Sequential Design for Achieving Estimated Accuracy of Global Sensitivities

John Guenther[1], Herbert K. H. Lee[1], and Genetha A. Gray[2]
[1]Department of Applied Mathematics and Statistics
University of California, Santa Cruz, CA 95064
[2]Quantitative Modeling and Analysis
Sandia National Laboratories, Livermore, CA 94551
(jguenthe@soe.ucsc.edu)

Abstract

Global sensitivity analysis provides information on the relative importance of the input variables for simulator functions used in computer experiments. It is more conclusive than screening methods for determining if a variable is influential, especially if a variable's influence is derived from its interactions with other variables. In this paper, we develop a method for providing global sensitivities with estimated accuracy. A treed Gaussian process serves as a statistical emulator of the black box function. A sequential experimental design makes effective and efficient use of simulator evaluations by adaptively sampling points that are expected to provide the maximum improvement to the emulator model. The method accounts for both sampling error and emulator error.

KEY WORDS: Sensitivity analsyis, Uncertainty quantification, Treed Gaussian process, computer model, emulator.

1 INTRODUCTION

Understanding the relative importance of variables and their contributions to the response is an important part of many computer simulation experiments. Knowledge of variable influence can provide feedback on the design of the simulator, and it can be used to winnow out the less important variables when faced with a large number of potential inputs. When the simulator is costly to run, such knowledge is ever more critical. Global sensitivity analysis is a variance-based method that has been proven effective in determining the importance of the input variables for computer experiments (Saltelli et al., 2010). Input variables that are determined to be not important by global sensitivity analysis can be eliminated from the computer experiment by the method of factor fixing discussed in Saltelli et al. (2008), saving processing time by reducing the dimension of the simulator input space. Alternative frameworks include variable screening (Box et al., 2005) and model order reduction (Schilders et al., 2008). Our approach represents a major advance in uncertainty quantification of the sensitivity estimations, which can be carried out to almost any desired degree of accuracy. We combine emulation and sequential experimental design to provide efficient uncertainty quantification for global sensitivity analysis.

Global sensitivity for a given variable is a measure of how much of the variance of the simulator output function, on average, is due to that variable. There are two main types of sensitivities for a given variable: 1) The first order sensitivity represents the main (linear) effects contribution of the variable to the variance of the simulator function; 2) The total sensitivity represents the total contribution of the variable to the variance of the simulator function. Using the simulator function directly to compute sensitivities would take considerable time and expense for most computer experiments, since the simulator typically requires a significant amount of time to compute a value for an input point. The alternative is to use an emulator to create a statistical model of the simulator surface which then can be used to predict large samples of input points, making efficient and effective use of the actual costly simulator evaluations. The traditional statistical model for emulation is the Gaussian process, which provides a computationally accessible nonparametric model that can accommodate a range of response functions (Santner et al., 2003). An improvement in emulators, the treed Gaussian process (TGP) emulator, has the capability of modeling functions with regions of different variability better than a traditional stationary Gaussian process, and we use TGP throughout this work (Gramacy and Lee, 2008). The emulator predictions need to be a relatively good match to the simulator evaluations if this procedure is to provide accurate results. This problem can be solved by a sequential experimental design that adds training points where most needed to enhance the emulator model accuracy. After accurate modeling of the simulator is achieved, the sensitivity estimation provides quantifiable uncertainty with estimated accuracy.

Several alternatives exist in the literature for computing global sensitivities. Methods of computing sensitivities and screening variables are discussed in Saltelli et al. (2004, 2008). Taddy et al. (2009) made use of a TGP emulator and the Monte Carlo estimation formulas given in Saltelli (2002) for the estimation of sensitivities. This is one of the first articles that used the predictions of the emulator for sensitivity computation. Farah (2011) made use of a Gaussian process emulator and formulas similar to those in Taddy et al. (2009) to estimate the sensitivities of the variables of a Leaf-Canopy Model simulator. In a recent article, Saltelli et al. (2010) discussed several Monte Carlo estimation formulas, demonstrating the total sensitivity formula developed by Jansen (1999) to be more effective than other formulas. Thus we use the Jansen formula for this research. Another important reason for this choice is that the sensitivity variances are unbiased (Jansen, 1999; Owen, 2013), in the sense that the values target the actual value of the variances instead of systematically overestimating or underestimating them. The use of a dummy variable in sensitivity estimation to distinguish influential variables from non-influential variables was demonstrated in Linkletter et al. (2006), which inspired the use of a dummy variable for all computations in this document. The standard deviation of the total sensitivity of the dummy variable is a key indicator for the convergence of the sensitivities of other variables, as we know it should converge to zero, but as we are proceeding it will take nonzero values because of the uncertainty in all of our estimations, so it provides a clear benchmark for our progress.

The method of estimating global sensitivities accurately using emulator predictions is explained in Section 2. An illustrative example is presented in Section 3. An automated stopping rule is discussed in Section 4. In Section 5, the method is applied to a real world problem involving the remediation of groundwater contamination site.

2 METHOD

The method of computing global sensitivities is presented in two parts: the formulas for the computation of the sensitivities, and the steps required to provide accuracy for the global sensitivity computations.

2.1 Computation of Sensitivities

The sensitivities needed for optimization herein are the first order and total sensitivities. Denote the input variables $x = (x_1, ..., x_k)$ and the simulator function $y = f(x_1, ..., x_k)$ where the x_i are independent. Define the matrices A and B as independent draws of random samples of size N from the input space. If Latin hypercube samples (LHS's) are used, they are scrambled by a random permutation. (Some practitioners prefer quasi-random sequences.) Each row of these matrices represents a point in input space. The evaluations of the matrix points are written as $f(A)$ and $f(B)$. Define the matrices $A_B^{(i)}$ by

$$A_B^{(i)} = [A[,1], ...A[,i-1], B[,i], A[,i+1], ..., A[,k]].$$

where k is the number of independent variables x_i, $A[,j]$ with $j = (1, ..., i-1, i+1, ..., k)$ represents the j^{th} column of matrix A, and $B[,i]$ represents the i^{th} column of matrix B. The evaluations of their points are written as $f(A_B^{(i)})$.

The first order sensitivity, $S_i, i \in \{1, \ldots, k\}$, represents the main effects contribution of input variable x_i to the variance of the output Y. The formula for the first order sensitivity (Jansen, 1999) is

$$\begin{aligned} S_i &= V_{X_i}(E_{X_{-i}}(Y|X_i))/V(Y) = 1 - E_{X_i}(V_{X_{-i}}(Y|X_i))/V(Y) \\ &\approx 1 - [\frac{1}{2N}\sum_{j=1}^{N}(f(B)_j - f(A_B^{(i)})_j)^2]/V(Y). \end{aligned} \tag{1}$$

where $E_{X_{-i}}(Y|X_i)$ is the expectation of the function, $f(x) = Y$, taken over all possible values of all the other variables $X_{-i} = (x_1, ..., x_{i-1}, x_{i+1}, ..., x_k)$ while holding $X_i = x_i$ constant; and, $V_{X_i}(E_{X_{-i}}(Y|X_i))$ is the variance of that expectation taken over all possible values of $X_i = x_i$. Likewise, $V_{X_{-i}}(Y|X_i)$ is the variance of the function, $f(x) = Y$, taken over all possible values of $X_{-i} = (x_1, ..., x_{i-1}, x_{i+1}, ..., x_k)$ while holding $X_i = x_i$ constant and $E_{X_i}(V_{X_{-i}}(Y|X_i))$ is the expectation of that variance taken over all values of $X_i = x_i$.

The total sensitivity, S_{Ti}, represents the total contribution of variable x_i, including all interactions with other variables, to the variance of the output Y. The formula for the total sensitivity (Jansen, 1999) is

$$\begin{aligned} S_{Ti} &= E_{X_{-i}}(V_{X_i}(Y|X_{-i}))/V(Y) \\ &\approx [\frac{1}{2N}\sum_{j=1}^{N}(f(A)_j - f(A_B^{(i)})_j)^2]/V(Y) \end{aligned} \tag{2}$$

where $V_{X_i}(Y|X_{-i})$ is the variance of the function, $f(x) = Y$, taken over all values of $X_i = x_i$ holding all other variables $X_{-i} = (x_1, ..., x_{i-1}, x_{i+1}, ..., x_k)$ constant; and, $E_{X_{-i}}(V_{X_i}(Y|X_{-i}))$ is the expectation of that variance taken over all values of $X_{-i} = (x_1, ..., x_{i-1}, x_{i+1}, ..., x_k)$.

