

# Global sensitivity analysis using sparse grid interpolation and polynomial chaos

Gregery T. Buzzard

*Department of Mathematics  
Purdue University  
West Lafayette, IN 47907*

---

## Abstract

Sparse grid interpolation is widely used to provide good approximations to smooth functions in high dimensions based on relatively few function evaluations. By using an efficient conversion from the interpolating polynomial provided by evaluations on a sparse grid to a representation in terms of orthogonal polynomials (gPC representation), we show how to use these relatively few function evaluations to estimate several types of sensitivity coefficients and to provide estimates on local minima and maxima. First, we provide a good estimate of the variance-based sensitivity coefficients of Sobol' [1] and then use the gradient of the gPC representation to give good approximations to the derivative-based sensitivity coefficients described by Kucherenko and Sobol' [2]. Finally, we use the package HOM4PS-2.0 [3] to determine the critical points of the interpolating polynomial and use these to determine the local minima and maxima of this polynomial.

*Keywords:* Sparse grid, polynomial interpolation, stochastic collocation, polynomial chaos, sensitivity analysis, optimization

---

## 1. Introduction

A common task in fitting a model to data is to find parameters  $p = (p_1, \dots, p_n)$  to minimize some cost function,  $C(p)$ , often a sum of squared differences between model output and experimental data. This is a particularly difficult task when the dimensionality of the parameter space is large and the dependence of  $C$  on  $p$  is nonlinear. One approach to this problem is to sample the function at some set of points and try to estimate relevant

quantities, such as various types of sensitivity coefficients and the location of local minima, from this sample. Often, these samples are used to construct a simpler model (e.g., linear, polynomial, sum of Gaussians, etc.) that may be used to approximate the original model in a computationally inexpensive way. Such approximate models are described with various terms, including metamodels, surrogate models, response surfaces and model emulators. In settings in which the sampling points are given in advance, common approaches include RS-HDMR, cut-HDMR, ANOVA decomposition, kriging, and moving least squares. In settings in which the sampling points may be chosen at will, two common approaches are sparse grid interpolation and generalized polynomial chaos (gPC) using cubature<sup>1</sup>. In this paper we focus on these last two metamodels, the relationship between them, and their application to computing global sensitivity coefficients and global maxima and minima.

More precisely, sensitivity methods can be divided into global (the focus in this paper) and local, while global methods can in turn be divided into screening methods, non-parametric methods, variance-based methods, and moment-independent or density based methods. The classic paper on screening methods is [4], which details a method for sampling model outputs over a high-dimensional input space in order to estimate the mean and variance of partial derivatives of the output with respect to each input. A number of non-parametric approaches for global SA, including locally weighted regression, additive models, projection pursuit regression, and recursive partitioning regression are detailed in [5]. Further non-parametric methods, along with a description for using these methods to estimate values and confidence intervals for variance-based sensitivity coefficients are given in [6]. An overview of global SA methods is provided in [7], which also introduces a new, moment-independent importance measure; this measure is discussed also in [8]. Many global SA methods are discussed in [9]. In terms of other metamodels, an overview of Kriging and discussion of bootstrapping to estimate the variance in the Kriging predictor is given in [10]. A discussion of gPC and its application to computing sensitivity coefficients appears in both [11] and [12].

Another approach to constructing a polynomial metamodel is sparse grid

---

<sup>1</sup>Abbreviations in the text: generalized polynomial chaos (gPC), Monte Carlo (MC), Chebyshev-Gauss-Lobatto (CGL)

interpolation, which has been used widely in recent years as a means of providing a reasonable approximation to a smooth function,  $f$ , defined on a hypercube in  $\mathbb{R}^n$ , based on relatively few function evaluations [13]. This method produces a polynomial interpolant using Lagrange interpolating polynomials based on function values at points in a union of product grids of small dimension [14, 15]. However, for many purposes, there are computational advantages to a representation in terms of orthogonal polynomials; such a representation is also known as a generalized polynomial chaos (gPC) representation. Most relevant for the discussion here is the efficient calculation of the Sobol' sensitivity coefficients of a polynomial in gPC form.

In this paper we start with the efficient conversion from an interpolating polynomial in Lagrange form to the gPC form as described in [16]. We combine this with the efficient calculation of the Sobol' sensitivity coefficients of [12] and [11] to produce an efficient algorithm for estimating these coefficients using a relatively small number of function evaluations. As seen in numerical examples, this method is both accurate and efficient for smooth functions when compared with other approaches for estimating these values. We also show how to use the gPC representation to estimate two derivative-based sensitivity measures discussed in [2]. Finally, we discuss the use of polynomial homotopy methods for finding the critical points of the interpolating polynomial [3]. In cases in which the global maximum or minimum does not lie on the boundary of the interpolating hypercube, this allows us to find the global minimum or maximum (within the hypercube) directly. In addition to these applications, we note that sparse grid interpolation likely has applications in the context of other global SA methods as well. We leave this as a topic for future research.

This research is partially supported under NSF grant DMS-0900277. The NSF had no role in the completion of this work or in manuscript preparation.

## 2. Theory

In this section we provide further background on sparse grid interpolation, generalized polynomial chaos, and sensitivity analysis.

