An extreme non-linear example of the use of emulators with simulators using the Stommel Model

Robin Tokmakian[a,*], Peter Challenor[b], Yiannis Andrianakis[b]

[a]*Department of Oceanography, Naval Postgraduate School, Monterey, California.*
[b]*National Oceanography Centre, Southampton, UK.*

Abstract

Using the simple Stommel 1961 model, we illustrate how emulators can reasonably represent the full sampling space of a non-linear, bimodal system. This extreme example shows how emulators can be useful to explore the parameter space (initial conditions, process parameters, and boundary conditions) of a complex computer model such as ocean and climate general circulation models, even when the model outcomes contain steps in their response. In addition, we show that the emulator can help to elucidate interactions between parameters as well as help in the calibration of parameter values for specific outcomes and in the determination of uncertainty in the prediction of outcomes.

Keywords: emulator, simulator, uncertainty, GCM

1. Introduction

Statistical emulators have been used to understand models and their parameter space in a wide set of applications. An established community uses

*Corresponding author: R. Tokmakian
Email address: `rtt@nps.edu` (Robin Tokmakian)

the advanced statistical methods of designed experiments combined with emulators to study and analyze computer simulations of complex phenomena. Applications include computational physics for nuclear weapons, models use in support of exploring oil fields, issues in aircraft engine design, weather prediction and climate science (Higdon et al., 2004; Williams et al., 2006; Sansó et al., 2008; Sansó & Forest, 2009). All of them contain similar requirements, that is, the necessity to calibrate input parameters or a need to estimate the uncertainty of a prediction (O'Hagan, 2006). An example of an application of the method in a complex simulation more akin to ocean/atmosphere general circulation models is described in the recent cosmology paper of Heitmann et al. (2006). In that paper, uncertainties and sensitivities of the underlying model's parameter space were explored through the use of an emulator and calibrated with respect to recent observations of the large-scale structural statistics of the cosmos. Emulators are extremely adaptable ways of analyzing the structure of a non-linear simulation. However, they do make an assumption that the relationship between the model inputs and outputs are fairly smooth.

A question is often asked: can emulators of strongly non-linear models be generated successfully, especially models that result in bimodal outcomes of a specific system? To address this, we examine the use of emulators as applied to a simple dynamical simulator, the classical Stommel box model (Stommel, 1961). (Note: the word simulator is used rather than model, to distinguish the dynamical model or simulator from the statistical model.) This simulator results in two possible stable states at equilibrium, depending upon the initial conditions of the system. The output is the result of complex non-linear interaction between two variables, temperature and salinity and results in two states with differing density flux at the end. The paper describes, first, how the Stommel simulator is set up and used, followed by a section describing what an emulator is and how it is implemented. Finally, we show the results of applying the emulator methodology to explore the output space of the Stommel simulator dependent upon its initial conditions and suggest that the emulator techniques are a useful methodology to explore process parameters, initial conditions, and boundary conditions in complex general circulation models of the ocean, atmosphere, and climate.

2. Experiment Setup

The Stommel box model or simulator consists of 2 boxes: an equatorial box and a polar box (Figure 1). Each box has a given temperature and salinity. The density difference between the boxes determines the flux (d) and is defined as

$$d = (R\Delta T - \Delta S)/\lambda, \tag{1}$$

where ΔT is a normalized temperature difference between the two boxes, a value between 0 and 1 and ΔS is a normalized salinity difference between 0 and 1. R is a measure of the effect of salinity and temperature on the density. λ is a non-dimensional defined inverse flushing rate. For our test, we set $R = 2$, $\lambda = 0.2$. Because we want to examine only how an emulator treats a bimodal problem, we keep R and λ constant and only vary the initial values of ΔT and ΔS. If we wanted to examine the full range of possible solutions, we would build an emulator to include how changes in R and λ influence the solution.

To evaluate the ability of the emulator to recover the equilibrium flushing rate, we first run the simulator across a large subset of the possible initial non-dimensional temperature and salinity difference values from 0 to 1. The resulting equilibrium density difference field of a uniform sampling of 100 points for each ΔT and ΔS is shown in Figure 2a. It is a spatial map of the d as a function of ΔT and ΔS. Generally, the flux is either close to -1.07 or close to 0.2. In the classic study, there is an unstable region between the two stable regimes with a value at around -0.3. Figure 2b shows the time evolution of temperature differences and salinity differences for several initial values illustrating the convergence of the ΔT and ΔS towards the two distinct densities. Figure 2c shows the corresponding plot for density differences, while Figure 2d illustrates the evolution of the two inputs together against the distribution of density (contour lines). The task is to create an emulator which approximates this distribution, using a very limited set of simulations outcomes.