An explanation of the relationship of the first order and total sensitivities can be made clear by writing all sensitivity terms for three input variables.

$$\sum_1^3 S_i + \sum_{i=1}^3 \sum_{j>i}^3 S_{ij} + S_{123} = S_1 + S_2 + S_3 + S_{12} + S_{13} + S_{23} + S_{123} = 1$$

Among these terms, the first order sensitivities are S_1, S_2, and S_3. The other terms are the interaction terms. S_{ij} is the sensitivity due to the interaction of x_i and x_j. S_{123} is the interaction of all three variables. The total sensitivities are written as:

$$S_{T1} = S_1 + S_{12} + S_{13} + S_{123},\ S_{T2} = S_2 + S_{12} + S_{23} + S_{123},\ S_{T3} = S_3 + S_{13} + S_{23} + S_{123}$$

Note that the total sensitivity of each variable includes its first order sensitivity and all interactions that include the given variable.

Equations 1 and 2 are Monte Carlo approximations. For S_{Ti}, the implied integral is simplified to that of two variables x and z with the simulator function being $f(x, z)$. Here z represents all other variables. Also, for simplicity, let the interval be [0,1] for all variables. Then sum of the differences squared is the Monte Carlo estimate of

$$\tfrac{1}{2}\int\int\int(f(x,z) - f(x',z))^2 dx dx' dz = \tfrac{1}{2}\int\int\int(f(x,z)^2 - 2f(x,z)f(x',z) + f(x',z)^2) dx dx' dz$$

Note that this leads to the simplification below since $f(x, z)$ is free of x' and vice versa.

$$\tfrac{1}{2}\int\int\int(f(x,z) - f(x',z))^2 dx dx' dz = \int\int(f(x,z)^2) dx dz - \int\int\int(f(x,z)f(x',z)) dx dx' dz$$

Then, by definition, we have that $Y = f(x, z)$, $z = X_{-i}$, and $x = X_i$. This means that

$$\frac{1}{2}\int\int\int(f(x,z)-f(x',z))^2dxdx'dz = E_{X_{-i}}(E_{X_i}(Y^2|X_{-i})) - E_{X_{-i}}((E_{X_i}(Y|X_{-i})^2) = E_{X_{-i}}(V_{X_i}(Y|X_{-i}))$$

Equation 1 for the first order sensitivity, S_i, can be justified in a similar way.

2.2 Steps Required to Provide Estimated Accuracy

The unique and original design of our approach is embodied in the steps utilizing sequential experimental design for sampling, combined with indicators that signal convergence to accurate sensitivities. The main consideration in using the predicted values from an emulator to compute sensitivities is how well does the emulator surface match the simulator surface. If there is a mismatch in regions of the input space, there will be an error induced in the sensitivity computations. If enough adaptively sampled points are added to the training point set to explore the input space adequately, the emulator surface converges to the simulator surface with the desired degree of accuracy. The number of sampled points needed is obviously dependent on the simulator function. Also, different regions of the input space of the simulator function may vary more than others. More sampled points are needed for a variable region than for a relatively flat region. Standard deviations for predicted points provided by the treed Gaussian process can be used to guide the process of adding points effectively. However, this brings up the following problem: Since the simulator surface is unknown, how can one know when adequate matching occurs? This is determined by adding sampled points in an efficient way to explore the whole input space and having indicators showing that adequate matching has occurred.

The accuracy of the estimated sensitivities, which are functions of predicted point estimates, can be obtained by summing two errors: the sampling error and the emulator error (Janon et al., 2013). Sampling error is estimated by computing the sensitivities for several iterations. The iterates of the sensitivities are then averaged to get the mean sensitivities along with their standard deviations. Sampling error is twice the standard deviation divided by the square root of the number of iterations. Emulator error is defined as the error due to the error in the predicted points. Emulator error has been estimated by Janon et al. (2013) and Taddy et al. (2009) by two different methods. In Janon et al. (2013), this is done by a numeric method for computing an upper and lower bound for the sensitivity based on the model's predicted points and their associated errors. In Taddy et al. (2009), a Bayesian approach is used in which the sensitivities are recomputed for each iterate of the Bayesian model. Here we approximate emulator error by a normal approximation which is explained later in this section. The method of Taddy et al. (2009) and the method used herein give emulator error estimations that are close in value for small standard deviations. However, the method used herein has the advantage that it takes less computation time and requires much less memory so it can be computed for large samples on a PC.

Exploring the simulator surface: As mentioned above, the emulator surface must match the simulator surface to a certain degree before accurate computations of sensitivities can be obtained. Exploring the simulator surface refers to the process of using the information obtained from the emulator (predicted points and their standard deviations) to add adaptively sampled training points to achieve effective matching. To start the process of exploring the simulator surface, data points of a relatively small LHS are evaluated for the initial training data. (A larger LHS is desirable if the simulator surface is expected to have variability throughout the input region.) Then the exploration of the surface proceeds in steps.

Preceding the discussion of these steps, some explanation of sequential experimental design and the choices made herein for exploring the simulator surface are needed. MacKay (1992) states that the single point that gives the most information is the predicted point with the highest standard deviation (ALM or Active-Learning MacKay). Cohn (1996) states that the single point that gives the most information is the one which maximizes the expected reduction in the squared error over the input space (ALC or Active-Learning Cohn). In practice, both approaches provide points that are usually close, if not identical (Gramacy and Lee, 2009). However, finding the reduction of the squared error for a sample of predicted points over the entire input space is much more computationally expensive than using the returned value for the standard deviation. Therefore, the standard deviation is the chosen criterion herein. The next choice to be made is how to select multiple points. The general consensus for selecting multiple points is that they should be spaced apart from other points (Sacks et al., 1989; Gramacy and Lee, 2009). Selecting single points would be very time consuming, since a new model would need to be created at each step. In selecting a small set of points, one needs to take into account their closeness. Selecting two points with large standard deviations close together does not give as much information as selecting two points with comparable standard deviations that are a distance apart. In Gramacy and Lee (2009), points are selected from a large set of predicted points using a treed maximum entropy design. This spaces the selected points apart from existing training points and themselves. Then this set of points is ordered by decreasing criterion (ALC or ALM) and sent one at a time to the simulator for evaluation. This process may not emphasize enough the importance of the variables in spacing the points selected. They are treated equivalently, except for their different correlation distances. The method explained next takes this into account.

Continuing with the approach used herein, at each step, the emulator (TGP) is called with the current training points to build a model and predict a large (e.g., 1000 in small dimensional spaces, more in higher dimensions) random sample of input points. Then these points are ordered by decreasing standard deviation. From these ordered points, a set of points with largest standard deviations is selected, we have found 100 to be a reasonable number. The point with the largest standard deviation is added to the training points. The remaining points in this set are further filtered by selecting a number (e.g., 25) of these which maximize their minimum distance to current training points and themselves. This minimum distance is computed based on the current total sensitivity of each variable in order to emphasize important coordinate differences. Finally, a small number (4) of these points is selected, becoming the adaptively sampled points to be evaluated by the simulator. This final selection is a random sampling based on point standard deviations. Points with greater standard deviations have a higher probability to be chosen. So, while adding the point that gives maximum improvement according to MacKay (1992), other adaptively sampled points are added to effect additional learning. Since these other points are selected probabilistically based on their standard deviation, points with lower standard deviation have a chance of being selected, ensuring the whole input space is explored.

The data computed at each step includes:

1. The first order and total sensitivities for each variable for n iterations.

2. The mean of the first order and total sensitivities, their standard deviations, and emulator

errors.

3. The plots of each variable's average sensitivity versus the number of training points for the current and preceding steps. These plots also show concurrently the standard deviation for each step, the average sensitivity plus and minus that standard deviation, and, the emulator error using the predicted points for the last iteration for that step. Since the emulator error is computed only once per step, it may display more variation initially than the standard deviation.

4. The errors and standardized errors of the adaptively sampled points. (The errors are the point evaluations minus their predicted values. The standardized errors are the errors divided by the predicted standard deviations of the points.)

The data can now be evaluated to determine if another step is needed to accurately determine the sensitivities. The six indicators for this evaluation are:

- **Indicator 1:** The values for each sensitivity should converge to a common value.
- **Indicator 2:** Consistent with indicator 2, the standard deviations from step to step should begin to stabilize.
- **Indicator 3:** The dummy variable total sensitivity and standard deviation should converge to zero.
- **Indicator 4:** The errors should decrease to relatively small values.
- **Indicator 5:** The standardized errors should be between -3 and 3 and normally distributed about 0.
- **Indicator 6:** The emulator error for each sensitivity, though starting out large, should become small, converging to zero. (See the paragraph below for a discussion of the emulator error.)