### 2.1. Sparse grid interpolation and polynomial chaos

In one variable, Lagrange interpolation proceeds by selecting a set of points,  $x_1, \dots, x_n$  and degree  $n - 1$  polynomials  $L_j$  so that  $L_j$  is 1 at  $x_j$  and 0 at  $x_k$  for  $k \neq j$ . Given function values  $f(x_j)$ , we obtain an interpolating

polynomial that agrees with  $f$  at each  $x_j$  by taking  $P(x) = \sum_{j=1}^n f(x_j)L_j(x)$ . For a well-chosen set of points, such as the Chebyshev-Gauss-Lobatto (CGL) points, and for smooth  $f$ , the resulting polynomials converge to  $f$  quite rapidly as the number of interpolating points is increased [14]. The simplest generalization of this to  $d$  dimensions is to use a full product grid obtained from the product of  $d$  one-dimensional interpolating sets, with Lagrange polynomials obtained by taking products of the one-dimensional Lagrange polynomials. However, since the resulting number of points is exponential in  $d$ , this method is not practical for anything but small  $d$ . As described in [14] and elsewhere, Smolyak [15] devised a method in which the interpolating points in a hypercube,  $[0, 1]^d$ , in  $d$  dimensions are obtained as a union of smaller product grids. Examples of full and sparse grids in two dimensions appear in Figure 1. In the sparse grid, the dotted points show the locations

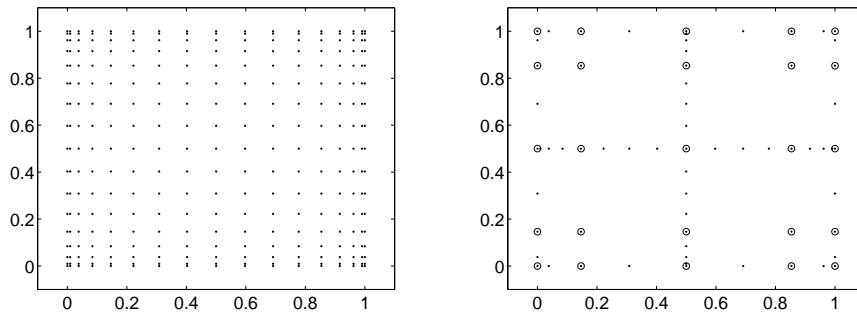


Figure 1: Full and sparse grids in 2 dimensions using CGL points. Left: The full grid is obtained as a product of the points along the coordinate axes in the sparse grid. Right: The dotted points show the entire sparse grid of nesting depth 4, while the circled points show one of the component product subgrids.

of all the points in the sparse grid, while the circled points show the points in one product subgrid. This example shows that using sparse grid points, the function is sampled heavily along the coordinate axes, then somewhat less near corners and boundaries, and even less in the interior. With these values, we can again represent  $f$  as a sum of products of one variable Lagrange polynomials. On each of the smaller product grids in the sparse grid, this is a simple tensorization of the one-dimensional Lagrange representation given above. The contributions for different subgrids are then summed with appropriate weights to produce the interpolating polynomial on the entire

sparse grid. The use of a nested sequence of points in one dimension, such as appropriate subsets of CGL points, implies that the good convergence properties of interpolation in one dimension carry over to higher dimensions. Details of the sparse grid construction and precise estimates on the rate of convergence may be found in [14]. With appropriate modifications, sparse grid interpolation may also be performed on hypercubes in which some of the sides are unbounded [16].

In the gPC representation of  $f$ , the function is still represented as the sum of products of polynomials in one variable, but now the underlying one dimensional polynomials are orthogonal in the weighted  $L^2$  sense. That is, over some fixed interval,  $I$ , in the real line, and with some fixed positive weight function  $w(x)$ , they satisfy  $\int_I P_j(x)P_k(x)w(x)dx = \delta_{jk}$ . A common example is the Legendre polynomials, which are orthogonal over the interval  $[-1, 1]$  using the weight function  $w(x) = 1$ . The weight function often corresponds to a probability distribution. More details may be found in [13] and [12].

## 2.2. Sensitivity analysis

As described in [1, 17] and elsewhere, a useful decomposition of a function  $f(x) = f(x_1, \dots, x_n)$  defined on a hypercube in  $\mathbb{R}^n$  is to write it as a normalized sum of functions that depend on a specified subset of the variables:

$$f(x) = f_0 + \sum_{i=1}^n f_i(x_i) + \sum_{1 \leq i < j \leq n} f_{ij}(x_i, x_j) + \dots + f_{12\dots n}(x_1, \dots, x_n),$$

with a (weighted) orthogonality condition imposed on pairs of functions in this decomposition and a 0 mean condition in each variable separately imposed on each function individually. In various contexts this decomposition is known as the Sobol' decomposition, the ANOVA decomposition, or the HDMR representation of  $f$ . As described in [17, 12], one method for achieving this decomposition is by expanding  $f$  in terms of a basis of tensored one-dimensional orthogonal polynomials. As noted above, such a representation is also known as a gPC expansion of  $f$ . A gPC representation is essentially a refinement of the above decomposition in which each component function in the sum above is written as a sum of a product of one-dimensional orthogonal polynomials. The gPC expansion in terms of several different families of orthogonal polynomials (corresponding to different weight functions) is discussed in detail in [18].

### 2.2.1. Variance-based sensitivity coefficients

As seen in [1], the Sobol' decomposition gives rise to variance-based global sensitivity coefficients. Let  $H^n$  denote the hypercube  $[0, 1]^n$  and assume that the decomposition above takes place on this set. For a function,  $g$ , on this set, define the variance of  $g$  to be

$$D(g) = \int_{H^n} g^2(x)dx - \left( \int_{H^n} g(x)dx \right)^2.$$

In terms of the Sobol' decomposition given above, the Main Effect sensitivity coefficient for  $x_j$  is  $S_j = D(f_j)/D(f)$ . This is generalized to interaction coefficients,  $S_M$  by replacing  $f_j$  by  $f_M$ , where  $M$  is any subset of  $\{1, \dots, n\}$ . Also, the Total Effect sensitivity coefficient  $S_{T_j}$  is the sum of  $S_M$  over all  $M$  containing  $j$ .

