# Substituent Ordering and Interpolation in Molecular Library Optimization

**Neil Shenvi, John M. Geremia, and Herschel Rabitz***

*Department of Chemistry, Princeton University, Princeton, New Jersey 08544*

*Received: August 22, 2002*

The substituent ordering problem in molecular libraries refers to identifying a rational ordering for molecular moieties such that coarse sampling and interpolation over the full space of possible library molecules may be efficiently performed. A practical solution to the ordering problem is proposed on the bases of (a) coarse sampling of the molecular substituents, (b) radial basis function interpolation over the full space, and (c) the use of genetic algorithms to find rational moiety orderings. The procedure is shown to be extremely effective for a variety of simulated libraries. This algorithm is also used to reorder and predict the glass transition temperature $T_g$ for a combinatorial polymer library.

## I. Introduction

Recent advances in combinatorial synthesis have made possible the creation of libraries of thousands of structurally diverse compounds.[1,2] These libraries are often based on a single molecular scaffold with several substituent sites.[3−5] In polymer libraries, substituents may be attached to sites on the repeat unit or incorporated into the polymer backbone.[6,7] Each compound in the library is characterized by a unique combination of substituent groups. If there are $S$ different substituent groups at each scaffold site and $N$ scaffold sites, then there are a total of $S^N$ different compounds that can be synthesized from the scaffold. The scaffold may also be considered as a variable for sampling, but in the present paper the scaffold will be taken as fixed.

Once a compound library is synthesized, it is usually tested for some set of desirable properties. In the case of drug synthesis, these properties may be biological activity, toxicity, etc., as measured by a screening assay. In the case of polymer synthesis, these properties may be the air−water contact angle, the glass transition temperature, rheological measures, etc.[6,7] The problem of property optimization involves finding the set of substituents that optimizes the properties of interest. This task becomes more complex if multiple, and possibly competing, properties must be simultaneously optimized. If the library synthesis is complete (i.e., all of the $S^N$ possible compounds are synthesized and their properties measured), then the best compound can be determined directly from the screening data.

In practice, the screening data is often incomplete. The synthesis and assaying of combinatorial libraries can be hindered by a number of factors including low product yields, low quality assays, and incomplete sampling. Furthermore, split-and-mix techniques which in theory allow for the creation of complete libraries cannot be easily applied to certain materials, such as polymers, since large amounts of the compounds are often needed for testing. Multiple parallel syntheses can be performed on a larger scale than split-and-mix syntheses, but require resources which grow linearly with the number of compounds synthesized. As a result, the parallel synthesis of large libraries can be prohibitively expensive. In most cases, the number of compounds synthesized and tested is a small fraction of the possible compounds.

Incomplete library data makes the problem of property optimization more difficult, especially in the common case where a balance between multiple properties must be sought. Let $M$ be the number of compounds synthesized and tested out of the $S^N$ possible compounds, where typically $M \ll S^N$. In this context, property optimization may be viewed as a sparse interpolation problem.[8,9] Let $S_1, S_2, ..., S_N$ be the number of different substituent groups available at each of the $N$ scaffold sites. Then we define the variables $X_1, X_2, ..., X_N$ to represent the type of substituents at scaffold sites 1, 2, ..., $N$ such that each variable $X_i$ can take on $S_i$ integer values on the range $[1,S_i]$. Hence, every compound in the library can be represented as an $N$-dimensional vector $\mathbf{X}$ of the discrete variables, $X_1, X_2, ..., X_N$. The measurement $y$ associated with each compound $\mathbf{X}$ can be expressed as

$$y = g(\mathbf{X}) \tag{1}$$

The problem of optimizing the property described by $y$ is equivalent to optimizing the function $g(\mathbf{X})$ over a discrete variable space. The perspective taken here does not call for the use of any molecular descriptors or detailed knowledge of the assay (e.g., the receptor).

Many algorithms for high-dimensional function optimization are available, but molecular library optimization is not readily amenable to traditional optimization methods. All nonlinear optimization routines, such as gradient descent, simulated annealing and genetic algorithms, rely on repeated evaluation of the model function, $g(\mathbf{X})$, to find an optimal solution.[10,11] However, in the case of library optimization, extensive function sampling is prohibitively time-consuming. To determine the value of $g(\mathbf{X})$ for an unknown compound, this compound must be synthesized and then tested.

