property, unique points in 4-dimensional parameter space can project onto identical points in a 2-dimensional subspace.

To evaluate the predictive skill of the emulator, we construct a separate validation ensemble of 40 members, distributed in a Latin hypercube over the interior of the parameter ranges. We choose a Latin hypercube design over a grid so that a wider variety of inter-point distances can be represented in the validation ensemble. The model output from this ensemble is withheld when training the emulator with either of the two training designs. The emulator skill is judged by its ability

[2] Urban, N.M., Keller, K., 2010. Probabilistic hindcasts and projections of the coupled climate, carbon cycle, and Atlantic meridional overturning circulation systems: a Bayesian fusion of century-scale observations with a simple model. Tellus A, in revision.

to predict the known output at these unseen validation points.

3.3. Statistical emulator

The purpose of the emulator is to interpolate the Earth system model output between ensemble design points. Although the model output is a deterministic function of the input parameters, it is unknown except at the points in the ensemble design. Taking the Bayesian approach, we represent the model as an unknown (random) function $\eta(.)$ and represent the uncertainty surrounding it by means of a stochastic prior process.

The model has multiple time series outputs, so we use a multivariate Gaussian process prior, based on that used by Conti and O'Hagan (2007). We adapt their emulator structure to include time as an additional input. That is, instead of treating each of the three output time series as a vector function of the input parameters, we treat the three model outputs each as a scalar function of both the input parameters and time. For a given time t and configuration of input parameters p we form an augmented input vector $x = (p, t)$, and the corresponding three scalar outputs are $\eta(x) \in \mathbb{R}^3$.

The Gaussian process prior describes our beliefs about the model output, i.e. the function $\eta(.)$, before having seen the ensemble of training data. The prior distribution for $\eta(.)$, conditional on some hyperparameters $\{B, \Sigma, \phi\}$, is the three-variate Gaussian process

$$\eta(.)|B, \Sigma, \phi \sim GP_3[m(.), \Sigma c(., .)] \,. \tag{1}$$

The GP prior has two components: the mean function $m(.)$ and the 3×3 matrix covariance function $\Sigma c(., .)$. The mean function is $m(x) = B^T h(x)$, a linear combination of r regressors on the inputs ($h(x)$), with a matrix of unknown regression coefficients B. The purpose of including a non-constant mean function is to provide a surface which represents global trend of the model output across input space. This helps the emulator to interpolate in regions of parameter space which are sparsely sampled by the training ensemble. We assume a linear trend surface with parameter-time interaction terms, so that $h(x) = h(p, t)$ is a linear combination of the $r = 10$ regressors 1, $\{p_i\}$, t, and $\{p_i t\}$. We fit independent regressions for each output type, so the regression coefficients B form a 10×3 matrix.

The prior covariance function is the product of an output covariance matrix Σ and a parameter-time correlation function $c(., .)$ with correlation lengths set by roughness hyperparameters ϕ. For the correlation function we use a squared-exponential ("Gaussian") function:

$$c(x, x') = \prod_{i=1}^{5} \exp[-\phi_i (x_i - x'_i)^2] \,. \tag{2}$$

The squared-exponential correlation function is mathematically convenient and Kennedy and O'Hagan (2001) found that, in practice, emulation is robust to the form of the correlation function when the model output behaves smoothly across parameter space. (This covariance assumption may not be suitable for the emulation of chaotic climate models, which are not considered here.) The 3×3 output covariance matrix $\hat{\Sigma}$ specifies the output variances and the between-output covariances. The roughness hyperparameters $\{\phi_1, ..., \phi_4\}$ determine the correlation length scale of the prior process in each direction in parameter space (with characteristic length scale $1/\sqrt{\phi}$). The correlation length scale over time is determined by ϕ_5. Correlation length scales specify the rate at which our uncertainty about the model output increases as we move away from a known design point. As such, they describe how 'smooth' we expect the process to be. Smaller roughnesses correspond to longer correlation lengths and smoother processes.