A variety of approaches to the estimation of  $S_j$  and  $S_{T_j}$  have been given in the literature. Since the coefficients are obtained from integrals, Sobol' [1] suggested the use of Monte Carlo (MC) or quasi Monte Carlo methods. The MC method has the advantage that the rate of convergence, which is  $O(1/\sqrt{N})$ , where  $N$  is the number of function evaluations, is independent of dimension. However, this rate is fairly slow. Quasi-MC methods, in which sampling is deterministic but still retains some features of randomness, can improve this rate to  $O((\log n)^d/N)$ , as seen in Kucherenko, et al. [19]. Both of these methods work well, particularly when the function under consideration is not smooth, such as the  $g$ -function of Sobol' and other functions with absolute values. Saltelli, et al. [20], describe the Extended FAST method and show that it has some advantages relative to the original method of Sobol'.

When  $f$  is  $C^m$  smooth (has continuous derivatives up to order  $m$ ), then other methods may yield faster convergence to the sensitivity coefficients. Crestaux, et al. [12], show that all of the coefficients  $S_M$  may be computed very efficiently based on the gPC expansion of  $f$ . The computation of  $S_M$  is essentially the computation of the square of the (weighted)  $L^2$  norm of  $f_M$ . Since  $f_M$  is obtained as a sum of gPC polynomials that are orthogonal in this  $L^2$  sense, the  $L^2$  norm squared of  $f_M$  is simply the sum of squares of the coefficients of  $f_M$  in its gPC expansion. Since the gPC expansion may be obtained from a cubature rule (a method for approximating the integral of a function over a hypercube based on specified evaluation points), the convergence of the coefficients is directly related to the convergence of the

cubature rule, which is typically  $O((\log N)^{dm}/N^m)$  for  $C^m$  functions (the exponent of  $\log N$  is not quite precise but is correct in spirit). However, the cubature rules with the best rates of convergence are not nested. That is, the evaluation points used in a cubature rule of one level of accuracy are not contained in a cubature rule of a higher level of accuracy. This is evident, for instance, in the Gaussian quadrature nodes in one variable.

Sparse grid interpolation provides an alternative to more general cubature rules since the resulting interpolating polynomial may be integrated directly to estimate the integral of the original function. In [21], sparse grid interpolation using nested evaluation points was used to estimate the coefficients  $S_j$  and  $S_{T_j}$ . It was shown there that the convergence rate of this method is essentially the rate shown above for cubature. Additionally, numerical results indicate that this method converges significantly more quickly than some quasi-MC methods or Extended FAST when  $f$  is smooth. Additionally, the nested property of the sparse grids implies that we may reuse existing function evaluations when adding more points for a more accurate estimate of the sensitivity coefficients. A drawback of the method in [21] is that computation of the interaction effects is difficult relative to the computation using the gPC expansion.

Buzzard [16] gives an efficient algorithm for conversion from the sparse grid interpolating polynomial in Lagrange form to the gPC form of the same polynomial. By combining this algorithm with Crestaux et al.'s method for computing  $S_M$  from the gPC representation, we obtain an efficient, accurate method for estimating all the values  $S_M$  and  $S_{T_j}$  for smooth functions. This method allows for refinement of these estimates by adding points systematically. As seen in Section 3, this method compares favorably with quasi-MC and Extended FAST when the underlying function is sufficiently smooth.

### 2.2.2. Derivative-based sensitivity coefficients

As described by Campolongo, Cariboni, and Saltelli [22] and further developed by Sobol' and Kucherenko [2], an alternative to variance-based sensitivity measures is to consider sensitivity coefficients based on the value of the partial derivatives  $\partial f/\partial x_j$ . For  $f$  defined on the unit hypercube,  $H^n$ , the coefficient introduced in [22] is shown in [2] to be an approximation to

$$\mu_i = \int_{H^n} \left| \frac{\partial f}{\partial x_i} \right| dx.$$

As an alternative, Sobol' and Kucherenko [2] also define

$$\nu_i = \int_{H^n} \left( \frac{\partial f}{\partial x_i} \right)^2 dx$$

and show that  $\mu_i \leq \sqrt{\nu_i}$ , that  $\nu_i \leq C\mu_i$  if  $|\partial f/\partial x_i| \leq C$ , and that  $S_{T_i} \leq \nu_i/(\pi^2 D)$ , where  $D$  is the total variance of  $f$ . Since  $\mu_i$  is the  $L^1$ -norm of  $\partial f/\partial x_i$  and  $\nu_i$  is the square of the  $L^2$ -norm of  $\partial f/\partial x_i$ , we refer to these as the  $L^1$  and  $L^2$  derivative sensitivity coefficients, respectively.

It was shown in [19] that quasi-MC integration methods based on Sobol' sequences provide an efficient method for estimating  $\mu_i$  and  $\nu_i$  and that convergence with quasi-MC is much faster in these cases than for estimation of the Sobol' sensitivity indices.