3. Emulator details

We denote a model or simulator output as $Y = F(\mathbf{x})$, where $\mathbf{x}$ is a vector of tunable inputs and F is the mathematical function being simulated, linear or non-linear. By making a few runs of the simulator with a carefully designed set of parameter input values, a small set of known outputs Y for a given vector $\mathbf{x}$ is produced. This set of outputs Y (sometimes referred to as "training data") has zero uncertainty. The outputs and parameter settings are then used to create an emulator, $f(\mathbf{x})$. The emulator reflects the true value of Y at points x. At other points, we expect the distribution of $f(\mathbf{x})$

should give a mean value for $F(\mathbf{x})$ and associated uncertainty that represents a plausible value of output Y given any vector $\mathbf{x}$. The probability distribution should be a realistic view of the uncertainty in the simulator.

A Gaussian process (GP) is used for $f(\mathbf{x})$ under the assumption that the uncertainty in the emulator can be described with such a process. A GP can be understood as a generalization of a Gaussian distribution over an infinite vector space. Just as a Gaussian distribution has a mean and variance, a GP has a mean *function* and a *covariance function*. It does not mean that either the distributions of the input parameters or the final metrics are Gaussian. Normally the function f is smooth and continuous over its parameter space, although anything known about the response can be incorporated into the emulator by how the mean function is defined, including strong nonlinearities and discontinuities. Under such a model, the uncertainty regarding a response Y at some vector location $\mathbf{X}$ is easily obtainable. In the case we are looking at in this paper, the output Y is not continuous. $f(\mathbf{x})$ is modeled first by a mean function given by:

$$m_0(\mathbf{x}) = h(\mathbf{x})^T\beta, \tag{2}$$

where $h(\mathbf{x})^T$ is a vector of q regression functions and β is a vector of q parameters. In the case presented in this paper, the mean function is represented by a simple linear function such that

$$h(\mathbf{x})^T = \begin{pmatrix} 1 & \mathbf{x} \end{pmatrix}. \tag{3}$$

More complex functions can be considered. The Gaussian process then models the residual, so that the joint distribution of two points, (x_1, x_2), is also normal with the covariance given by

$$v_o(x_1, x_2) = \sigma^2 \chi(x_1, x_2). \tag{4}$$

$\chi(x_1, x_2)$is, in the example here, equal to $e^{(-(x_1-x_2)^T B(x_1-x_2))}$, a Gaussian correlation function that assumes stationarity and gives a smooth emulator. B is a matrix of smoothing parameters normally set to be diagonal. The b_{ii} are smoothing parameters and $1/\sqrt{b_{ii}}$ are the correlation length scales. Since these methods are Bayesian, they can incorporate expert knowledge (prior knowledge) to define prior distributions of β, σ^2, and B. To form the posterior probability distribution, these prior distributions are combined with the results of the simulation runs (Y) in the realization of the emulator. Thus, we use the regression functions associated with the vector β to determine an outline of the function f, and the Gaussian process model to emulate the systematic variation of the response around our values of Y. Parameters may be constrained by a-priori knowledge of the parameter of interest. For our test problem, we use a linear prior and a Gaussian covariance function with non-informative priors for μ and σ^2. B is estimated by maximizing the marginal likelihood; i.e. we estimate the b_{ii} by determining their most probable values, given the data. For further details on the GP emulators see Oakley & O'Hagan (2004) or the Managing Uncertainty in Complex Models (MUCM) website at mucm.ac.uk. The advantage of using an emulator is that it is very quick to compute so can be used instead of the expensive full simulator for inference. This is not the case for our example where our simulator is itself fast to run, but our example allows us to easily compare the emulator to the full output of the simulator.