The final step in prior model specification is to choose prior distributions for the hyperparameters. For B and Σ we assume there is little prior knowledge and use the conventional 'non-informative' priors given in Conti and O'Hagan (2007). For the correlation length scales we elicit plausible ranges from the model author, and use a uniform prior distribution on the $\{\phi_i\}$ over those ranges.

Given the above modeling choices, the prior is updated ("trained") using the ensemble output data D, a $nq \times 3$ matrix. For an ensemble with n design points, each column of D represents one of the three types of model output, stacked into a block vector containing n time series each q years in length. The emulator itself is the resulting GP posterior distribution, a multivariate Student-t process with $n - r$ degrees of freedom which is conditioned to exactly interpolate the ensemble data:

$$\eta(.)|D, \phi \sim T_3[m^*(.), \hat{\Sigma} c^*(., .); n - r] \,, \tag{3}$$

where

$$m^*(x) = \hat{B}^T h(x) + (D - H\hat{B})^T A^{-1} t(x) \,, \tag{4}$$

$$c^*(x, x') = c(x, x') - t^T(x) A^{-1} t(x') \tag{5}$$

$$+[h(x) - H^T A^{-1} t(x)]^T (H^T A^{-1} H)^{-1} [h(x') - H^T A^{-1} t(x')] \,,$$

$$\hat{B} = (H^T A^{-1} H)^{-1} H^T A^{-1} D \,, \tag{6}$$

$$\hat{\Sigma} = (n - r - 4)^{-1} (D - H\hat{B})^T A^{-1} (D - H\hat{B}) \,, \tag{7}$$

and where $A = \{c(x_i, x_j)\}$ is the $nq \times nq$ matrix of correlations between design points, $t^T(x) = [c(x, x_1), ..., c(x, x_n)]$ is the $nq \times 1$ vector of correlations between the prediction point and all the design points, and $H^T = [h(x_1), ..., h(x_n)]$ is the $r \times nq$ matrix of regressors in the prior mean function. The Bayesian updating provides generalized least squares (GLS) posterior estimates $\hat{B}$ and $\hat{\Sigma}$ of the response surface regression coefficients and residual output covariance.

The posterior mean, $m^*(x)$, provides point predictions of the model output at any specified input x. The posterior mean is identical to the ensemble output D at points which are in the training ensemble, and interpolates D at other points not in the ensemble. The posterior covariance, $\hat{\Sigma}c^*(.,.)$, provides estimates of the uncertainty in the predictions. The posterior variance vanishes at every training point in D, because there is no uncertainty about the known output in the model ensemble. It is important to note that the posterior covariance represents the emulator's uncertainty about the model output at a new point not contained in the emulator's training ensemble, and does not imply the climate model itself has stochastic or chaotic uncertainty.

As noted above, the hyperparameters B and Σ are inferred analytically by Bayesian updating. On the other hand, the likelihood function for $\{\phi_i\}$ is highly intractable and analytic Bayesian inference is not possible. Instead we calculate point estimates of the $\{\phi_i\}$ and treat them as fixed (known) in the prior to posterior analysis. In kriging one often estimates correlation length scales or range parameters by variogram analysis. In situations with smooth model output and sparse data, as is the case here, we prefer to estimate each $\{\phi_i\}$ by its posterior mode. Since we use a uniform prior distribution, posterior mode estimation is equivalent to the maximum likelihood estimation with the parameters constrained to be within the elicited plausible ranges.

4. Results

4.1. Model output

Any test of an emulator should begin with an inspection of the ensemble output on which the emulator is to be trained. An examination of the output clarifies the behavior of the model, verifies whether the output conforms to the assumptions of the emulator, and exposes problems such as parameter combinations which produce numerical instabilities in the model. Figure 3 shows the time series model output for all runs in the grid and Latin hypercube ensembles.

Several features of the model output are of interest. First, the model output appears highly predictable. Each of the three outputs increases almost monotonically with time for all of the ensemble members. The time series for different ensemble members have similar functional forms. For example, the temperature for runs with high warming is more or less a rescaling of the temperature for runs with low warming. This implies that any emulator is likely to be able to predict the model output well regardless of the design, as we confirm in Section 3.3.