As explained above, emulator error is a measure of the difference between the sensitivity estimated by the emulator model and the sensitivity if it could be estimated by the simulator for the same input points. Emulator error is based on the predicted point values and their standard deviations for the current step's last iteration. To compute emulator error, a Taylor series expansion is done for the sensitivity formulas:

$$S^{(j)} = S(y_m) + \sum_{i=1}^{N} \partial S(y)/\partial y_i|_{y_m} \epsilon_i^{(j)} \quad \text{with } \epsilon_i^{(j)} \text{ drawn from } N(0, \sigma_i^2)$$

where $S^{(j)}$ is a linear approximation to the sensitivity at $y_m + \epsilon^{(j)}$ with $\epsilon^{(j)} = (\epsilon_1^{(j)}, ..., \epsilon_N^{(j)})$, y_m is the vector of mean predicted values ($y_m = (y_{m_1}, ..., y_{m_N})$), $S(y_m)$ is the mean sensitivity, $\epsilon_i^{(j)}$ is a draw from the normal distribution with σ_i, the standard deviation of the point prediction y_{m_i}. Note that this expansion represents a normal distribution centered at $S(y_m)$. This has a high probability of including the simulator function value $S(y_0)$ where $y_0 = (y_{0_1}, ..., y_{0_N})$ since the linear approximation is a good approximation for small σ_i and $|y_{0_i} - y_{m_i}| \leq 2\sigma_i$ for almost all differences between the predicted point values and the simulator point values. Twice the standard deviation of this distribution is the estimated emulator error:

$$\textit{emulator error} = 2\sqrt{\sum_{i=1}^{N} (\partial S(y)/\partial y_i|_{y_m})^2 \sigma_i^2}$$

This estimate usually overestimates the emulator error, so our convergence results are conservative. As the standard deviations of the predicted points decrease, emulator error converges to zero.

Regarding the errors and standardized errors, there may be some standardized errors at the beginning of the processing that have larger absolute values than 3. This is due to the model not being adequate initially. These large values should go away as convergence is achieved. Compliance for indicators 4 and 5 imply: 1) The emulator (treed Gaussian process) model is a good fit to the simulator function (Jones et al., 1998; Bastos and O'Hagan, 2009); 2) The emulator error computed from the predicted standard deviations is a good estimate. When all indicators signal convergence, a credible interval for the sensitivities can be obtained from the sum of the sampling error plus the emulator error. For the computations herein, the sensitivities are averaged over the converged region to get even better accuracy.

3 ILLUSTRATIVE EXAMPLE

To illustrate the method, a synthetic example is presented. The test function is taken from

Gramacy and Lee (2009). The function shows great variability in one quadrant (see Figure 1). There is no noise added to the function for this illustration. The treed Gaussian process is well suited to model this function since it is designed to emulate functions with regions of different variability. The equation of the function is

$$f(x_1, x_2, x_3, x_4) = (4x_1 - 2)\exp(-(4x_1 - 2)^2 - (4x_2 - 2)^2) \tag{3}$$

$$\text{where } 0 \le x_1 \le 2,\ 0 \le x_2 \le 2,\ 0 \le x_3 \le 2,\ 0 \le x_4 \le 2$$

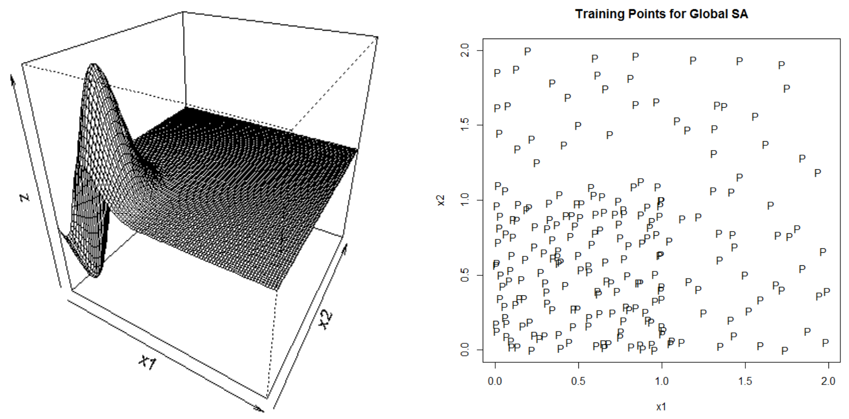

Figure 1: Test Function and Training Points for Global Sensitivity Method Illustration: The perspective of the test function given by Equation 3 on the left. It shows the variation of the function in the input space of the important variables, x_1 and x_2. The variables, x_3 and x_4, are dummy variables. Notice the function is quite variable in the lower left quadrant and flat elsewhere. This type of variability is well handled by the treed Gaussian process which divides the input space into regions based on differing variability. The graph on the right shows the training points used to model the function. It can be seen that most of the adaptively sampled points are in the region where the function has its greatest variability.

The variables with positive sensitivities are x_1 and x_2. The dummy variables x_3 and x_4 have been added to show the role of dummy variables in determining when the sensitivity estimations have converged to their actual values. The actual sensitivities are:

$$S_1 = 0.31,\ S_{T1} = 1.00,\ S_2 = 0.00,\ S_{T2} = 0.69,$$

$$S_3 = 0.00,\ S_{T3} = 0.00,\ S_4 = 0.00,\ S_{T4} = 0.00$$

The actual sensitivities were determined by 200 repetitions with random samples of size 10,000 for the A, B, and $A_B^{(i)}$ matrices using the actual function values and the formulas from Section 2.1.1.

The training points accumulated for the run are shown in Figure 1. One can see that, although training points have been added to the regions outside the variable region, most of the added training points are in the variable region. The graphs in Figure 2 show the sensitivities computed after each step of processing. For each step, 25 iterations of the sensitivities are computed. These are then averaged and their standard deviation is computed. In the graphs, the sensitivity values for each step are shown as the S's, the red (dark) line is the standard deviation, and the dashed lines are mean sensitivity value plus and minus the standard deviation. The green (light gray) line is the emulator error computed after each step. The black line is the actual sensitivity as shown above. The total number of steps is 35. The first step was an LHS of 50 training points. For each successive step, 5 training points were added, one with the highest standard deviation, and 4 by probability sampling based on their standard deviations from a set of 25 filtered points as explained above. The indicators show the following:

- **Indicator 1:** As seen in Figure 2, the sensitivity values (S's) converge to a common value in that they fall along the black horizontal line which is the actual sensitivity.
- **Indicator 2:** The standard deviations from step to step have stabilized. They are the irregular black (red) solid lines near the bottom of the graphs of Figure 2. In all the graphs above, the standard deviations have stabilized.
- **Indicator 3:** The dummy variable (x_3 and x_4) total sensitivities and standard deviations converge to zero as seen in the plots of S_{T3} and S_{T4} that are to the extreme right of Figure 2. Other sensitivities appear to have converged at about the same training set size.
- **Indicator 4:** A review of the errors in the added points at each step show that they are decreasing and relatively small for the last several steps.
- **Indicator 5:** A review of the standardized errors show they are normally distributed between -3 and 3.
- **Indicator 6:** The emulator error has dropped below the standard deviation for each variable,

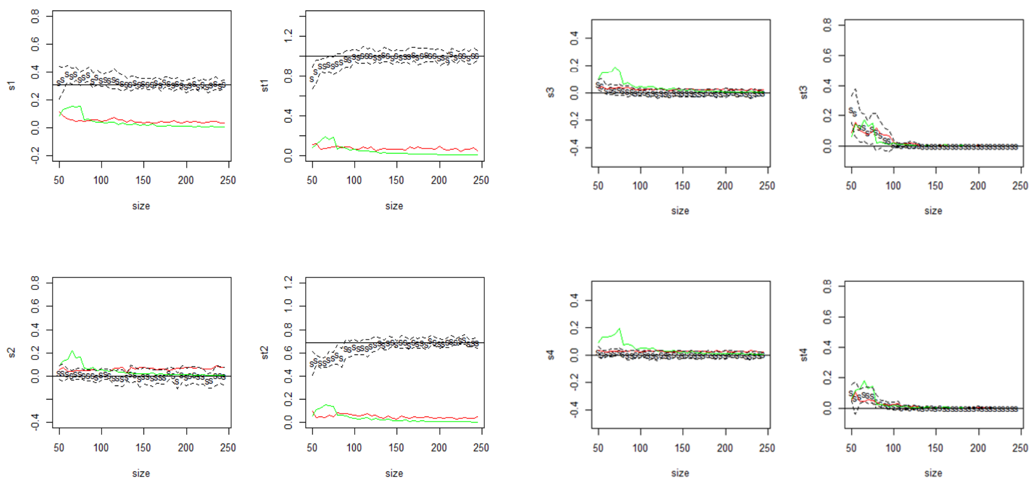

Figure 2: Graphs Showing Convergence of Global Sensitivities: The x-axis is the training point size. The y-axis is the sensitivity. The sensitivities are S_1, S_{T1}, S_2, S_{T2}, S_3, S_{T3}, S_4, S_{T4} where the S_i are the first order sensitivities for variable i, S_{Ti} are the total sensitivities of variable i, the 'S's are the mean sensitivity estimates at each processing step converging to the straight horizontal black line which is the actual sensitivity, the dark irregular (red) lines near the bottom of the graphs are the standard deviations which stabilize as the training point size increases, the light gray (green) lines near the bottom of each graph are the emulator errors which decrease as the training point size increases, and the dashed black lines are the sensitivities plus and minus the standard deviation.

except for the dummy variable total sensitivities. For the dummy variables, both emulator error and standard deviations are very small, showing the variables to be insignificant.