4. Experiment Design

Before showing results, an explanation of the design of the experiment is described. We set up the design in the following manner. First, we define a sampling strategy of the initial conditions ΔT or ΔS for n initial simulations. The resulting emulator is created using the $n-1$ outcomes. one simulator outcome is withheld as a test point. Since we know the outcomes of the deterministic Stommel model, we can examine the result of our emulator in terms of the fully sampled initial condition space. In the areas where we believe the emulator solution to be far from the true solution, we can re-sample our initial conditions constrained to the area that has a large uncertainty. Even under the conditions where the full space is unknown, we can still run sequential and further constrain the initial condition region.

There are several well understood sampling strategies we could follow. Experimental designs such as the Latin hypercubes (McKay et al., 1979) and Sobol sequences (Sobol, 1967) minimize the number of runs we need to do to build an emulator. In the context of statistical sampling, a square grid containing sample positions is a Latin square if and only if there is only one sample in each row and each column. A Latin hypercube is the generalization of this concept to an arbitrary number of dimensions, in that each sample is the only one in each axis-aligned hyper-plane. Sobol sequences, on the other hand, discretize space using a base 2 system with some reordering of the resulting sequence. It is a pseudo-random process. In this paper, we use Sobol sequences for our design. We conduct a two stage experiment, first, an initial design for a set of points before using an additional sets of simulations to refine the emulator.

5. Emulator evaluation

Once an emulator has been built, it is necessary to evaluate its quality. A number of methods have been proposed including some that consider how far the solution is from independent validation points (Bastos & O'Hagan, 2009). The first step is create a set of one or more validation points that are not included in the creation of the emulator such that Y represents the simulation outcomes at the validation locations $\mathbf{X}$. Next, use the emulator to create a set of predicted outcomes $f(\mathbf{X})$ with its associated variance $V(f(\mathbf{X}))$. This validation data set then can be used in one of more set of diagnostics. One of the diagnostics is called the Mahalanobis distance:

$$D^{MD}(Y) = [\mathbf{Y} - f(\mathbf{X})]^T V(f(\mathbf{X}))^{-1}[\mathbf{Y} - f(\mathbf{X})]. \tag{5}$$

When $D^{MD}(Y)$ is extreme (i.e. much greater or much smaller than the number of points in the validation set), then the emulator solution should be examined closely to identify regions which need to be improved. This diagnostic assumes that the solution should be smooth. In our test problem, we have a jump in the solution and thus, fails the smoothness requirement. Instead, we can also use a simpler diagnostic. We can estimate the "skill" of the emulator by

$$D_i^I(Y) = [Y_i - f(X_i)]/\sqrt{V(f(X_i))}, \tag{6}$$

By plotting the D^I values against the location of the validation points, $\mathbf{X}$, we can examine the locations in parameter space that are contributing large errors in the emulator solution and decide how to further refine the emulator for this region of space.

6. Results and Discussion

We first create an emulator using $n = 10$ simulations of the Stommel model which vary in the value for the inputs: ΔT and ΔS. A Sobol sequence is used to sample the parameter space for each to determine what values of ΔT and ΔS to use. This allows for the initial condition space to be sampled such that the interactions between the two parameters, ΔT and ΔS will be sufficiently sampled.

Figure 3 shows plots of two distributions of the density difference field (d), given two different draws of 9 samples from the 10 member Stommel simulation set. The white contour line represents the true bimodal separation line between the two solutions of the dynamical model given a fully sampled system (see Figure 2a). The open circles represent the outcomes from the simulator, the Stommel model that are used to create the emulator. The colored dot, with the embedded black open circle is the 10th outcome, which should fall within in the emulator space when the emulator is a reasonable representation of the simulator. If the color matches the background field, then the excluded point fits the emulator. For the Figure 3a, the excluded point, located in the top right portion of the field, is some distance from the emulator estimate as seen by its reddish color with $D^I = 0.46$. This emulator fails a diagnostic for a good emulator - that is, the distribution of $f(x)$ should give a mean value for $F(x)$ that represents a plausible value of output Y. In the second case, Figure 3b, the emulation is more successful because the excluded point (lower right region) falls within the emulator distribution. This emulator gives a reasonable estimate for a validation point ($D^I = 0.13$).