A second, related point is that while the training ensembles may be sparse in parameter space, the ensemble output is relatively dense in response space. The training runs in both designs span a continuous range of responses from small to large, with few obvious gaps where no output exists in the ensemble. This also suggests that both emulators should work well.

A third feature of the model output is that it is nonstationary in time. The ensemble output shows very little variation at early times, because all the runs were started with the same initial conditions, and large variation at late times. This could potentially prove troublesome for a statistical emulator which assumes the output is stationary in time (given the mean function), as ours does. However, we show in Sections 4.2 and 4.3 that the emulator's predictive skill is nevertheless quite good. This due to the smoothness of the model output noted above, and because violations of stationarity are expected to affect the precision of the emulator prediction error bars more than the accuracy of the prediction itself.

Fourth, the grid ensemble shows a somewhat larger variation in model output than does the Latin hypercube design. This is because the grid has more points on the boundaries of parameter space, with more extreme parameter values producing stronger model responses. An emulator trained on the Latin hypercube may have difficulty predicting the largest responses from points near the boundary, as it will have to extrapolate rather than interpolate. Our results in the following sections suggest this is not the case: the LHD emulator can predict boundary responses rather well and its overall prediction quality is high.

4.2. Emulator validation

To test the skill of the emulator, we train it on the grid and Latin hypercube ensembles and compare its predictions to the known model output in an independent validation ensemble. The validation ensemble and emulator construction are described in Sections 3.2 and Sections 3.3, respectively.

We use a root mean squared error (RMSE) skill metric to assess model skill, with the average taken over

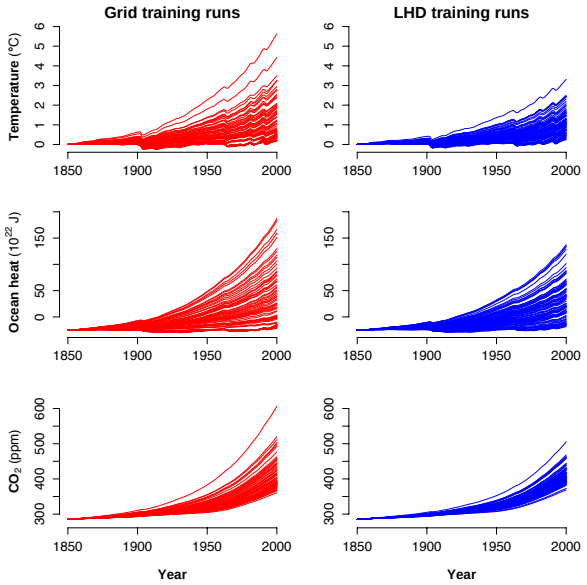

Figure 3: Model output for all 81 members in grid (left, red) and Latin hypercube (right, blue) training ensembles, for global temperature anomaly (top), ocean heat anomaly (middle), and atmospheric CO_2 concentration (bottom).

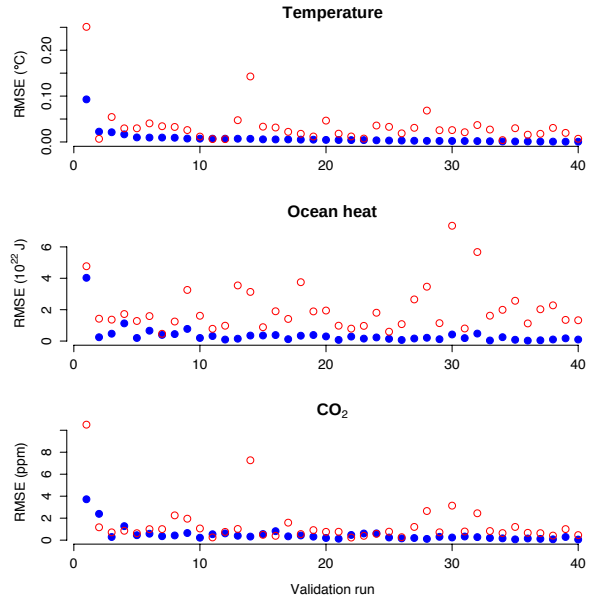

Figure 4: Validation plots of emulator prediction skill for grid (red circles) and Latin hypercube (blue dots) designs, for global temperature anomaly (top), ocean heat anomaly (middle), and atmospheric CO_2 concentration (bottom). Prediction skill is measured by root mean squared error (RMSE). The 40 validation runs are sorted in order of decreasing RMSE for temperature.

time. In Figure 4 we compare the predictive skill of the two emulators on the same 40-member validation ensemble. The Latin hypercube emulator is clearly a better predictor than the grid emulator for all three output types: its RMS error is several times smaller for almost every prediction in the validation set.