Table 1 summarizes the data for the steps with training set sizes from 175 to 225. This is clearly a conservative choice. Actual convergence appears to have occurred by training set size of about 150. $\overline{S}_i$ is the average of the sensitivities for the steps after convergence. The quantity Δ_e is the combined error obtained by adding the quantity $2se(S_i)$, twice the average standard error (or twice the standard deviation divided by the square root of the number of iterations), and e_e, the average emulator error. The upper and lower interval limits are obtained by adding and subtracting Δ_e from the average sensitivity. The interval covers the actual sensitivity in all cases. The actual errors in the sensitivity values are all below 0.016. More accuracy could be obtained with more iterations per step, larger sample sizes, and more processing steps. Note that x_2, while having zero first order sensitivity, is influential through its interaction terms.

Table 1: Sensitivities after Convergence: The sensitivity corresponding to each row entry is shown in the first column (Row Label). Note here that S_i is the first order sensitivity of variable i and S_{Ti} is the total sensitivity of variable i. The second column is the lower bound for the mean sensitivity, $S_i^L = \overline{S}_i - \Delta_e$. The third column is the mean sensitivity, $\overline{S}_i$, averaged over 5 processing steps. The fourth column is the upper bound for the mean sensitivity, $S_i^U = \overline{S}_i + \Delta_e$. The fifth column is the total error, $\Delta_e = e_e + 2se(S_i)$. The sixth column is twice the standard error for the sensitivity, $2se(S_i)$. The seventh column is the emulator error, e_e, for the sensitivity. The last column is the actual sensitivity, S^{act}.

Row Label	S_i^L	$\overline{S}_i$	S_i^L	Δ_e	$2se(S_i)$	e_e	S_i^{act}
S_1	0.2672	0.3252	0.3832	0.0580	0.0358	0.0222	0.31
S_{T1}	0.9395	1.0137	1.0880	0.0743	0.0552	0.0191	1.00
S_2	-0.0877	-0.0066	0.0745	0.0811	0.0626	0.0185	0.00
S_{T2}	0.6415	0.6964	0.7512	0.0548	0.0334	0.0215	0.69
S_3	-0.0457	-0.0034	0.0388	0.0422	0.0173	0.0249	0.00
S_{T3}	-0.0044	0.0005	0.0055	0.0049	0.0004	0.0045	0.00
S_4	-0.0451	-0.0036	0.0379	0.0415	0.0176	0.0239	0.00
S_{T4}	-0.0044	0.0007	0.0059	0.0052	0.0006	0.0046	0.00

4 AUTOMATIC STOPPING

In discussing global sensitivity estimation, convergence is determined by indicators, plots, errors and standardized errors for adaptively sampled points. We implement automatic stopping for our global sensitivity analysis to provide a more efficient estimation of sensitivities that does not depend on viewing graphs and reviewing data files. This automatic stopping is achieved by user controlled parameters for which default values are provided. The user need only input those control parameters that he/she wishes to change for the given application.

For sensitivity estimation, the user wants to know how accurately the sensitivities have been estimated and whether the statistical model has validity. The estimation of the accuracy of sensitivities is reliably provided if convergence of the estimates has occurred and the model fit to the simulator function has been established. These concerns are satisfied by testing certain indicators during the processing steps of the estimation procedure. Some indicators required more than one test, and some tests apply to more than one indicator. The indicators and the tests for them are explained below. When the tests are satisfied, processing is complete with provision of the sensitivity accuracy and credibility of the model fit. There are a number of control parameters that can be adjusted in these tests, and we specify our default values and provide some guidance on those that might sometimes require additional tuning.

One can ask: Why not just estimate sensitivity accuracy by summing the emulator error and the sampling error? When the model provides the desired accuracy, processing is complete. Possibly, this may be the case, but consider this: The model is transitioning as adaptively sampled points are added towards a more accurate representation of the simulator function. A small sampling error only means the sensitivities accurately reflect the model's actual sensitivities, not the simulator function sensitivities. Also, even if the emulator error is small, this does not necessarily indicate that the model fits the simulator function. As the model transitions to a more and more accurate representation of the simulator function through the addition of adaptively sampled points, the six criteria from the previous section are met. A statistical analysis can automatically verify all these occurrences in a more complete and comprehensive manner than one that focuses on just a few key statistics. Also, it does not take much more processing time since the bulk of the processing is done by the treed Gaussian process in model fitting and prediction of samples of input points. The following sections explain the tests for the indicators, processing considerations to obtain estimated accuracy, and test cases which demonstrate their use.

4.1 Indicator 1, Convergence to a Common Value

There are two tests to determine that sensitivities have converged to a common value. The first test has to do with the sensitivity variability. If variability is controlled, it confirms that convergence is occurring. In all of these tests, a fixed number of previous steps and the current step mean sensitivities and standard deviations are used, with a default of 5 steps of data. This test compares the differences between the mean sensitivities over this last set of steps and the average of the mean sensitivities to a multiple (default of 3) of the averaged standard error. If any of these differences exceed that multiple of the standard error, the test fails for that sensitivity. A schematic of this is shown below. This shows a circumstance where the test has failed. The mean sensitivities (small circles) for the previous and current five steps are not all within three average standard errors (red dashed lines) of the average mean sensitivity (green solid line). What this means is that either the model is still varying from step to step or that the sample size does not provide sufficient coverage

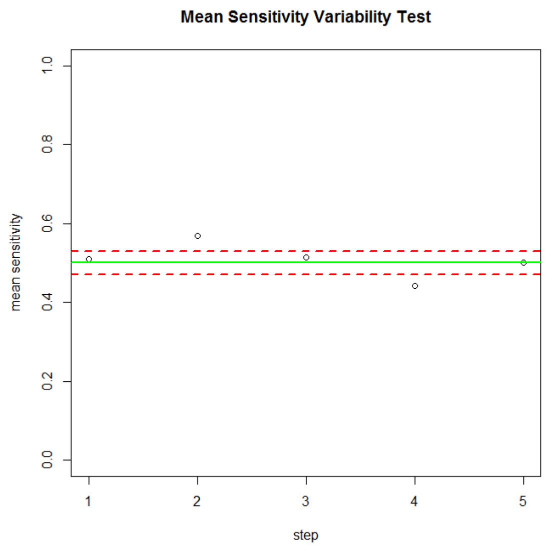

Figure 3: Indicator 1 - Test for Sensitivity Variability: The x-axis is the processing step number. The y-axis is the mean sensitivity. The mean sensitivities are shown as 'o's at each processing step. The gray (green) line is the average of the mean sensitivities over 5 steps. The black (red) dashed lines are the averaged sensitivity plus and minus three times the averaged standard error. The mean sensitivities outside the red lines indicate the variability test has failed.

of the input space. This latter is the case only if the test continues to fail with additional processing steps.

The second test is perhaps more intuitive. This tests for trending of the sensitivity. Trending occurs when the sensitivity is adjusting as the model accuracy is being improved by adaptively sampled points. Trending is occurring if a linear regression yields a slope coefficient that has an absolute value greater than a given tolerance slope level, with a default of 0.003. This tolerance slope level is chosen based on the number of steps averaged. On convergence, the slope should not show a change of more than say, 0.01, over five steps. This is another control parameter which can be changed by the user. The schematic on the next page shows a circumstance where this test has failed. The green (solid) line is an acceptable slope that increases by about 0.01 in five steps. The red (dashed) line is the fitted regression line that has a slope showing an increase in the mean sensitivity about 0.04 in the five steps.

For more robustness, especially useful for smaller sample sizes, the trend slope is computed for

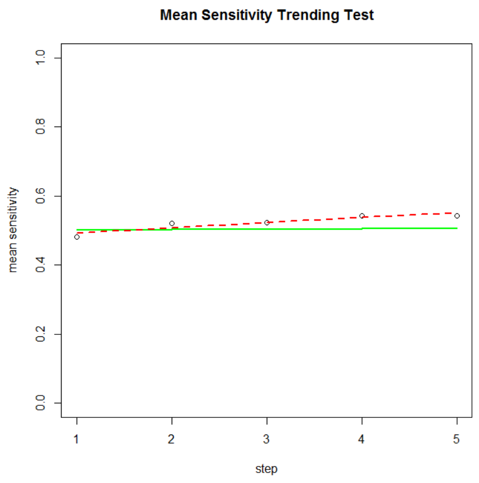

Figure 4: Indicator 1 - Test for Sensitivity Trending: The x-axis is the processing step number. The y-axis is the mean sensitivity. The mean sensitivities are shown as 'o's at each processing step. The red dashed line is the regression line for the mean sensitivities. A slope equivalent to the green (solid) line or below indicates no trending for the mean sensitivities. In this example, the trend test has failed since there is a positive trend to the sensitivity.

the sensitivity data collected over two times the averaged number of steps, usually 10 steps, when the number of processing steps is more than twice the number of averaged steps. This is done so that cyclical variations in sensitivity that occur with processing steps are factored out of the trend slope.