Extensive function evaluation can be circumvented by using a suitable regression technique to fit an analytic function $f(\mathbf{X})$ to the model function $g(\mathbf{X})$ at the $M$ sampled points. The analytic function $f(\mathbf{X})$ can then be used to interpolate the sampled points in order to predict the properties of the unsampled compounds. However, interpolation is only possible if the property function $g(\mathbf{X})$ is relatively smooth between sampled points. In general, the smoothness of the property function is dependent on the appropriate assignment (i.e., ordering) of the substituents to the discrete values 1, 2, ..., $S_i$ for every lattice site, $i = 1, 2, ..., N$. If an incorrect assignment is made, then the property function may be highly nonsmooth, and function interpolation will be inaccurate.[8,9] Property optimization can be broken up into three

10.1021/jp021932n CCC: $25.00 
Published on Web 03/05/2003

steps: First, a method must be devised for treating the ordering problem of assigning integer values to the substituents over the range [1, $S_i$] for all $i$. Multiple properties, $g_p(\mathbf{X})$, $p = 1, 2, ...$, may also be considered, with each property function likely having its own substituent ordering. Second, an accurate sparse data interpolation technique must be employed to represent the data points as an analytic function, $f(\mathbf{X})$. Third, an optimal combination of substituents must be found by optimizing $f(\mathbf{X})$ over all the possible compounds. Though the second and third steps are straightforward, they cannot be executed without dealing with the ordering problem. Until now, no satisfactory methods were expressed for finding an optimal ordering of the substituents.[8,9] In this study, an efficient method for solving the substituent ordering problem is presented along with simulations of its use with steps two and three. This method facilitates accurate property prediction for compound libraries, which is of vital importance in property optimization algorithms. In actual practice, a full algorithm could best operate in a closed loop fashion starting with as few sample points, $\tilde{\mathbf{X}}_1, \tilde{\mathbf{X}}_2, ..., \tilde{\mathbf{X}}_M$, as possible, with iterative refinements to achieve the desired resolution of $g(\mathbf{X})$. Guidance from analogous studies or modeling with standard descriptors could also be used to provide initial hints at possible acceptable substituents and their orderings.

Section II defines the central ordering assignment problem and describes the algorithm for its practical treatment. Section III demonstrates the performance of the algorithm for several test cases, and conclusions are presented in section IV.

## II. Algorithm

Let $R_{i1}, R_{i2}, ..., R_{iS_i}$ be the $S_i$ substituents that can be attached to scaffold position $i$. Then $X_i$ can take on the integer values 1, 2, ..., $S_i$. We define an ordering $Z_i = [Z_{i1}, Z_{i2}, ..., Z_{iS_i}]$ of the substituents $R_{i1}, R_{i2}, ..., R_{iS_i}$ as any permutation of the integers 1, 2, ..., $S_i$. Each scaffold site $i = 1, 2, ..., N$ will have its own ordering $Z_i$. We can uniquely assign integer values to the substituents $R_{i1}$, $R_{i2}$, and $R_{iS_i}$ on the basis of this ordering such that $R_{i1}$ corresponds to $X_i = Z_{i1}$; $R_{i2}$ corresponds to $X_i = Z_{i2}$, ...; and $R_{iS_i}$ corresponds to $X_i = Z_{iS_i}$. We define a total ordering, $\mathbf{Z} = [\mathbf{Z}_1, \mathbf{Z}_2, ..., \mathbf{Z}_N]$, to be a set of $N$ orderings which contains an ordering for each axis. Because each of the lattice site substituents can be ordered independently, there are $S_1! \times S_2! \times ... \times S_N!$ possible total orderings of the substituents.

An ordering possibly may be imposed on the substituents based on physical intuition, prior analogous property behavior, or modeling. However, unless the dependence of the property $g(\mathbf{X})$ on $\mathbf{X}$ is reliably known, a total ordering based on such simple arguments can be misleading. Ideally, we would like to find an ordering which is optimal based only on knowledge of $g(\mathbf{X})$ at a set of $M$ sampled points, with an aim toward further refinement using additional sampling if better resolution of $g(\mathbf{X})$ is required. An optimal total ordering of the variable axes $X_1, X_2, ..., X_N$ is defined as the set $\mathbf{Z} = [\mathbf{Z}_1, \mathbf{Z}_2, ..., \mathbf{Z}_N]$ such that $g(\mathbf{X})$ is a smooth function over $\mathbf{X}$. The assumption of smooth behavior over $\mathbf{X}$ is plausible on physical grounds, as any property $g(\mathbf{X})$ should draw on the substituent characteristics in a systematic fashion. However, there will generally be multiple total orderings that will serve nearly equivalently, since (a) smoothness can be quantitatively defined in different ways and (b) the smoothness property is only needed to make the subsequent interpolation reliable. The physical/chemical origin of the systematic smooth behavior may remain buried when utilizing the algorithm, as its existence is all that is required for success of the ordering algorithm. It is possible that some physical systems could exhibit nonsmooth behavior. However, in this case, traditional interpolation methods would also fail without specific a priori knowledge of the property function. Turning these arguments around, the patterns of observed ordering regularity (or even their deviations) should be able to provide the basis for a deeper physical understanding of molecular properties.