Examples of time series predictions for the two emulators are given in Figure 5. For each emulator and output type, the validation runs are sorted by predictive skill. The worst predicted time series is shown along with the prediction at the 25th percentile of skill and the median quality prediction. (The 75th percentile and best predictions are not shown because the emulator quality is nearly perfect in virtually all cases.) The results confirm the general superiority of the Latin hypercube emulator over the grid emulator. The LHD emulator prediction is visually indistinguishable from the true model output in almost all cases. However, both emulators appear to perform well in most cases. The emulators show the best skill (smallest error) in predicting CO_2. This probably reflects the fact that the model output is smoothest in CO_2 (see Figure 3).

A statistical emulator not only predicts the computer model output, but also gives an estimate of its prediction error. The emulator quality should be judged not only by its predictive accuracy (bias), but also by the precision of its error estimates (variance). Figure 5 shows the 95% predictive credible intervals of the two emulators as shaded bands. A credible interval is the Bayesian version of a frequentist confidence interval. In Figure 5, the black curves should lie within the shaded bands for approximately 95% of the years.

The Latin hypercube-trained emulator has much smaller predictive intervals than the grid-trained emulator. In most cases the intervals are smaller than the line width in the plot. This reflects the high precision of the emulator. However, for the worst-fitting emulator predictions, these small intervals are overconfident: the true model output does not lie within them.

The grid emulator has larger predictive intervals than the LHD emulator, reflecting its relative imprecision. Overconfidence is also visible in the grid emulator. In many of the plots the true model output lies outside, or only barely inside, the grid emulator confidence intervals. In several cases the grid intervals have the opposite problem of underconfidence: they are much wider than necessary to enclose 95% of the data (e.g., in some of the CO_2 predictions), given the accuracy of the emulator. (Some of this may be due to the emulator's stationary covariance assumption. By assuming that the prior uncertainty is equal at all points in time, the emulator can overestimate the small uncertainty in model output at early times.) In general, the Latin hypercube emula-

tor appears to produce better-calibrated uncertainty estimates than does the grid emulator.

4.3. Single-parameter scans

As discussed in Section 2, one commonly touted virtue of grid designs is the ease with which they facilitate single-parameter sensitivity studies. A linear scan of model output along a parameter dimension, altering one parameter's value while holding the others fixed, can be achieved by moving from one grid point to the next. This cannot happen in a Latin hypercube design, since there is only one design point for each parameter value, and therefore it is impossible to hold a parameter fixed while moving from one point to another.

Despite this supposed weakness of Latin hypercube designs, it is not clear whether a grid design is the better choice for parameter scans. In a 3^4 grid, for instance, a parameter scan intersects at most three grid points. A grid-trained emulator which predicts the model response along a scan line is limited by extremely sparse data availability. While no more than one point in a Latin hypercube design can lie along any given scan line, there may be many design points which are close to the scan line. It is thereby possible that an LHD emulator can outperform a grid emulator for single-parameter sensitivity studies.

To test this hypothesis, we conduct a linear scan of each of the four parameters around the central point in our grid design, $\{Q_{10}, \kappa, S, \alpha\} = \{2.5, 2, 4, 1\}$. In Figure 6 we plot the emulator-predicted model response in the year 2000 for each of the three output types, as a function of the parameter varied in the scan. The predictions are evaluated against the true model output at 40 validation points for each scan.

By definition, the grid emulator prediction is perfect at the three points in each scan which coincide with grid training points (the black dots in Figure 6). However, for intermediate points, the predictions of the LHD emulator are not only superior to those of the grid emulator, but also in almost every case are visually indistinguishable from the true model output.