On the other hand, the gPC representation of the interpolating polynomial for  $f$  provides an alternative method for estimating  $\mu_i$  and  $\nu_i$ . That is, the gPC representation may be differentiated directly to yield a gPC approximation,  $g_i$ , to  $\partial f/\partial x_i$ . In the case of  $\nu_i$ , we may compute the square of the  $L^2$ -norm of  $g_i$  immediately by taking the (weighted) sum of squares of the coefficients in the gPC expansion. This gives an efficient and accurate method to estimate  $\nu_i$ . In the case of  $\mu_i$ , the presence of the absolute value means that  $\mu_i$  requires the integration of a function which is continuous but generally not differentiable throughout the hypercube. In this case, we may integrate  $|g_i|$  using cubature methods, but the convergence is likely to be slower relative to the convergence for  $\nu_i$  or even the variance-based coefficients. In the next section we compare these method for estimating the derivative-based sensitivity coefficients and find that the gPC approach compares favorably with quasi-MC even for computing  $\mu_i$ .

### 3. Results and Discussion

In this section we provide results of numerical experiments using the methods described in the previous section. All computations were performed in Matlab on a Dell Precision PWS690 with an Intel Xeon running at 3GHz with 3GB of RAM.

We used the test functions labeled OSCILLATORY ( $F_1$ ) and GAUSSIAN ( $F_2$ ) in [14],

$$F_1(x) = \cos \left( 2\pi w_1 + \sum_{i=1}^d c_i x_i \right), \quad F_2(x) = \exp \left( - \sum_{i=1}^n c_i^2 (x_i - w_i)^2 \right)$$

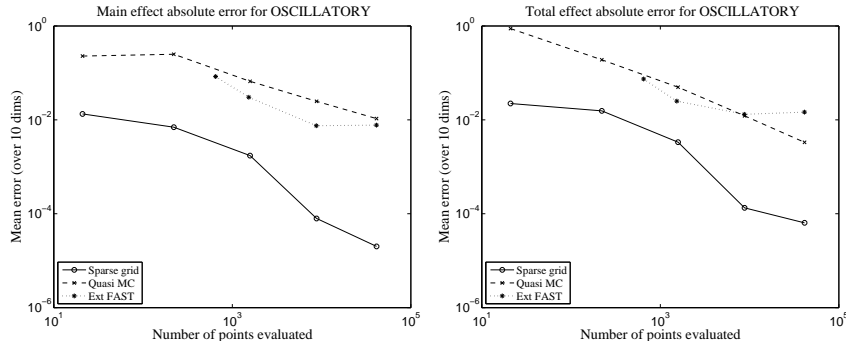


Figure 2: Main and total effect errors for the function OSCILLATORY. The sensitivity coefficients for each dimension were calculated using each of the sparse grid method, quasi-MC using the Sobol’ sequence, extended FAST, and analytically. The errors shown are the mean over the 10 coordinate dimensions.

where  $c_i$  and  $w_i$  were chosen at random as indicated in [14] and the dimension,  $d$ , was fixed at 10. The domain of definition for both functions is  $[0, 1]^d$ . These test functions were proposed originally in [23]. Other choices of smooth test functions gave results similar to those shown here. The sparse grid for a given depth was created using the Matlab package spinterp, version 5.1.1 [24]. The resulting functional values were then used as input to the algorithm described in [16] for conversion to gPC form using Legendre polynomials as basis. The variance-based sensitivity coefficients were then computed as described in [12], while the derivative-based coefficients were computed by first taking the gradient and then approximating the appropriate integral. For the calculations involving quasi-MC methods, we used the academic version of the Sobol’ sequence generator available from the British-Russian Offshore Development Agency (BRODA) [25]. For the calculations involving extended FAST, we used the implementation in SimLab 3 [26].

### 3.1. Variance-based sensitivity coefficients

We estimated the main and total effect Sobol’ sensitivity coefficients using 4 different methods: quasi-MC, extended FAST, the sparse grid/gPC method given here, and analytically using Mathematica. Using the analytic values as the reference, we computed the mean error taken over the 10 dimensions. As seen in Figure 2, the sparse grid method displayed considerably better accuracy and faster convergence for the test function  $F_1$  than either quasi-MC or extended FAST. In Figure 3, we plot selected values of the main effect

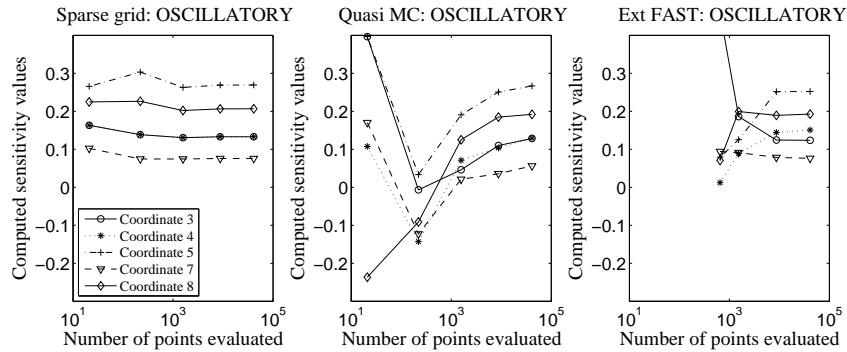


Figure 3: Selected main effect values for the function OSCILLATORY. Sensitivity coefficients for the indicated coordinate dimensions were calculated using each of the sparse grid method, quasi-MC using the Sobol' sequence, and extended FAST. While each method converges to correct values, the sparse grid method displays more robust ordering of sensitivities for a small number of function evaluations.

coefficients  $S_j$  for  $F_1$  using the first 3 methods above. Here we see that not only is the error smallest for the sparse grid method, but also the ordering of the values is consistent, even for a small number of functions evaluations. The convergence results for interaction effects and for the function  $F_2$  were very similar and are not shown here.