Other knowledge or assumptions regarding the structure of the property function can also be made to aid in property prediction. An arbitrary property function $g(\mathbf{X})$ can be exactly expressed as

$$g(\mathbf{X}) = f_0 + \sum_i^N f_i(X_i) + E(\mathbf{X}) \tag{2}$$

where $f_0$ is a constant, $f_i(X_i)$ describes the independent action of the substituent at scaffold site $i$ on the observed property, and $E(\mathbf{X})$ is a term which describes the dependence of the observed property on the cooperative action of two or more substituent sites. If the substituents on each scaffold site contributes independently to the observed property, then $E(\mathbf{X}) = 0$. Such a model function is said to be separable. This type of analysis including consideration of higher-order cooperation, is conveniently expressed in terms of a high-dimensional model representation (HDMR) expansion.[12−14]

There are a number of alternative ways of exploiting the HDMR algorithm for molecular property interpolation.[8,9] In the simplest approach, the constant $f_0$ and the functions $f_1(X_1)$, $f_2(X_2)$, ..., $f_N(X_N)$ can be evaluated with respect to a representative compound, $\bar{\mathbf{X}}$, which is known as the cut-center,

$$f_0 \equiv g(\bar{X}_1, \bar{\cdots}, \bar{X}_N) \tag{3}$$

$$f_i(X_i) \equiv g(\bar{X}_1, \bar{\cdots}, X_i, \bar{\cdots}, \bar{X}_N) - f_0 \tag{4}$$

where "$\bar{\cdots}$" denotes all arguments set to their cut-center values. The constant $f_0$ is evaluated by sampling the property function at the cut-center. Each of the functions $f_i(X_i)$ is evaluated by sampling the model function along the $X_i$-axis through the cut-center. Thus, a representation for $g(\mathbf{X})$, initially neglecting cooperativity effects in eq 2, can be obtained by structured sampling of only $\sum_i^N (S_i - 1) + 1$ compounds. If $g(\mathbf{X})$ is separable, then this representation will be exact. Furthermore, the interpolation of a separable function is independent of the substituent ordering and the choice of cut-centers.

The assumption of separability is not generally valid, as the observed property may depend on the cooperative action of multiple substituents. If a property function is nonseparable, then the accuracy of interpolation will be dependent on both the substituent ordering and the choice of cut-centers. However, the cooperative action of substituents is often small relative to the independent action described by the functions, $f_1(X_1)$, $f_2(X_2)$, ..., $f_N(X_N)$. Thus, structured cut-center sampling can yield valuable information regarding the behavior of the model function. Furthermore, the effects of cooperativity between substituents can be included by using multiple cut-centers, additional random sampling of the library compounds, and the selected inclusion of explicit cooperative terms residing in $E(\mathbf{X})$ in eq 2. These alternatives will not be fully explored in this work but in section III, we will compare the performance of the ordering algorithm on the bases of (a) randomly sampled compound data and (b) data which has been sampled along the substituent axes in reference to a cut-center and at additional random points on the domain.

Regardless of how the $M$ samples are initially chosen, given some total ordering, $\mathbf{Z} = [\mathbf{Z}_1, \mathbf{Z}_2, ..., \mathbf{Z}_N]$, of all $N$ variable axes, we define the smoothness $J$ of the total ordering as

$$J(\mathbf{Z}) = \frac{1}{R}\sum_{i}^{R} ||\nabla g^{Z}(\mathbf{Z}_i)||^2 \quad (5)$$

where $\mathbf{X}_1$, $\mathbf{X}_2$, ..., $\mathbf{X}_R$ can be any sufficiently large set of the library compounds and $g^{\mathbf{Z}}(\mathbf{X})$ is the property represented with total ordering $\mathbf{Z}$. The gradient $\nabla$ is taken with respect to the $N$ variables constituting the $\mathbf{X}$-space. Smoothness could be defined in terms of higher order derivatives as well, but here we chose the simplest case in eq 5. The norm $||...||$ in eq 5 is also subject to definition of how and/or where it is evaluated in the full domain of the $\mathbf{X}$-space. We will approximate $g^{Z}(\mathbf{X}_i)$ by an analytic function $f(\mathbf{X})$, which will be determined from $g(\mathbf{X})$ at the $M$ data points, $\tilde{\mathbf{X}}_1$, $\tilde{\mathbf{X}}_2$, ..., $\tilde{\mathbf{X}}_M$. Generally, a regression technique will be the easiest way of determining $f(\mathbf{X})$, since $\tilde{\mathbf{X}}_1$, $\tilde{\mathbf{X}}_2$, ..., $\tilde{\mathbf{X}}_M$ will be scattered data points. Any suitable basis could be used for the regression, provided that it exhibits good high-dimensional sparse data interpolation properties. Here we chose the $M$ radial basis functions $\phi_1$, $\phi_2$, ..., $\phi_M$, where

$$\phi_k(\mathbf{X}) = (|\tilde{\mathbf{X}}_k - \mathbf{X}|^2 + 1)^{1/2} \quad (6)$$