5. Discussion

In this case study, a Latin hypercube ensemble design produces a better statistical emulator of an Earth system model than does a regular grid design. The LH-trained emulator is superior to the grid emulator in terms of root mean squared prediction error, the precision of its predictive uncertainty intervals, and its ability to conduct single-parameter model sensitivity studies. The latter is an application for which grid designs are often favored. Our results suggest this preference for grids should be reconsidered.

This study comes with a number of caveats. Although the grid-trained emulator performs worse than the LHD emulator, in practical terms there is not a large difference between their predictions. We intentionally use a very coarse grid design, and the difference between grid and Latin hypercube emulators is likely to disappear with a finer grid, if the number of varied parameters is constant. In contrast, the superiority of a Latin hypercube emulator is likely to increase if the number of ensemble parameters is larger.

However, the task of emulation is particularly easy for the simple model used here because of the smoothness of its outputs over parameter space. Such smoothness is likely to be a general feature of many climate time series which show a gradually increasing response to anthropogenic forcing. The emulator could also potentially be improved by using the forcing input, rather than time, as a regressor in the prior mean function. When emulation is easy, even a coarse grid design will do well. Not all climatic time series are as easily emulated as those studied here, particularly when the system is very nonlinear or has large noise. Emulation will also be more difficult for multidimensional spatial or space-time fields. In more difficult emulation tasks, a Latin hypercube ensemble design may show a greater practical superiority than it does in this study.

6. Conclusions

The analysis of computer model sensitivity and uncertainty requires that the model can be evaluated at many different combinations of parameter settings. Complex computer models are too computationally intensive to permit a full exploration of model parameter space. A statistical emulator is a tool to predict computer model output at unseen parameter settings by interpolating the output from a training ensemble of limited size. Different choices for the design of the training ensemble may improve or degrade the quality of the resulting emulator.

In this case study, a Latin hypercube design shows a small but clear advantage over a regular grid design for training a statistical emulator of an Earth system model. This advantage persists when conducting single parameter model sensitivity studies, an application for which grid designs have commonly been preferred.

It should be noted that Latin hypercubes may not always prove a superior design choice. If the ensemble output is to be used for purposes other than model

emulation, a grid design can retain the advantages discussed in Section 2. Simple sensitivity studies and model-data comparisons can be carried out more easily with a regular grid. It may be that no more complex analysis is needed. If it is not known in advance whether an emulator will be constructed for the planned ensemble, a simpler grid design could be warranted. Also, the Cartesian product structure of a grid can speed the matrix inversions required in statistical emulation (Kennedy and O'Hagan, 2001), so computational gains are possible with a grid design although emulator accuracy may suffer. But improved speed may be moot if the parameter space is of high enough dimension that a grid design cannot adequately cover it.

However, based on the case study in this paper, we recommend that Latin hypercube designs generally should be preferred over grid designs when constructing a statistical emulator from an ensemble of model output. As far as emulator skill is concerned, grid designs have no obvious advantage over Latin hypercube designs. An emulator can be trained with equal ease on a design of any form, so the simplicity of a grid is irrelevant. Single parameter scan predictions are equally simple for an emulator with any kind of design, as are marginalization integrals via Markov chain Monte Carlo. Latin hypercubes are not difficult to generate, have good theoretical properties, and their predictive superiority is borne out in practice (Section 4).

We conclude by noting that although we have shown that Latin hypercubes are good candidates for the design of an emulator training ensemble, there are numerous other space filling designs which have not been discussed here. For example, designs based on Sobol' sequences are sometimes used (e.g. Santner et al., 2003) as alternatives to Latin hypercubes. A Sobol' sequence is a quasirandom sequence of numbers that can be shown to fill an interval uniformly for large samples (Sobol', 1976). The advantage of using a Sobol' sequence design is that it can built sequentially. If it is later decided that a larger ensemble is desired, a longer Sobol' sequence can be constructed by extending an existing sequence. LH designs, on the other hand, must be recomputed from scratch if a larger design is required, and will not share any points in common with a smaller LH design. The disadvantage of Sobol'-sequence based designs is that their small sample properties are not clear, and unlike LH and grid designs, they do not have the guarantee of filling each parameter dimension uniformly.