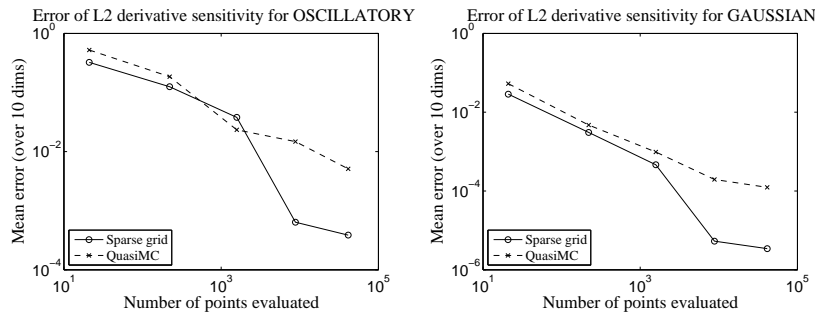


Figure 4: Error in  $L^2$  derivative coefficients for the functions OSCILLATORY and GAUSSIAN. The sensitivity coefficients for each dimension were calculated using the sparse grid method, quasi-MC, and analytically. The errors shown are the mean over the 10 coordinate dimensions.

### 3.2. Derivative-based sensitivity coefficients

We estimated the  $L^2$  derivative sensitivity coefficients  $\nu_i$  using 3 different methods: quasi-MC with a finite difference approximation to the derivatives, sparse grid/gPC, and analytically using Mathematica. Again using the analytic values as reference, we computed the mean error taken over the 10 dimensions. As seen in Figure 4, the quasi-MC and sparse grid/gPC methods were very comparable for a relatively small number of function evaluations, with somewhat better accuracy in the sparse grid method for a larger number of evaluations. The fact that these 2 methods are comparable for this problem is consistent with the result mentioned earlier that quasi-MC converges faster for  $\nu_i$  than for the Sobol' sensitivity coefficients, together with the fact that the error in the derivative of an interpolating function is typically larger than the error in the function itself.

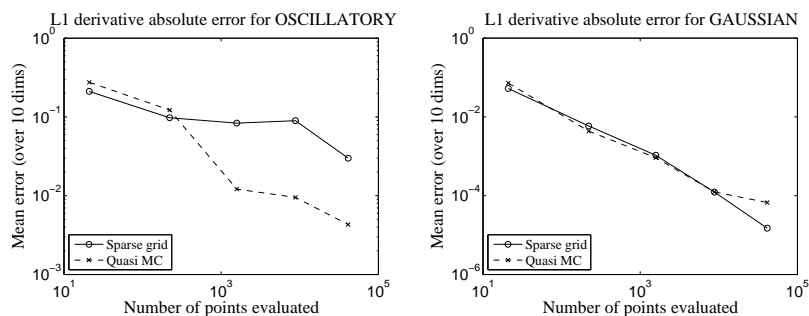


Figure 5: Error in  $L^1$  derivative coefficients for the functions OSCILLATORY and GAUSSIAN. The sensitivity coefficients for each dimension were calculated using the sparse grid method, quasi-MC, and quasi-MC with  $6 \times 10^5$  evaluations as reference. The errors shown are the mean over the 10 coordinate dimensions.

For the  $L^1$  derivative sensitivity coefficients  $\mu_i$ , no analytic solution is available. Hence we estimated these coefficients using the sparse grid/gPC method and using quasi-MC. To provide an estimate of the error, we also estimated these coefficients using quasi-MC with more than 600,000 function evaluations and used the resulting values as reference values. As seen in Figure 5, in this case quasi-MC did as well as or better than sparse grid/gPC. This is consistent with the fact that the sparse grid method has much better convergence for smooth functions than functions with absolute values, such as found in the calculation of  $\mu_i$ . Nevertheless, sparse grid/gPC is reasonably competitive with quasi-MC. Additionally, Figures 6 and 7 show selected

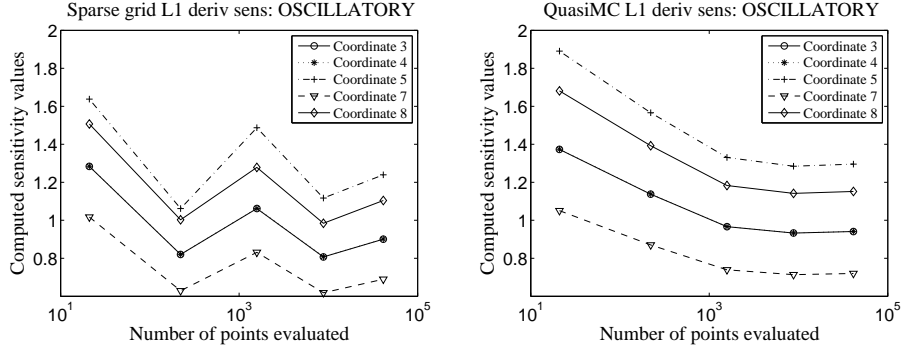


Figure 6: Selected values of the L1 derivative coefficients for the function OSCILLATORY. The sensitivity coefficients for the indicated dimension were calculated using the sparse grid method and quasi-MC. Here quasi-MC displays a faster, monotone convergence compared to the sparse grid method.

values of  $\mu_i$  for these two functions and two methods. Here we see as in the case of the variance based coefficients that even for a small number of function evaluations and relatively large error, the estimates provided by sparse grid/gPC have the correct ordering.