Figure 3a and b also show that a large part of the simulator space is void of any sample points. The overlaid dashed contour lines show the variance at any given point, and thus, the uncertainty in its estimate. From the result of our 9 member ensemble emulation, we can further explore the initial condition space by sampling the region to the right of ΔS between 0.4 and 1. Even if we didn't know the underlying distribution of d, we might believe that with the strong gradient in the initial estimate of the d field, further sampling of the region with the gradient might be useful to further define the emulator. For our example, we resample using a simple scheme of choosing 10 addition points between 0.4 and 1 for the ΔS parameter and leave ΔT to be sampled between 1 and 0 again. Figure 3c is the resulting emulator density difference field using this expanded set of 19 simulator points. It is easily seen that the distribution in the emulation space is much closer to the true spatial distribution of the outcome d. (The validation point has a D^I of close to 0.) The variance of the emulated solution is also reduced in the Figure 3c with the additional simulator points. There is a shift in location of the high (0.22) bimodel region. It is shifted so that it is more contained within the white contour that denotes the true division between the regimes. This is to be expected because of the additional points within that area that are being used to create the emulator from the simulator output locations. This illustrates the use of a sequential design process to explore regions of high uncertainty.

The sampling characteristics on an emulator solution can also be shown by using a Sobol sequence of $n = 40$ rather than 10. Using 39 of the 40 simulator outputs, we created an emulator with its solution shown in Figure 4a. It shows a much more realistic representation of the expected density space over all the possible ΔT and ΔS values. The 40th point, not included in the emulator creation, is located at about 0.18 and 0.81 with the thinner black circle. The white curve is the true curve which delineates the two stable solutions. Figure 4b shows the same figure, but with areas that are not within two standard deviations of mid point (-0.42) of the two solutions. (The dashed lines represent the variance distribution.) Ideally, we would want the emulator solution, represented by the solid black line to lie on the red line, the truth. If not that, then a very successful emulator would have the colored shaded region contain the red curve. As this figure shows, this emulator has produced a region that is overconfident around ΔS = 0.4 and ΔT between 0.35 and 0.55, as well as under confident in the areas with large spread about the black contour. We might also want to further sample the left and bottom right corner regions, because of their particularly high variance.

Figure 4b can also be used to estimate the probability that this system will flip from one stable regime to another. For example, if we had a $\Delta S = 0.9$ and a $\Delta T = 0.35$, we could give an estimate with an associated uncertainty that the system will flip if the ΔT value increases to 0.4. While this simulator and its emulator are straight forward to understand, a system with more parameters and more complexity will add additional complications towards understanding such predictions. However, this type of methodology allows us to explore the space in a systematic manner. The D^I values for a set of 100 validations is shown in Figure 4c, along with the black contour line separating the the two states of the model. It is clear that the points that have the least skill (values greater than +/- 2) are in the region where the jump between the two states occur. We would expect such a result, given the extreme nature of the non-linearity. This can be quantified also using the D^{MD} diagnostic. When using all the 100 validation points, $D^{MD} \approx 10,000$. If we remove from the calculation, those points with the greatest uncertainty, ($D^I > 2$, and marked by a black dot on Figure 4c), then $D^{MD} = 92.8$ for this set of 81 points. This is what we would expect for a reasonable emulation.

Last, we show the sensitivity of the solution (d) to each of the input parameters (ΔT and ΔS) in Figure 5. Figure 5 explicitly illustrates the division between the influence of the initial conditions of the temperature and salinity differences on the outcome. It shows the response of one of the two inputs, given the other input is held constant at 0.5. For ΔT, the important shift is between 0.2 and 0.3, while the salinity shift is between 0.4 and 0.5. Again, while these relationships can be easily seen with the model without the emulator, the plot is shown to illustrate how input parameters relate to one another and how one can determine the importance of one variable over another and the interaction between the multiple variables.

7. Conclusion

This test problem illustrates how emulators can be useful to explore aspects of complex models when the resources are not available to run thousands of simulations. It is an extreme illustration, in that most systems will not have distinct bimodal regimes, but rather more continuous solutions that have less stringent fitting requirements. These emulators, thus, should prove useful to explore the full space of complex simulations (such as atmosphere-ocean general circulation models, AOGCMs) including its parameters, initial conditions, and/or boundary spaces. These AOGCMs, especially in the context of climate projections or seasonal forecasts would benefit from such an exploration of their space through the use of emulators.

Acknowledgments

David Stevens prompted us to justify that emulators can be used in bimodel or bifurcation type problems. We thank the Isaac Newton Institute of Mathematical Sciences at the University of Cambridge for allowing us to spend time at the institute where this research was completed. J. Rougier provided helpful comments for improving the manuscript. This work was also completed with funding under NSF Grant No. 0851065. Jim Price is thanked for making available his Stommel (1961) model matlab code (pubic domain software).