4.2 Indicator 2, Stabilization of Sensitivity Standard Deviations

There is one test to determine that sensitivity standard deviations have stabilized. This test also utilizes the data from the previous steps and the current step. It is expected that as the model improves the standard deviations for the sensitivities should stabilize. The previous models should be similar to the current model so the average mean standard deviation should be at least equal to or greater than three times (this multiple could be adjusted if needed) the standard deviation of the standard deviations.

Figure 5 shows an example where the test has been passed. The circles are the standard deviations for the different steps. The green (solid) line is the average mean standard deviation of the sensitivity. The orange dotted lines are the average standard deviation plus and minus two

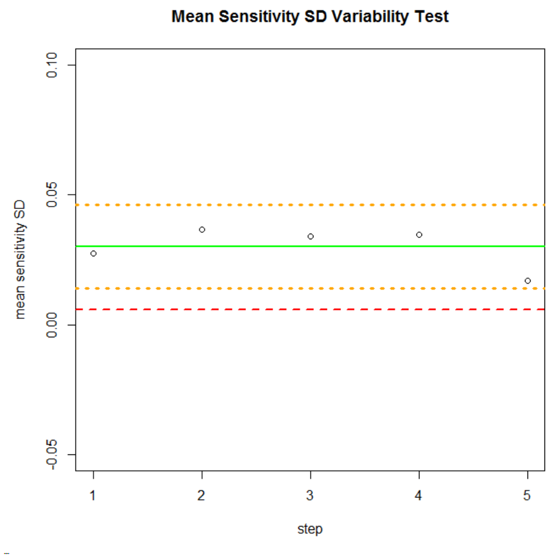

Figure 5: Indicator 2 - Sensitivity Standard Deviation Stability Test: The x-axis is the processing step number. The y-axis is the mean sensitivity standard deviation. The mean sensitivity standard deviations are as 'o's at each processing step. The green (solid) line is the average standard deviation. The orange dotted lines are the average standard deviation plus and minus two times the standard deviation of the mean sensitivity standard deviations, $s(\overline{s})$. The red dashed line is the average standard deviation minus $3s(\overline{s})$. This test passed here, with the average standard deviation is greater than $3s(\overline{s})$.

times its standard deviation, $s(\overline{s})$. The red dashed line is the mean standard deviation minus three times its standard deviation. It is just above zero indicating the average standard deviation is greater than three times $s(\overline{s})$. If this tolerance is not met, the sample size of the predicted points can be increased. Although this means more processing time, more accurate values can be obtained for the sensitivities since the increased sample size reduces sampling error. This test is not done if the average standard deviation falls below 0.01, as small standard deviations indicate that total the sampling error is small. The default of the factor of three was determined from a series of simulations that found stability in the limit for the ratio $\overline{s}/sd(\overline{s})$, with its value primarily depending upon the number of iterations used for estimating sensitivities.

For fewer iterations, this ratio goes down in value. Simulations were run for 2, 4, 8, 12, 16, 20, and 24 iterations. It was found that the ratio, $\overline{s}/sd(\overline{s})$, varied only slightly for different σ and μ but that it varied significantly for the number of iterations. In cases where it is impractical to run 25 iterations, the ratio must be set lower. Figure 6 shows the ratio to be used for the range of iterations from 2 to 24.

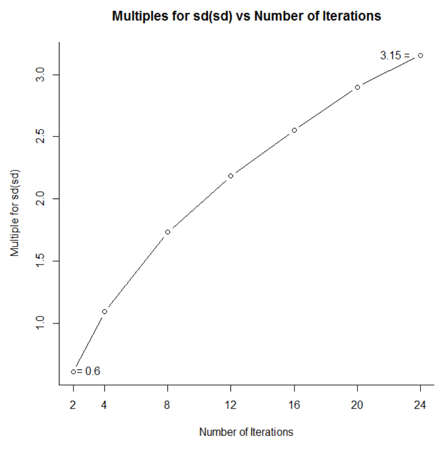

Figure 6: Ratios $\overline{s}/sd(\overline{s})$ versus Number of Iterations: The x-axis is the number of iterations for estimating sensitivities. The y-axis is the ratio, $\overline{s}/s(\overline{s})$, that is to be used for the indicator 2 test. This multiple goes down as the number of iterations is decreased since the standard deviation varies more for fewer iterations.

4.3 Indicator 3, Dummy Variable Total Sensitivity

There is a double purpose for having a dummy variable. Another variable that behaves like a dummy variable can be considered non-influential. As a convergence criterion, the dummy variable is an indicator that the statistical model distinguishes that the variable has no importance which means it is more accurately modeling the simulator surface. The test is very simple. When the dummy variable average total sensitivity falls below a specified tolerance level (default of 0.005, which represents a minimal contribution to the response), a degree of convergence has been attained by the statistical model. The total sensitivity from the previous and current steps are used to find this average as in all other tests described herein.

4.4 Indicator 4, Mean Absolute Value of Adaptively Sampled Point Errors

The errors of adaptively sampled points are polled at each step of processing to determine if their mean absolute value is small enough to indicate the function is being accurately represented. The user can specify an error tolerance or can set a limit based on the variability of the simulator function, which defaults to one twentieth of the simulator function observed output range. Errors should become progressively smaller as adaptively sampled points are added to the training points. However, discontinuous functions present a problem in that the discontinuities may cause large errors when sampling takes place near a discontinuity. In general, discontinuous functions require more adaptively sampled points to reduce the mean absolute value of the errors to the desired level. Initially, the mean absolute error of all adaptively sampled points are computed and compared to the error tolerance. When the error count exceeds the number associated with the number of steps over which data is averaged, say 25 (5 sampled points for each of 5 steps), the test compares only the last 25 errors until the error count exceeds 50 (twice the sampled points over which the steps are averaged). Then it compares the mean absolute value of the last half of the errors to the error tolerance. The plot in Figure 7 represents 100 adaptively sampled points. The last 50 absolute errors are compared to the error tolerance. The absolute errors are the little circles. The vertical blue dashed line is at the 50 error count and the vertical red solid line is at the 100 error count. The horizontal orange dashed line is the tolerance value and the horizontal green (solid) line is the mean absolute value for the errors between the vertical blue dashed line and vertical red solid line. The horizontal black line is at zero. The comparison shows this test has passed since the green (gray solid) line lies under the orange dashed line.

4.5 Indicator 5, Standardized Errors within $[-3, 3]$

In order for standard deviations of predicted points to be good estimates of the errors for points, the standardized errors of predicted points should be in the interval [-3,3]. These limits have been advocated by Jones et al. (1998) and Bastos and O'Hagan (2009). Initially, standardized errors may fall outside this interval since the emulator model may not be very accurate. However, as more adaptively sampled points are added to the training points, accuracy should improve. So the initial standardized errors are not as important as the more recent standardized errors. Therefore, the polling of standardized errors is done similarly to the polling for the errors. After enough adaptively sampled points are added, only the last half of the standardized errors are tested to determine if

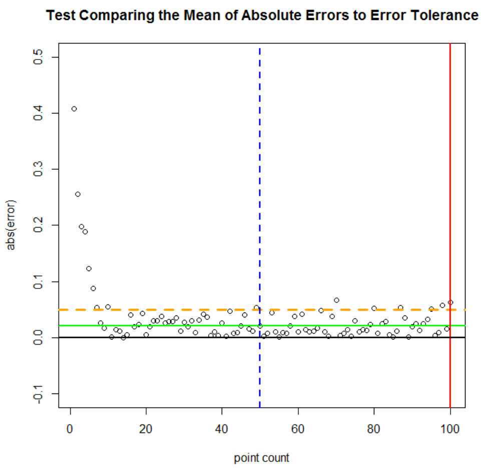

Figure 7: Indicator 4 - Absolute Value of Mean Errors Test: The x-axis is the point count for the sampled points. The y-axis is the absolute value of the "simulated" error for these points. The vertical blue dashed line at 50 is the lower bound for testing the mean absolute error and the vertical red (solid) line is the upper bound or last error. The horizontal orange dashed line is the error limit. The horizontal green (gray solid) line is the actual mean of the errors between the two bounds. In this illustration the test for mean errors has passed.

they are within the desired interval. This test fails if more than 1.5% of the absolute standardized errors are greater than 3.