### 3.3. Genetic toggle

A more realistic example is given by the dynamics of a genetic toggle switch, which was developed in [27] and considered numerically in [28] and [29]. Biologically, the dynamics are determined by two mutually inhibitory promoters. Each promoter transcribes a repressor for the other, and either repressor may be induced by an external chemical signal. Following [27], this behavior is modeled as a differential-algebraic equation

$$\begin{aligned} \dot{u} &= \frac{\alpha_1}{1 + v^\beta} - u \\ \dot{v} &= \frac{\alpha_2}{1 + w^\gamma} - v \\ w &= \frac{u}{(1 + [\text{IPTG}]/K)^\eta}. \end{aligned}$$

Here  $u$  and  $v$  are the concentrations of the two repressors,  $\alpha_1$  and  $\alpha_2$  are the effective rates of synthesis of the repressors, and  $\gamma$  and  $\beta$  describe the cooperativity of repression of the two promoters. IPTG is the chemical compound that induces the switch, while  $K$  and  $\eta$  describe the binding of IPTG

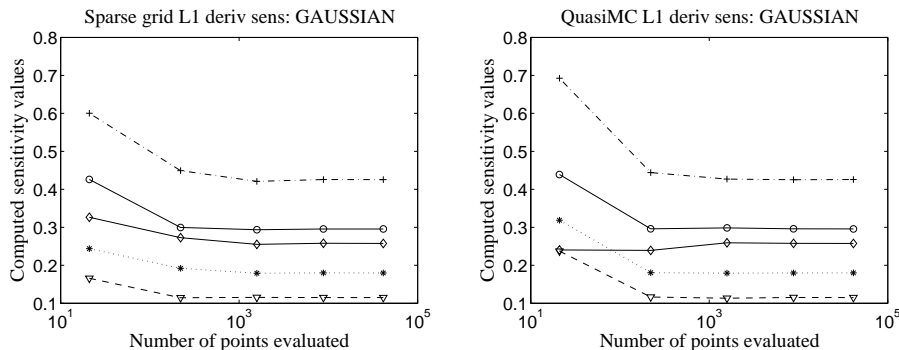


Figure 7: Selected values of the L1 derivative coefficients for the function GAUSSIAN. The sensitivity coefficients for the indicated dimension were calculated using the sparse grid method and quasi-MC. Here quasi-MC and the sparse grid method appear to converge at roughly the same rate.

with the first repressor. The quantity of interest is the steady-state value of  $v$ , which is near 0 for low values of [IPTG] and in the range of 12 to 18 for high values of [IPTG]. The concentration at which the switch takes place depends on the values of the 6 parameters  $(\alpha_1, \alpha_2, \beta, \gamma, \eta, K)$ . Following [29], we assume uniform, independent distributions for each parameter, centered at the nominal values given by [27]. That is, the bounds for parameter  $i$  are given by  $\bar{\theta}_i(1 \pm \zeta_i)$ , with

$$\begin{aligned} \bar{\theta} &= (156.25, 15.6, 2.5, 1, 2.0015, 2.9618 \times 10^{-5}), \\ \zeta &= (0.20, 0.15, 0.15, 0.15, 0.30, 0.20). \end{aligned}$$

We take [IPTG] =  $5.012 \times 10^{-5}$ , which is on the 'high' side, but not far from the switch point for the nominal values. As seen in Figure 8, the sparse grid and quasi-MC methods produce nearly the same estimates for the main effect coefficients. Interestingly, the sparse grid method appears not to have converged with the given number of points. This may be explained by examining Figure 9, which shows the function values obtained at the final level for each of the two methods. The sparse grid method samples more heavily from the boundaries than does quasi-MC; in this case there is a small region near the boundary in parameter space for which this value of [IPTG] is on the 'low' side. Two such points are found by the sparse grid method (shown by the isolated dot near 0 in Figure 9), and in general, this method produced many more samples heading down this 'cliff' (values near

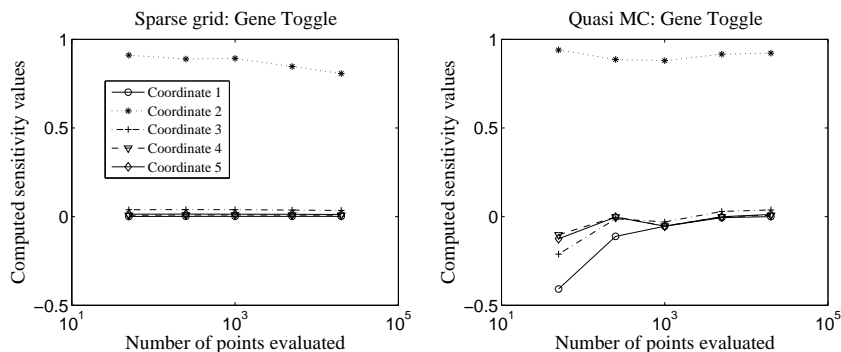


Figure 8: Selected main effect values for the genetic toggle. Sensitivity coefficients for the indicated coordinate dimensions were calculated using the sparse grid method and quasi-MC using the Sobol’ sequence.

8) than did quasi-MC.