References

Bastos, L., & O'Hagan, A. (2009). Diagnostics for gaussian process emulators. *Technometrics*, *51*, 425–438.

Heitmann, K., D., H., Nakhleh, C., & Habib, S. (2006). Cosmic calibration. *The Astrophysical Journal*, *64*.

Higdon, D., Kennedy, M., Cavendish, J., Cafeo, J., & Ryne, R. (2004). Combining field observations and simulations for calibration and prediction. *SIAM Journal of Scientific Computing*, *26*, 448–466.

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.

Oakley, J., & O'Hagan, A. (2004). Probabilistic sensitivity analysis of complex models: a bayseain approach. *J. R. Statist. Soc. B*, *66*, 751–769.

O'Hagan, A. (2006). Bayesian analysis of computer code outputs: a tutorial. *Reliability Engineering and System Safety*, *91*, 1920–1300.

Sansó, B., & Forest, C. (2009). Statistical calibration of climate system properties. *Journal of the Royal Statistical Society*, *58*, 485–503.

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

Sobol, I. (1967). Distribution of points in a cube and approximate evaluation of integrals. *U.S.S.R Comput. Maths. Math. Phys.*, *7*, 86–112.

Stommel, H. (1961). Thermohaline convection with two stable regimes of flow. *Tellus*, *13*, 224–230.

Williams, B., Higdon, D., Gattiker, J., Moore, L., McKay, M., & Keller-McNulty, S. (2006). Combining experimental data and computer simulations, with an application to flyer plate experiments. *Journal of Bayesian Analysis*, *4*, 765–692.

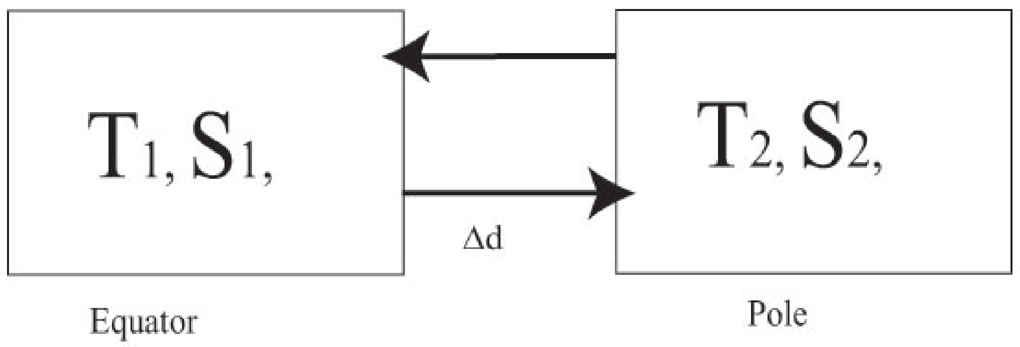

Figure 1: Stommel's 2-box model with T_1, T_2, S_1, S_2 as the mean values of each box. The density difference represents flux between the boxes.