For discontinuous functions, this threshold should be set to a higher percentage since adaptively sampled points near discontinuities can fall outside this interval. In this case, what is important is that other indicators show convergence. If the discontinuities are few, they represent a very small region of the simulator function input space. The region of the input space where the simulator function is continuous is dominant. It is in the continuous region that the emulator models the simulator function accurately. Convergence can be determined by the sensitivities, their standard deviations, the dummy variable total sensitivity, and the mean absolute value of the errors of adaptively sampled points. In other words, if indicators 1, 2, 3, and 4 over the number of processing steps averaged to get the sensitivities are free of errors, then the sensitivities have converged to an estimated accuracy equal to the sum of the sampling error plus the emulator error. In the case of a continuous function, this test tends to provide the same convergence information as the previous four indicators, so it may be more computationally efficient to skip this test.

The plot in Figure 8 below illustrates the testing for standardized errors for a continuous test function after 100 adaptively sampled points. A count is made of the number of absolute values of standardized errors exceeding 3 from the vertical blue dashed line at 50 to the red line at 100. Here the count of absolute values of standardized errors greater than 3 is just 1, which is within the limit of 1.5.

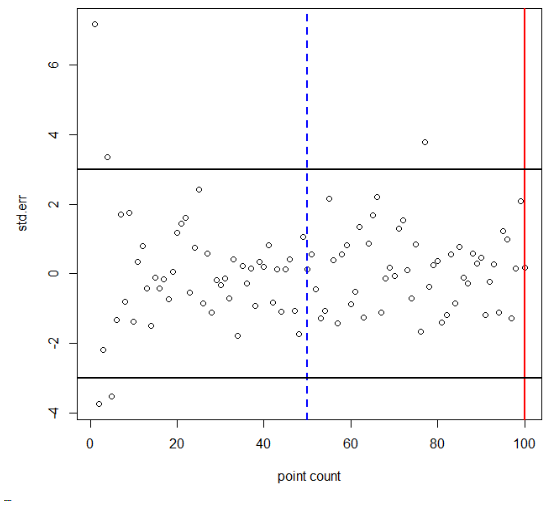

Figure 8: Test for Standardized Errors: The x-axis is the point count for the sampled points. The y-axis is the standardized error (error/SD) of these points. The vertical blue dashed line at 50 is the lower bound for testing the standardized errors and the vertical red line is the upper bound. Only one value is outside the interval $[-3, 3]$ between these lines. The standard error interval is represented by the two black horizontal lines. In this illustration, one point is within the accepted number of standardized errors outside this interval (1.5%), so the test for standardized errors has passed.

4.6 Indicator 6, Emulator Error within Tolerance Limit

As the emulator model becomes more accurate, the standard deviations of the predicted points get progressively smaller. These standard deviations are used to estimate the emulator error: the difference between the sensitivity as estimated by the emulator model and the sensitivity that would be estimated by the simulator function if it could be used to evaluate the same sample of points used for estimating the sensitivity. Since the emulator error and the sampling error make up the total error in the sensitivities, reducing the emulator error below a user controlled tolerance level is important for achieving accuracy in the estimated sensitivity. For continuous functions, this test generally matches the previous ones, and it can be skipped. For discontinuous functions, this test provides an important additional check. The tolerance level can be set based on the expected number and size of discontinuities.

4.7 Attaining a Specific Desired Accuracy for Sensitivities

The ultimate accuracy is mainly dependent on the sample size used to estimate the sensitivities. The sampling error stabilizes with increasing processing steps but it has a base value dependent on the sample size and the number of iterations that does not change. Although the number of iterations could be increased, the more efficient way to proceed is to increase the sample size after all the above tests have passed if the desired accuracy has not yet been attained. Increasing the sample size reduces the sampling error considerably. However, it does increase computation time. So, it may be better to have an effective but smaller sample size initially and then to increase the sample size once this is needed to reduce sampling error. Also, this first process run controls the emulator error.

The user can proceed as follows: Let the sensitivity function run to completion for an effective sample size. If the accuracy is sufficient, the task is done. If not, run the application using the training points from the first run with a larger sample size, such as a 50% increase, which should be chosen based on how much further one wants to reduce the sampling error. Run the application for five steps. If the results are still not adequate, increase the sample size and run it again.

4.8 Example with Automatic Stopping

We return to the test function of Section 3. Recall that x_1 and x_2 are the important variables while variables x_3 and x_4 are dummy variables. All the variability is in the lower left quadrant. For this example, the initial training point set is an LHS of 100 points. To improve accuracy, we use 1500 model-predicted points in each sensitivity matrix, compared to only 1000 points in Section 3. Indicators 1, 2, 3, and 4 are very effective in obtaining convergence with the resulting emulator errors being within 0.019. Twenty four processing steps were required. Table 2 shows the sensitivities obtained. Compare this table of sensitivities with that of Table 1. They are very similar except that the sampling errors and emulator errors are lower. This is due mainly to the larger sample size used for this processing.

Table 2: Sensitivities for Example 1 after Convergence with Automatic Stopping: The sensitivity corresponding to each row entry is shown in the first column (Row Label). Note here that S_i is the first order sensitivity of variable i and S_{Ti} is the total sensitivity of variable i. The second column is the lower bound for the mean sensitivity, $S_i^L = \overline{S}_i - \Delta_e$. The third column is the mean sensitivity, $\overline{S}_i$, averaged over 5 processing steps. The fourth column is the upper bound for the mean sensitivity, $S_i^U = \overline{S}_i + \Delta_e$. The fifth column is the total error, $\Delta_e = e_e + 2se(S_i)$. The sixth column is twice the standard error for the sensitivity, $2se(S_i)$. The seventh column is the emulator error, e_e, for the sensitivity. The last column is the true sensitivity, S^{act}.

Row Label	S_i^L	$\overline{S}_i$	S_i^L	Δ_e	$2se(S_i)$	e_e	S_i^{act}
S_1	0.2910	0.3208	0.3507	0.0299	0.0166	0.0133	0.31
S_{T1}	0.9455	0.9921	1.0388	0.0467	0.0283	0.0184	1.00
S_2	-0.0461	0.0013	0.0487	0.0474	0.0289	0.0185	0.00
S_{T2}	0.6481	0.6791	0.7101	0.0310	0.0178	0.0132	0.69
S_3	-0.0275	-0.0010	0.0256	0.0265	0.0100	0.0165	0.00
S_{T3}	0.0002	0.0012	0.0022	0.0010	0.0002	0.0008	0.00
S_4	-0.0269	-0.0005	0.0260	0.0264	0.0099	0.0165	0.00
S_{T4}	0.0000	0.0007	0.0014	0.0007	0.0001	0.0007	0.00

The actual error is within the upper and lower bounds except for the total sensitivities of the dummy variables (lines 6 and 8 in the table). Both the errors and the estimated values for these sensitivities are very small though, implying the sensitivities are for non-influential variables. Although the estimated errors are only within about 0.047, the actual errors are much smaller: < 0.011. It is interesting to see how the sensitivities varied throughout the processing and what indicators were in error at the different steps. Below are the plots of the sensitivities at each processing step along with their standard deviations and emulator errors. The 'S's are the mean sensitivities computed at each step. The horizontal straight black line is the actual sensitivity. The irregular red (dark) line is the standard deviation. The green (gray) line is the emulator error. The black dashed lines are the mean sensitivities plus and minus the standard deviations. From the graphs in Figure 9 convergence seems to have occurred when the training size reaches 160. The eleven more processing steps are due to the fact that sensitivity convergence errors occur up to step 19 and 5 more steps free of errors are done after convergence to obtain sensitivity values with accurate error estimates.

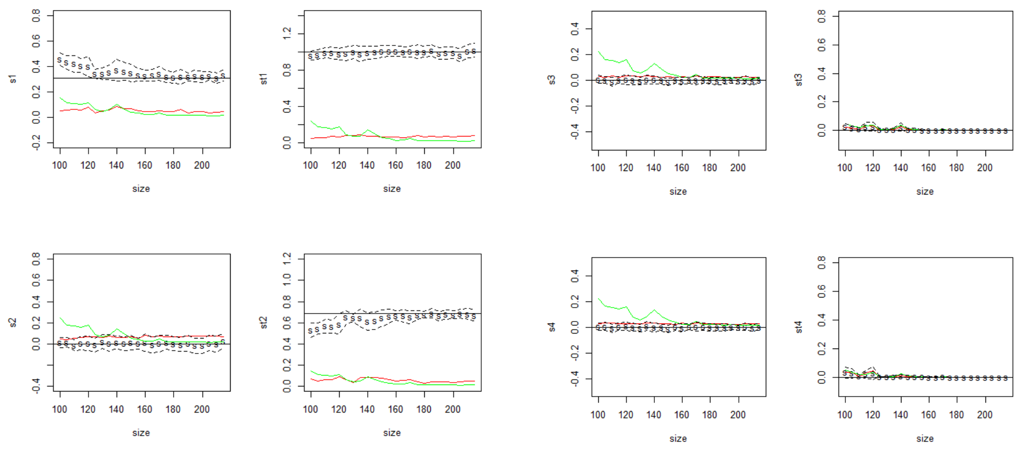

Figure 9: Graphs Showing Convergence of Sensitivities for Example 1 with Automatic Stopping: The x-axis is the training point size. The y-axis is the sensitivity. The sensitivities are S_1, S_{T1}, S_2, S_{T2}, S_3, S_{T3}, S_4, S_{T4} where the S_i are the first order sensitivities for variable i, S_{Ti} are the total sensitivities of variable i, the 'S's are the mean sensitivity estimates at each processing step converging to the horizontal straight black line which is the actual sensitivity, the irregular red (dark) lines are the standard deviations which stabilize as the training point size increases, the green (gray) lines are the emulator errors which decrease as the training point size increases, and the black dashed lines are the sensitivities plus and minus the standard deviation.