7. Acknowledgments

This study was inspired by work undertaken at the 2008 Probability and Uncertainty in Climate Modeling (PUCM) Research Playground workshop at Durham University. We would like to thank Klaus Keller, Jeremy Oakley, Josh Dorin, and the two anonymous reviewers for useful discussion and suggestions. N.M.U. would like to acknowledge funding support from the National Science Foundation (HSD 072 9413). Any opinions expressed here, as well as any errors, are the responsibility of the authors alone.

References

Banerjee, S., Carlin, B.P., Gelfand, A.E., 2004. Hierarchical Modeling and Analysis for Spatial Data, Chapman & Hall/CRC, Boca Raton, 512 pp.

Challenor, P.G., Hankin, R.K.S., Marsh, R., 2006. Towards the probability of rapid climate change, In: Schellnhuber, H. J., Cramer, W., Nakicenovic, N., Wigley, T., Yohe, G. (Eds.) Avoiding Dangerous Climate Change, Cambridge University Press, Cambridge, pp. 55-63.

Claussen, M., Mysak, L.A., Weaver, A., Crucifix, M., Fichefet, T., Loutre, M.-F., Weber, S., Alcamo, J., Alexeev, V., Berger, A., Calov, R., Ganopolski, A., Goosse, H., Lohmann, G., Lunkeit, F., Mokhov, I., Petoukhov, V., Stone, P., Wang, Z., 2002. Earth system models of intermediate complexity: Closing the gap in the spectrum of climate system models. Climate Dynamics 18, 579–586.

Conti, S., O'Hagan, A., 2007. Bayesian emulation of complex multi-output and dynamic computer models. Research Report No. 569/07, Department of Probability and Statistics, University of Sheffield, Sheffield, 11 pp.

Cressie, N., 1993. Statistics for Spatial Data, Wiley, New York, 928 pp.

Davidson, E.A., Janssens, I.A., 2006. Temperature sensitivity of soil carbon decomposition and feedbacks to climate change. Nature 440, 165–173.

Hansen, J., Russell, G., Lacis, A., Fung, I., Rind, D., Stone, P., 1985. Climate response times: Dependence on climate sensitivity and ocean mixing. Science 229, 857–859.

Hooss, G., Voss, R., Hasselmann, K., Maier-Reimer, E., Joos, F., 2001. A nonlinear impulse response model of the coupled carbon cycle-climate system (NICCS). Climate Dynamics 18, 189–202.

Husslage, B., Rennen, G., van Dam, E.R., den Hertog, D., 2008. Space-filling Latin hypercube designs for computer experiments. CentER Discussion Paper No. 2008-104, Tilburg University, Center for Economic Research, Tilburg, Netherlands, 14 pp.

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

Knutti, R., Hegerl, G., 2008. The equilibrium sensitivity of the Earth's temperature to radiation changes. Nature Geoscience 1, 735–743.

Knutti, R., Stocker, T.F., Joos, F., Plattner, G.-K., 2003. Probabilistic climate change projections using neural networks. Climate Dynamics 21, 257–272.

Kriegler, E., 2005. Imprecise probability analysis for integrated assessment of climate change. Unpublished Ph.D. Dissertation, University of Potsdam, Potsdam, 256 pp.

Lohmann, U., Feichter, J., 2005. Global indirect aerosol effects: A review. Atmospheric Chemistry and Physics 5, 715–737.

Matthews, H.D., Keith, D.W., 2007. Carbon-cycle feedbacks increase the likelihood of a warmer future. Geophysical Research Letters 34, L09702.

McKay, M.D., Beckman, R.J., Conover, W.J., 1979. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics 21, 239–245.

Morris, M.D., Mitchell, T.J., 1995. Exploratory designs for computational experiments. Journal of Statistical Planning and Inference 43, 381–402.

Panchuk, K., Ridgwell, A., Kump, L.R., 2008. Sedimentary response to Paleocene-Eocene thermal maximum carbon release: A model-data comparison. Geology 36, 315–318.