We note here that when [IPTG] is set at or very near the switch point for the nominal values, then the sparse grid method does not perform well. In this case, the output function is nearly discontinuous with low values on one side of a co-dimension submanifold of parameter space and high values on the other side. The sparse grid approximation to such a function is not accurate because such a function cannot be well-approximated by low-degree polynomials. Moreover, an approximation by high degree polynomials will have high-frequency oscillations, making estimates of the derivative unreliable. On the other hand, quasi-MC estimations of  $\mu_i$  and  $\nu_i$  are also not likely to be reliable in this setting, since most samples will miss the small region of parameter space in which derivatives are large, and finite difference approximations of derivatives are likely to have large errors. In the end, no method applies in all settings, and agreement of results from multiple methods is perhaps the most reliable way to be confident that calculated values are close to the true values.

### 3.4. Homotopy optimization

Finally, we note that the gPC representation derived from the sparse grid interpolating polynomial has many uses beyond the estimation of sensitivity coefficients. In some sense, the interpolating polynomial may be used as a surrogate for the original function. As one example, if  $P$  is a polynomial that closely approximates  $f$ , then by finding the minima and maxima of  $P$ , we may

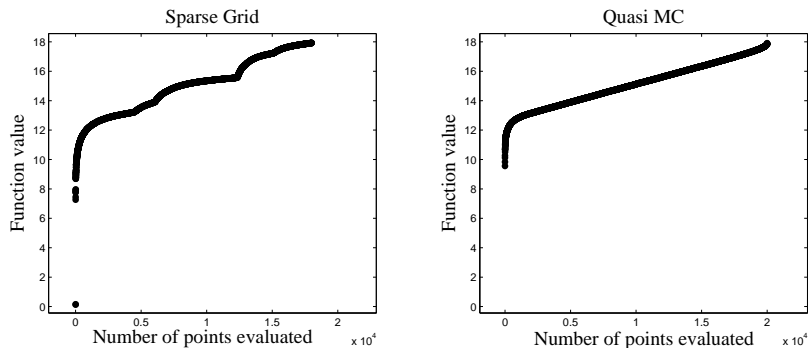


Figure 9: Sorted function values obtained for each method. Note the value near 0 obtained by the sparse grid method.

approximate the minima and maxima of  $f$ . Of course, the approximation of  $f$  by orthogonal polynomials often leads to spurious local maxima and minima in the same way that approximation of a function in one variable by a truncated Fourier series leads to high frequency artifacts. Even so, the local minima and maxima of  $f$  will be approximated by some local minima and maxima of  $P$ . Hence with no prior knowledge of the function  $f$ , we may evaluate  $f$  on a sparse grid, convert to gPC representation to obtain a polynomial  $P$ , calculate  $\nabla P$ , and find the roots of  $\nabla P = 0$  to determine candidates for the local minima and maxima of  $f$ . The package HOM4PS-2.0 [3] provides an efficient homotopy method for solving  $\nabla P = 0$ .

As an illustration of this method, we used a test function in 2 dimensions defined by selecting 3 points,  $x_1, x_2, x_3$  at random in  $H^2$  and setting

$$f(x) = \sum_{j=1}^3 \exp\left(\frac{-1}{30\|Ax - x_j\|^2 + 0.15}\right),$$

where  $A$  is anisotropic scaling followed by rotation. We evaluated this function on a sparse grid with 65 points, converted to gPC form, took the gradient, and used HOM4PS-2.0 to determine the critical points. As seen in Figure 10, the interpolating function not only captures the broad outline of the original function, but also produces a very close approximation to the 3 local minima of the original function.

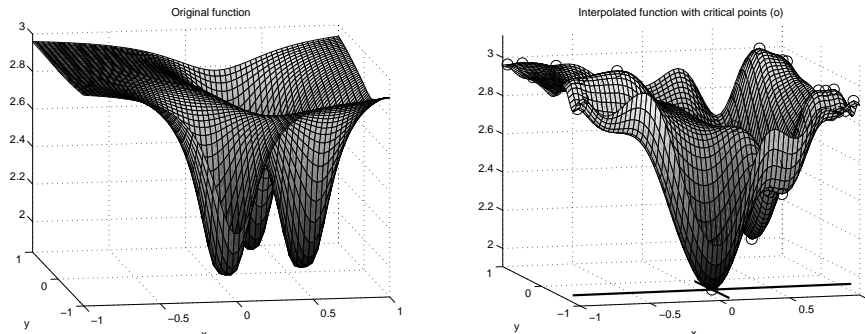


Figure 10: Surface plot of the test function indicated in the text. Left: The original function. Right: The sparse grid interpolating polynomial based on 65 function evaluations. The circles indicate critical points, and the dark lines at the bottom indicate the principal components for the Hessian computed at the global minimum.

#### 4. Conclusion

We have demonstrated that sparse grid interpolation followed by conversion to a gPC representation provides an efficient and accurate method for estimating variance-based sensitivity coefficients (including main effect, total effect, and interaction effect coefficients) and derivative-based sensitivity coefficients (including  $L^1$  and  $L^2$  derivative sensitivity coefficients) for the case of a smooth function. This method provides estimates for all of these coefficients based on function evaluations at a specified set of sparse grid points. Moreover, these estimates have known, good convergence rates that are relatively insensitive to the number of dimensions. We showed numerically that this method gives good estimates even with a relatively small number of points and that it converges faster than quasi-MC and Extended FAST when computing variance-based coefficients. The method given here also converges at a rate comparable to quasi-MC for the derivative-based sensitivity coefficients. Finally, we showed that the gPC representation may be combined with homotopy root finding methods to identify critical points and hence all local maxima and minima of the interpolating polynomial. This provides a useful method for estimating the minima and maxima of the original function. Given the wide range of applications from a single set of function evaluations, the combination of sparse grid interpolation and gPC representation provides a powerful method for exploring the behavior of a smooth function.