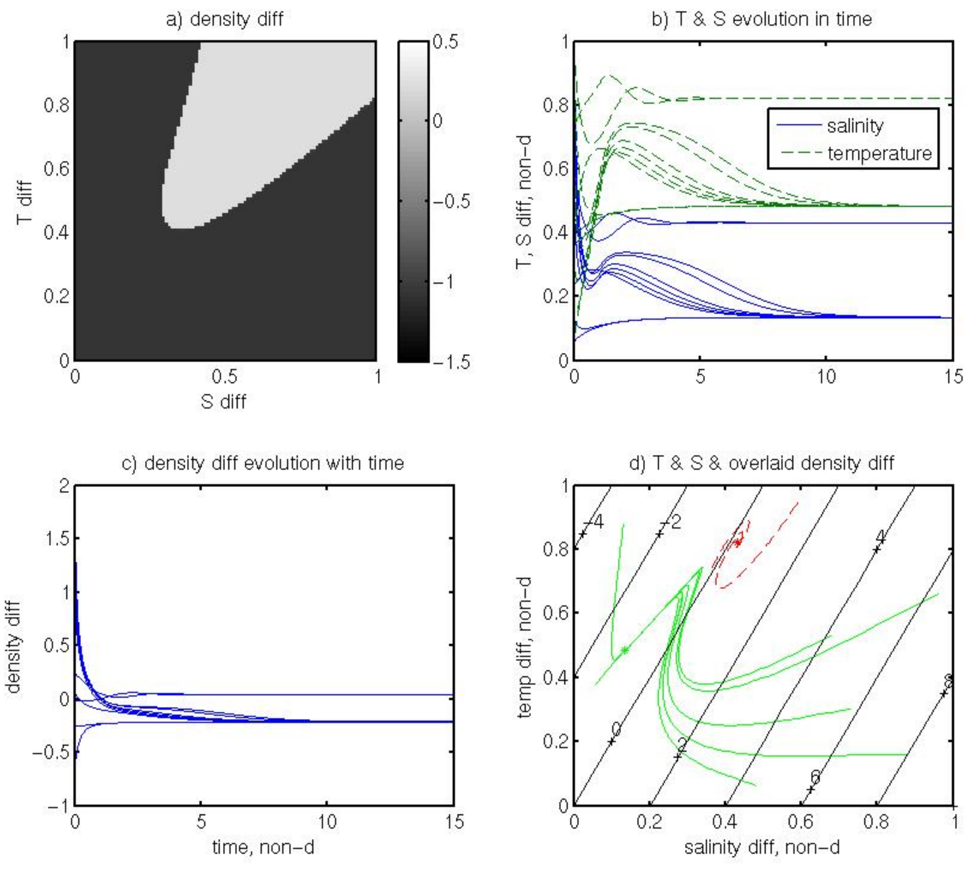

Figure 2: a) Distribution of density difference from Stommel model, illustrating the two states given the full sampling of ΔT and ΔS. b) 10 pairs of ΔT and ΔS evolving in time, given different starting values. c) density difference with time d) Trajectories of ΔT and ΔS. Stars indicate ending 2 points, one low at about -1.07 and the second around +.23 in density difference space.

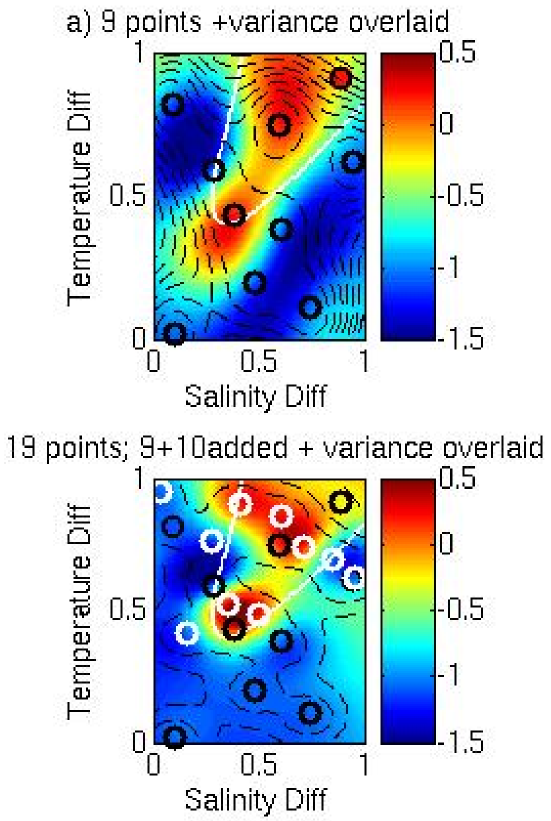

Figure 3: a) Emulator estimate of density difference (d) distribution with n = 9 and 1 unused simulator output result (red dot in top right hand corner); 9 simulator points denoted by open dots and filled dot is the one not included in emulator estimate. b) same as a) but the simulator point not used by the emulator is in the lower right corner; a blue dot with thin line. c) is the same as b) + 10 additional simulator outputs used to create the emulator. Dashed lines in all the plots are the variance of the output. The variance or uncertainty is higher for the fields that used only 9 simulator points in the emulator creation.