Oakley, J., O'Hagan, A., 2004. Probabilistic sensitivity analysis of complex models: A Bayesian approach. Journal of the Royal Statistical Society Series B 66, 751–769.

Ridgwell, A., Hargreaves, J.C., Edwards, N.R., Annan, J.D., Lenton, T.M., Marsh, R., Yool, A., Watson, A., 2007. Marine geochemical data assimilation in an efficient earth system model of global biogeochemical cycling, Biogeosciences 4, 87–104.

Sansó, B., Forest, C., Zantedeschi, D., 2008. Inferring climate system properties using a climate model. Bayesian Analysis 3, 1–38.

Santner, T.J., Williams, B.J., Notz, W.I., 2003. The Design and Analysis of Computer Experiments, Springer, New York, 283 pp.

Schmittner, A., Urban, N.M., Keller, K., Matthews, D., 2009. Using tracer observations to reduce the uncertainty of ocean diapycnal mixing and climatecarbon cycle projections. Global Biogeochemical Cycles 23, GB4009.

Schneider von Deimling, T., Held, H., Ganopolski, A., Rahmstorf, S., 2006. Climate sensitivity estimated from ensemble simulations of glacial climate. Climate Dynamics 27, 149–163.

Sobol', I.M., 1976. Uniformly distributed sequences with an additional uniform property. USSR Computational Mathematics and Mathematical Physics 16, 236–242.

Sokolov, A.P., Stone, P.H., Forest, C.E., Prinn, R., Sarofim, M. C., Webster, M., Paltsev, S., Schlosser, C.A., Kicklighter, D., Dutkiewicz, S., Reilly, J., Wang, C., Felzer, B., Melillo, J. M., Jacoby, H.D. Probabilistic forecast for 21st century climate based on uncertainties in emissions (without policy) and climate parameters, 2009. Journal of Climate 22, 5175–5204.

Tomassini, L., Reichert, P., Knutti, R., Stocker, T.F., Borsuk, M.E., 2007. Bayesian uncertainty analysis of climate system properties using Markov chain Monte Carlo methods. Journal of Climate 20, 1239–1254.

Wackernagel, H., 2003. Multivariate Geostatistics, third ed., Springer, New York, 387 pp.