## 5. Acknowledgements

I am grateful to Sergei Kucherenko for his assistance in obtaining the Sobol' quasi-MC sequence generator and for helpful discussions on quasi-MC methods. I am also grateful to the organizers of SAMO 2010 for the opportunity to present this research at the SAMO 2010 conference and to the referees for many helpful suggestions and references.

## References

- [1] Sobol' I. Sensitivity estimates for nonlinear mathematical models. *Math Modeling & Comp Experiment* 1993;1:407–14. Translation of Sensitivity estimates for nonlinear mathematical models. *Matematicheskoe Modelirovanie* 1990;2:112–118 (in Russian).
- [2] Sobol' IM, Kucherenko S. Derivative based global sensitivity measures and their link with global sensitivity indices. *Math Comput Simul* 2009;79(10):3009–17.
- [3] Lee T, Li T, Tsai C. Hom4ps-2.0: a software package for solving polynomial systems by the polyhedral homotopy continuation method. *Computing* 2008;83:109–33.
- [4] Morris MD. Factorial sampling plans for preliminary computational experiments. *Technometrics* 1991;33:161–74.
- [5] Storlie CB, Helton JC. Multiple predictor smoothing methods for sensitivity analysis: Description of techniques. *Reliability Engineering & System Safety* 2008;93(1):28 – 54.
- [6] Storlie CB, Swiler LP, Helton JC, Sallaberry CJ. Implementation and evaluation of nonparametric regression procedures for sensitivity analysis of computationally demanding models. *Reliability Engineering & System Safety* 2009;94:1735–63.
- [7] Borgonovo E. A new uncertainty importance measure. *Reliability Engineering & System Safety* 2007;92(6):771 –84.
- [8] Borgonovo E. Measuring uncertainty importance: Investigation and comparison of alternative approaches. *Risk Analysis* 2006;26(5):1349–61.

- [9] Saltelli A. Global sensitivity analysis: the primer. John Wiley; 2008. ISBN 9780470059975.
- [10] Kleijnen JP. Kriging metamodeling in simulation: A review. *European Journal of Operational Research* 2009;192(3):707–16.
- [11] Sudret B. Global sensitivity analysis using polynomial chaos expansions. *Reliability Engineering & System Safety* 2008;93(7):964–79. Bayesian Networks in Dependability.
- [12] Crestaux T, Maitre OL, Martinez JM. Polynomial chaos expansion for sensitivity analysis. *Reliability Engineering & System Safety* 2009;94(7):1161–72. Special Issue on Sensitivity Analysis.
- [13] Xiu D, Hesthaven JS. High-order collocation methods for differential equations with random inputs. *SIAM Journal on Scientific Computing* 2005;27(3):1118–39.
- [14] Barthelmann V, Novak E, Ritter K. High dimensional polynomial interpolation on sparse grids. *Adv Comput Math* 2000;12(4):273–88.
- [15] Smolyak S. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Soviet Math Dokl* 1963;4:240–3.
- [16] Buzzard G. Efficient basis change and regularization for sparse-grid interpolating polynomials; 2010. In revision.
- [17] Li G, Rabitz H, Hu J, Chen Z, Ju Y. Regularized random-sampling high dimensional model representation (rs-hdmr). *Journal of Mathematical Chemistry* 2008;43:1207–32.
- [18] Xiu D, Karniadakis GE. The wiener–askey polynomial chaos for stochastic differential equations. *SIAM J Sci Comput* 2002;24(2):619–44.
- [19] Kucherenko S, Rodriguez-Fernandez M, Pantelides C, Shah N. Monte carlo evaluation of derivative-based global sensitivity measures. *Reliability Engineering & System Safety* 2009;94(7):1135–48. Special Issue on Sensitivity Analysis.
- [20] Saltelli A, Tarantola S, Chan KPS. A quantitative model-independent method for global sensitivity analysis of model output. *Technometrics* 1999;41(1):39–56.

- [21] Buzzard G, Xiu D. Variance-based global sensitivity analysis via sparse grid interpolation and cubature. *Comm Comp Phys* 2011;9:542–67.
- [22] Campolongo F, Cariboni J, Saltelli A. An effective screening design for sensitivity analysis of large models. *Environmental Modelling & Software* 2007;22(10):1509–18.
- [23] Genz A. A package for testing multiple integration subroutines. In: Keast P, Fairweather G, editors. *Numerical Integration : recent developments, software, and applications*. Kluwer, Dordrecht; 1987, p. 337–40.
- [24] Klimke A, Wohlmuth B. Algorithm 847: Spinterp: piecewise multilinear hierarchical sparse grid interpolation in matlab. *ACM Trans Math Softw* 2005;31(4):561–79.
- [25] British-russian offshore development agency (broda) website. 2010. Accessed August 16, 2010; URL <http://www.broda.co.uk/index.html>.
- [26] Simlab website; 2009. Accessed July 2009; URL <http://simlab.jrc.ec.europa.eu/>.
- [27] Gardner TS, Cantor CR, Collins JJ. Construction of a genetic toggle switch in escherichia coli. *Nature* 2000;403:339–42.
- [28] Xiu D. Efficient collocational approach for parametric uncertainty analysis. *Comm Comp Phys* 2007;2:293–309.
- [29] Marzouk Y, Xiu D. A stochastic collocation approach to bayesian inference in inverse problems. *Comm Comp Phys* 2009;6:826–47.