The indicators in Figure 10 are '1' if an error occurred in a sensitivity test for that indicator and zero otherwise. Notice that no sensitivity test errors occur after step 19. As this function was continuous and relatively simple, we don't include indicators 5 and 6 as they are among the first to be satisfied, and do not provide any useful information here. Had the function contained discontinuities, it would have been important to include them.

5 APPLICATION

We demonstrate this method on the Pump-and-Treat problem of Matott et al. (2011), which involves a groundwater contamination scenario based on the Lockwood Solvent Groundwater Plume Site located near Billings, Montana. (An illustration of this site is in Figure 2 of Matott et al. (2011).) Two plumes (A and B) containing chlorinated solvents developed due to industrial prac-

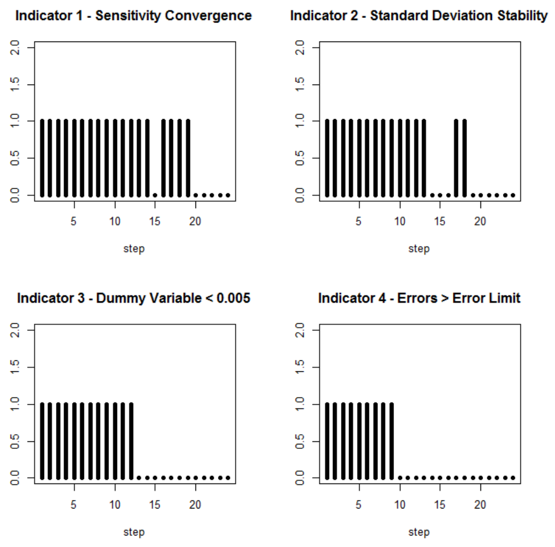

Figure 10: Indicators Showing Results of Tests for Automatic Stopping for Example 1: Indicators 1 through 4 were used to test for convergence. The x-axis is the processing step number. The y-axis is the indicator value which is 1 if any test for the indicator failed and 0 otherwise. After all indicators had no failed tests (all 0), five more steps were run having no failed tests to assure convergence.

tices near the Yellowstone river. The remediation to prevent contamination of the Yellowstone River involved drilling two pump-and-treat wells at Plume A and four pump-and-treat wells at Plume B. This problem has been modeled using a computer simulator where the inputs are pumping rates and pump locations, and the output is the cost. If a given input of these eighteen variables causes contamination of the river, a cost penalty is assessed, acting as a hidden constraint. The Lockwood plume site region is about 2 kilometers by 2 kilometers. Plume A is in the lower left part of the region where two pumps are installed. The pump locations are on the centroid of plume A but are shifted to the left within the plume. Plume B is more centrally located in the upper part of the region where 4 pumps are installed. Their locations are along the centroid of plume B approximately equally spaced. The application of this method demonstrates its effectiveness in estimating the sensitivities of the six pumping rates with the pumps in their positions given in Matott et al. (2011).

The seven input variables were the six well pumping rates and a dummy variable added to give verification that the emulator model is sufficiently accurate to detect a non-influential variable and, also, indicate if any other variables are non-influential. With seven dimensions, the number of samples of emulator predictions was set to a larger value, namely 2000. Also, the control variable setting the number of iterations per processing step for the emulator was set to 12, anticipating a long processing time for this application on a PC. With this small number of iterations, the tolerance parameter used to monitor the variability of the standard deviation of the sensitivities was set to 2 as indicated in the Section 4.2, Indicator 2, Stability of Standard Deviations. More iterations per processing step would be needed for more stable standard deviations, requiring much more processing time. The initial training set was a Latin hypercube sample of 200 points with the input bounds set to $[0, 20000]$. At each processing step, 10 more adaptively sampled points were added. All indicators were monitored for convergence. The emulator error tolerance was set to 0.01 and standardized error tolerance was keep at its default value of 1.5%. Convergence was attained by the 17_{th} processing step. Table 3 shows the sensitivities obtained.

The sensitivities for the pumping rates for pumps 1 and 2 in plume A are larger than the pumping rates sensitivities for pumps 3, 4, 5, and 6 of plume B. Another observation is that the pumping rate sensitivities get progressively larger for pumps 3, 4, 5, and 6, although the pumping rate sensitivities are very similar for pumps 5 and 6. The emulator errors are all less than 0.01. The sampling error, $2se(S_i)$, is less than 0.015 for all sensitivities. The largest total error, Δ_e, is 0.02. The small size of the dummy variable sensitivities, S_7 and S_{T7}, gives additional confirmation of the emulator model accuracy. More accuracy and more stability (less fluctuation) could be achieved by using a larger sample size and more iterations per processing step, although processing time would be increased considerably. In our opinion, this application demonstrates the capability of our global sensitivity method.

It is interesting to see how the sensitivities varied throughout the processing and what indicators were in error at the different steps. Figure 11 shows the plots of the sensitivities at each processing step along with their standard deviations and emulator errors. The S's are the mean sensitivities computed at each step. The black line is the actual sensitivity. The irregular red (solid) line is the standard deviation. The blue dotted line is the emulator error. The black dashed lines are

Table 3: Sensitivities for Pumping Rates after Convergence with Automatic Stopping: The sensitivity corresponding to each row entry is shown in the first column (Row Label). S_i is the first order sensitivity of variable i and S_{Ti} is the total sensitivity. The second column is the lower bound for the mean sensitivity, $S_i^L = \overline{S}_i - \Delta_e$. The third column is the mean sensitivity, $\overline{S}_i$, averaged over 5 processing steps. The fourth column is the upper bound for the mean sensitivity, $S_i^U = \overline{S}_i + \Delta_e$. The fifth column is the total error, $\Delta_e = e_e + 2se(S_i)$. The sixth column is twice the standard error for the sensitivity, $2se(S_i)$. The seventh column is the emulator error, e_e, for the sensitivity.

Row Label	S_i^L	$\overline{S}_i$	S_i^L	Δ_e	$2se(S_i)$	e_e
S_1	0.1970	0.2151	0.2332	0.0181	0.0122	0.0058
S_{T1}	0.2081	0.2158	0.2236	0.0077	0.0047	0.0031
S_2	0.1904	0.2086	0.2268	0.0182	0.0123	0.0059
S_{T2}	0.2014	0.2092	0.217	0.0078	0.0048	0.0030
S_3	0.0785	0.0971	0.1157	0.0186	0.0123	0.0063
S_{T3}	0.1071	0.1113	0.1155	0.0042	0.0020	0.0022
S_4	0.1200	0.1382	0.1565	0.0183	0.0121	0.0061
S_{T4}	0.1432	0.1492	0.1552	0.0060	0.0035	0.0025
S_5	0.1438	0.1621	0.1805	0.0184	0.0123	0.0060
S_{T5}	0.1594	0.1653	0.1712	0.0059	0.0033	0.0027
S_6	0.1462	0.1662	0.1862	0.0200	0.0140	0.0060
S_{T6}	0.1604	0.1669	0.1734	0.0065	0.0038	0.0027
S_7	-0.0189	0.0008	0.0205	0.0197	0.0131	0.0066
S_{T7}	0.0003	0.0005	0.0008	0.0002	0.0000	0.0002

the mean sensitivities plus and minus the standard deviations. The indicators in Figure 12 are '1' if an error occurred in a sensitivity test for that indicator and zero otherwise. Notice that no sensitivity test errors occur after step 12. As is typical for continuous functions, indicators 5 and 6 are passed early in the process; they are generally not needed for continuous functions, only for those simulators with discontinuities.