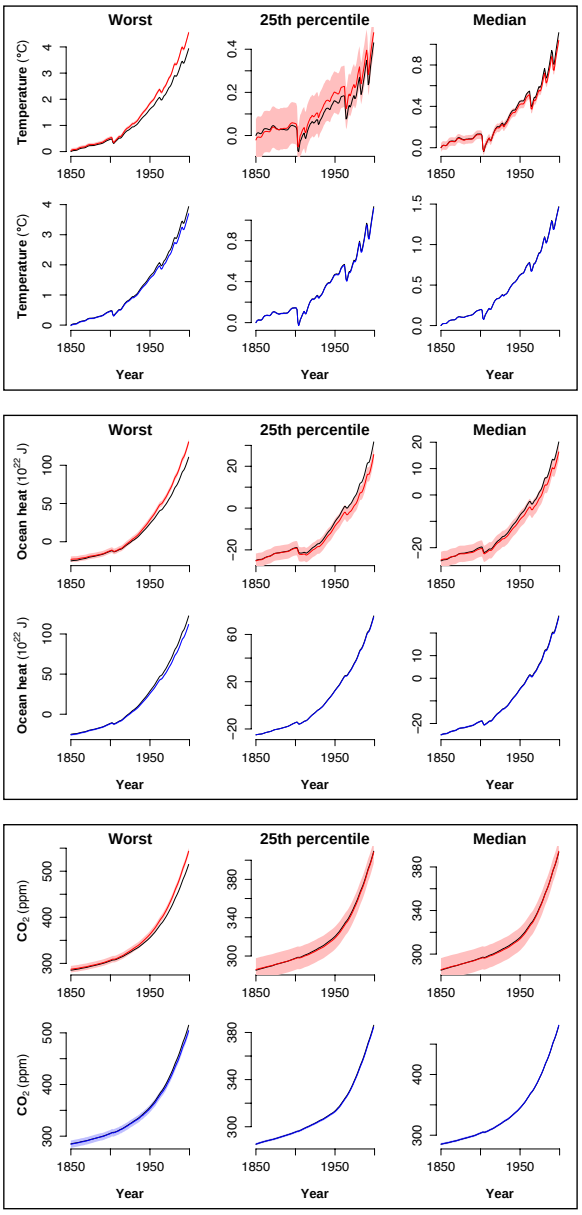

Figure 5: Validation plots for temperature anomaly (top), ocean heat anomaly (middle), and CO_2 concentration (bottom). For each output type, upper row shows grid emulator prediction (red), and lower row shows Latin hypercube emulator prediction (blue). Black curves are true model output at validation point. Shaded areas are 95% predictive credible intervals quantifying statistical uncertainty in emulator prediction. Depicted are validation points with worst, 25 percentile, and 50 percentile prediction quality (as measured by RMSE) for each emulator. Note vertical scale of plots differs between panels.

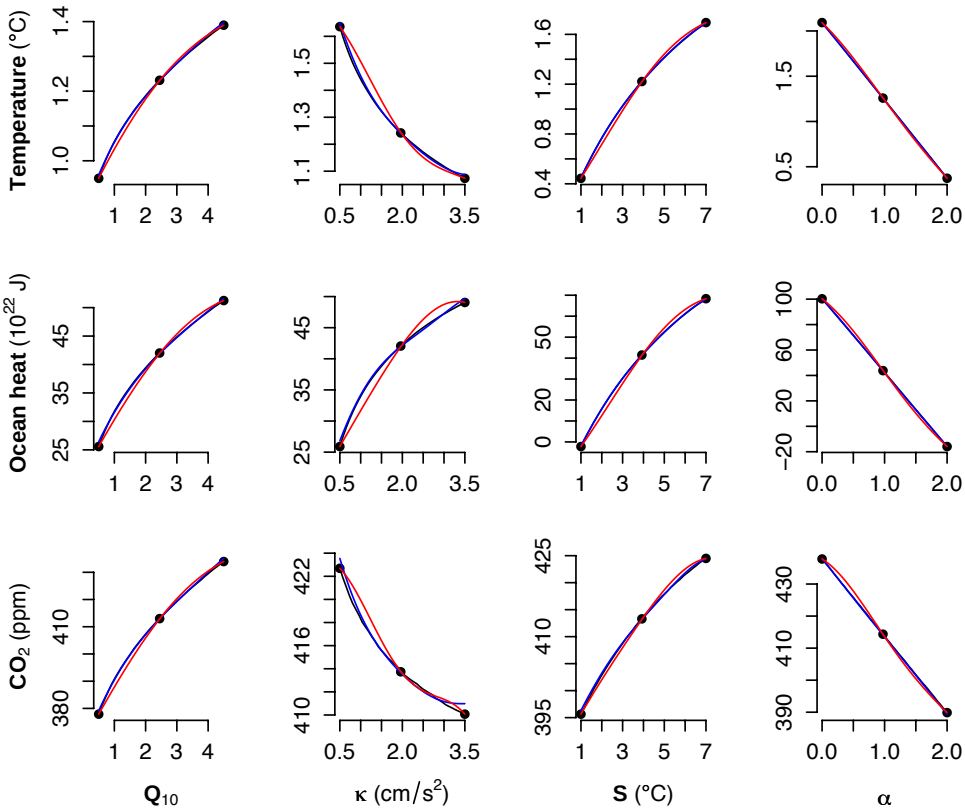

Figure 6: Single parameter sensitivity scans for grid (red) and Latin hypercube (blue) designs, compared to true validation model output (black). Each plot shows variation in model output in year 2000 as a single model parameter is varied, with all others held fixed. The black dots are model output at three grid points intersecting each scan. Depicted is output for temperature (top), ocean heat (middle), and CO_2 (bottom), for each of four parameters varied about a central point $\{Q_{10}, \kappa, S, \alpha\} = \{2.5, 2, 4, 1\}$. Note vertical scale of plots differs between panels.