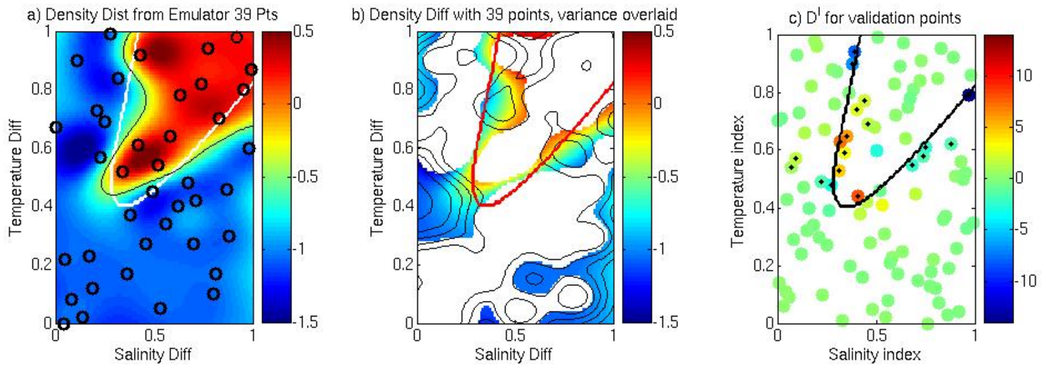

Figure 4: a) Emulator estimate of density difference(d) with n = 39 and 1 unused simulator output result, 39 points denoted by open dots. b) Same as a) except only the $f(\mathbf{x})$ within 2 standard deviations of the mid-point are shaded. Contour lines for 1) true division between low and high density difference states (red contour), and 2) relative variance for emulator estimates. c) D^I for a set of 100 validation points with the black dots indicating points excluded from D^{MD} calculation. Contour line indicates step between two outcomes of the dynamical system.

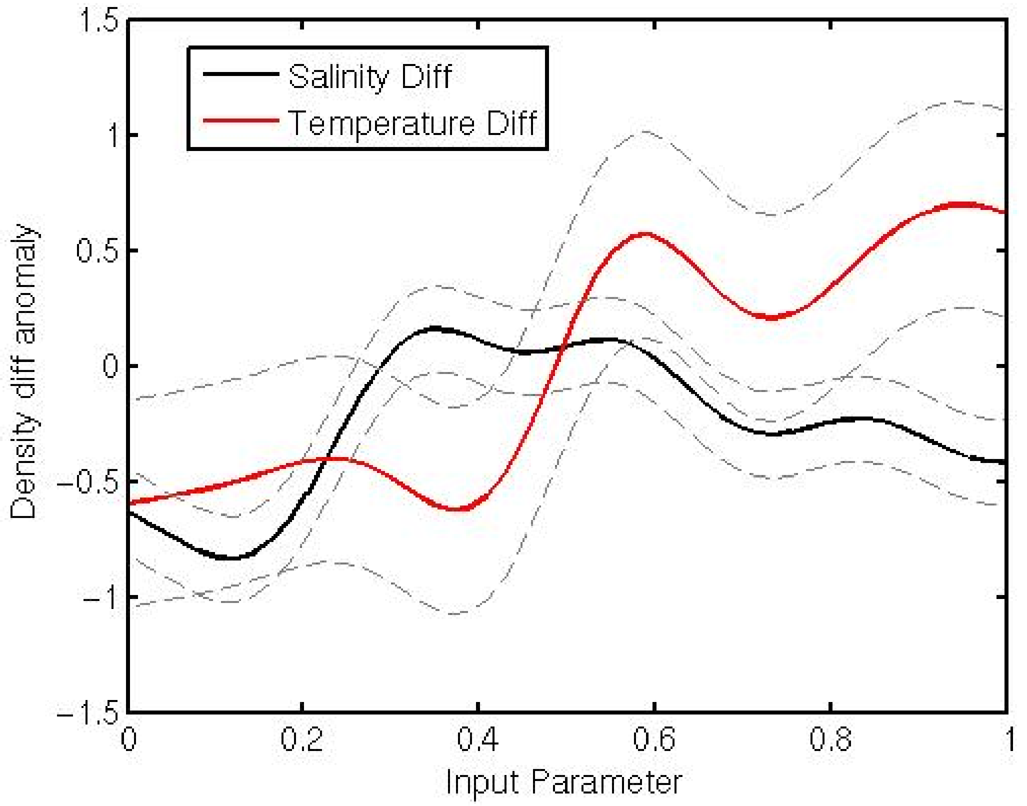

Figure 5: The sensitivity of the output metric, the density difference (d), to the input parameters (ΔS and ΔT).