6 CONCLUSIONS

We propose a global sensitivity method herein that is efficient in terms of simulator evaluations since it uses sequential experimental design to adaptively sample points that are expected to give maximum improvement to the model accuracy. It takes into account both sampling error and emulator error. It has the capability of automatic stopping controlled by parameters that can be set by the user to attain accuracies commensurate to their requirements. We have demonstrated our method on an example from hydrology, but it is applicable to virtually any field, for simulators

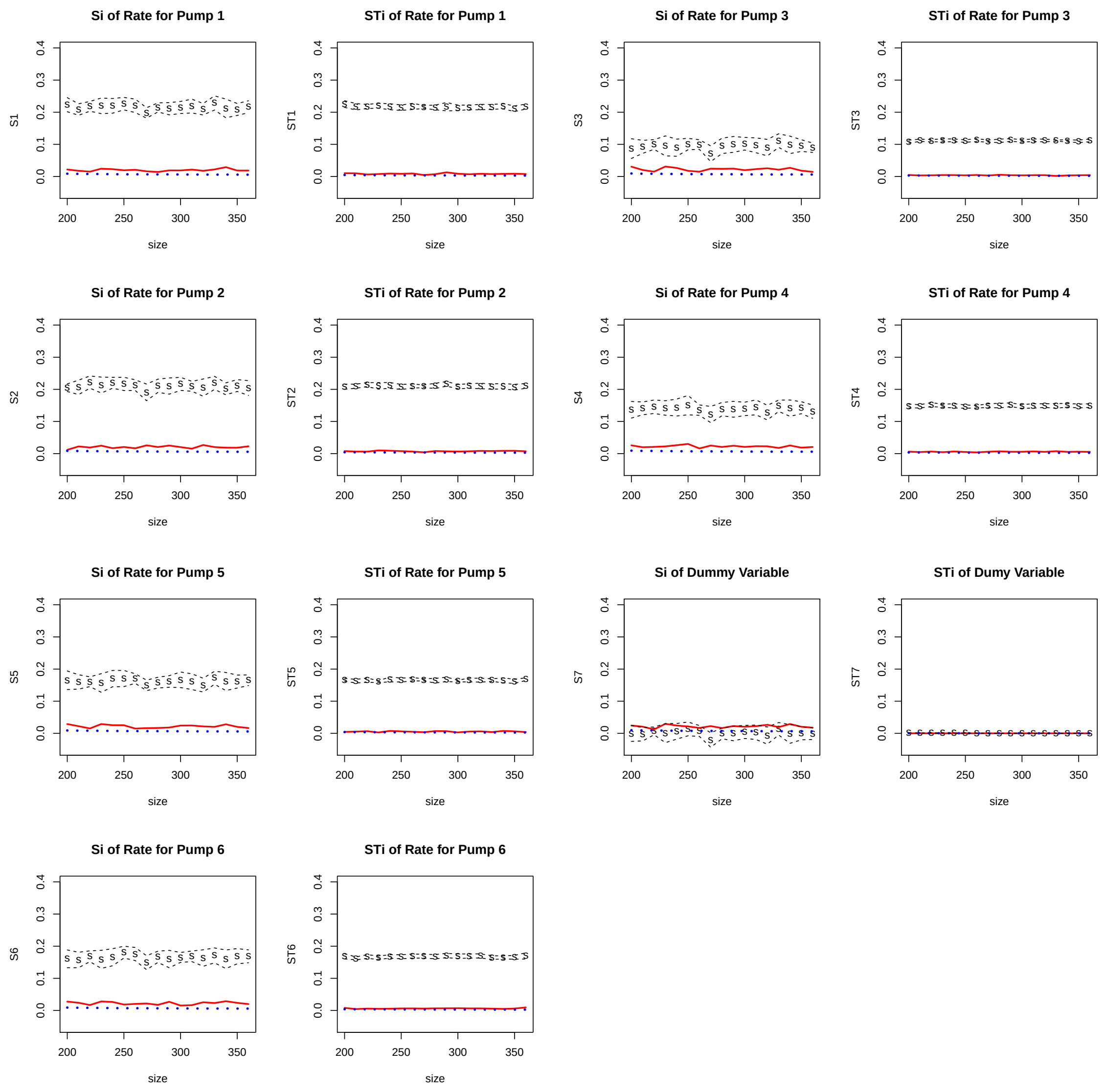

Figure 11: Graphs Showing Convergence of Sensitivities for the Lockwood Pump-and-Treat Problem with Automatic Stopping: The x-axis is the training point size. The y-axis is the sensitivity. The sensitivities are S_1, S_{T1}, S_2, S_{T2}, S_3, S_{T3}, S_5, S_{T5}, S_6, S_{T6}, S_7, S_{T7}, where the S_i are the first order sensitivities and S_{Ti} are the total sensitivities, the 'S's are the mean sensitivity estimates at each processing step which are converging in value, the red (gray) solid lines are the standard deviations which show relative stability, the blue dotted lines are the emulator errors which have remained small throughout for this application, and the black dashed lines are the sensitivities plus and minus the standard deviation. A larger predicted sample size and more iterations per processing step would show much less fluctuation, although these fluctuations are within the tolerances used for convergence.

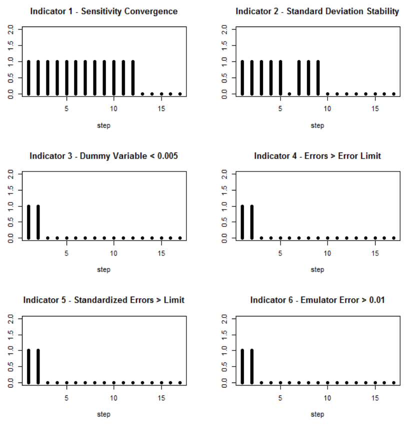

Figure 12: Indicators Showing Results of Tests for Automatic Stopping for the Lockwood Pump-and-Treat Problem: Indicators 1 through 6 were used to test for convergence. The x-axis is the processing step number. The y-axis is the indicator value which is 1 if any test for the indicator failed and 0 otherwise. After all indicators had no failed tests (all 0), five more steps were run having no failed tests to assure convergence.

with both continuous and discontinuous outputs.

Acknowledgments

Partial funding was provided by Sandia National Laboratories and by National Science Foundation grant DMS 0906720.

REFERENCES

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

G. E. P. Box, S. Hunter, and W. G. Hunter. *Statistics for Experimenters: Design Innovation, and Discovery*. Wiley, 2005.

D. Cohn. Neural networks exploration using optimal experiment design. *Neural Networks*, **6**(9): 679–686, 1996.

M. Farah. *Bayesian Nonparametric Methods for Emulation, Sensitivity Analysis, and Calibration of Computer Simulators.* PhD thesis, UC Santa Cruz, 2011.

R. Gramacy and H. Lee. Bayesian treed gaussian process models with application to computer modeling. *Journal of the American Statistical Association*, **103**(483):1119–1130, 2008.

R. Gramacy and H. Lee. Adaptive design and analysis of supercomputer experiments. *Technometrics*, **51**: 130–145, 2009.

A. Janon, M. Nodet, and C. Prieur. Uncertainties assessment in global sensitivity indices estimation from metamodels. *International Journal for Uncertainty Quantification, to appear*, 2013.

M. Jansen. Analysis of variance designs for model output. *Computer Physics Communications*, **117**:35–43, 1999.

D. Jones, M. Shonlau, and W. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, **13**: 455–492, 1998.

C. Linkletter, D. Bingham, N. Hengartner, D. Higdon, and K. Ye. Variable selection for gaussian process models in computer experiments. *Technometrics*, **48**:478–490, 2006.

D. MacKay. Information-based objective functions for active data selections. *Neural Computation*, **4**:589–603, 1992.

L. S. Matott, K. Leung, and J. Sim. Application of matlab and python optimizers to two case-studies involving groundwater flow and contaminant transport modeling. *Geospatial Cyberinfrastructure for Polar Research*, **37**(11):1894–1899, 2011.

A. Owen. Variance components and generalized sobol ́ indices. *SIAM/ASA Journal of Uncertainty Quantification*, 1:19–41, 2013.

J. Sacks, W. Welch, T. Mitchell, and H. Wynn. Design and analysis of computer experiments. *Statistical Science*, **4**: 409–435, 1989.

A. Saltelli. Making best use of model evaluations to compute sensitivity indices. *Computer Physics Communications*, **145**: 280–297, 2002.

A. Saltelli, S. Tarantola, F. Campolongo, and M. Ratto. *Sensitivity Analysis in Practice.* John Wiley and Sons Ltd., 2004.

A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, and S. Tarantola. *Global Sensitivity Analysis.* John Wiley and Sons Ltd., 2008.

A. Saltelli, P. Annoni, F. Campolongo, M. Ratto, and S. Tarantola. Variance based sensitivity analysis of model output. design and estimator for total sensitivity index. *Computer Physics Communications*, **181**: 259–270, 2010.

T. Santner, B. Williams, and W. Notz. *The Design and Analysis of Computer Experiments.* Springer, 2003.

W. H. Schilders, H. A. Vorst, and J. Rommes, editors. *Model Order Reduction: Theory, Research Aspects and Applications.* Springer, 2008.

M. Taddy, H. Lee, G. Gray, and J. Griffin. Bayesian guided pattern search for robust local optimization. *Technometrics*, **51**:389–401, 2009.