Thus,

$$f(\mathbf{X}) = \sum_{k}^{M} c_k(|\tilde{\mathbf{X}}_k - \mathbf{X}|^2 + 1)^{1/2} \quad (7)$$

where the vector of coefficients $\mathbf{c}$ was determined by fitting $f(\mathbf{X})$ to get $g(\mathbf{X})$ through linear regression. The gradient of the regression function $f(\mathbf{X})$ can be calculated analytically,

$$\nabla_j f(\mathbf{X}) \equiv \frac{\partial}{\partial X_j} f(\mathbf{X}) = \sum_{k}^{M} -c_k(|\tilde{\mathbf{X}}_k - \mathbf{X}|^2 + 1)^{1/2}(\tilde{X}_{kj} - X_j) \quad (8)$$

Combining eq 5 and eq 8,

$$J(\mathbf{Z}) \approx \frac{1}{R}\sum_{i}^{R}\sum_{j}^{N}(\sum_{k}^{M} - c_k(|\tilde{\mathbf{X}}_k - \mathbf{X}|^2 + 1)^{1/2}(\tilde{X}_{kj} - X_j))^2 \quad (9)$$

Thus, the smoothness $J(\mathbf{Z})$ of any total ordering $\mathbf{Z}$ is approximated by eq 9. Equation 9 can be evaluated over any large set of $R$ compounds. In this paper, we will evaluate eq 9 over every compound in the substituent space, i.e., $R = S_1 \times S_2 \times ... \times S_N$ to ensure that $f(\mathbf{X})$ is smooth over the entire substituent domain. It should be stressed that this process does not involve sampling (i.e., synthesizing) and measuring the properties of every compound in the substituent space, but only evaluating the gradient of the analytic approximation $f(\mathbf{X})$ at every point in the substituent space.

To find an optimal total ordering, $J(\mathbf{Z})$ was minimized over the set $\mathbf{Z}$ of all possible total orderings

$$\min_{\mathbf{Z} \in Z} J(\mathbf{Z}) \quad (10)$$

The minimization in eq 10 may have several extrema $\mathbf{Z}^*$ of possibly good quality. The algorithm need only find at least one good solution. The minimization of $J(\mathbf{Z})$ could be accomplished by a brute force algorithm that samples every possible total ordering. However, the number of possible total orderings grows as $\prod_i^N S_i!$. Even for a relatively small library with two scaffold sites and 10 substituents per site, there are over $1.3 \times 10^{13}$ possible permutations. Thus, the complexity of the problem makes a brute force implementation unfeasible. The number of possible substituent orderings $\prod_i^N S_i!$ grows faster than the number of compounds in the library $\prod_i^N S_i$. However, the search over the permutations of orderings can be done off-line using a modest number ($M \ll \prod_i^N S_i$) of samples, with the identified optimal ordering $\mathbf{Z}^*$ to be subsequently utilized for interpolation over the entire library of possible compounds in the property optimization stage.

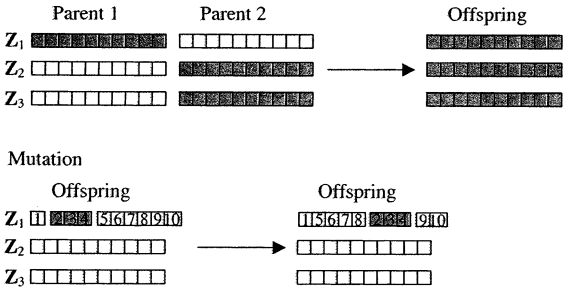


**Figure 1.** Schematic diagram of genetic mutation and crossover for the ordering of a library of compounds with three scaffold sites. Each site has a possible substituent ordering, $\mathbf{Z}_i$. The total ordering is given by $\mathbf{Z} = [\mathbf{Z}_1, \mathbf{Z}_2, \mathbf{Z}_3]$.

Finding a total ordering, $\mathbf{Z}^*$, which minimizes $J(\mathbf{Z})$ can be accomplished efficiently by a genetic algorithm.[11] Genetic algorithms perform function optimization by treating each total ordering as the genome for an individual "organism". For the purposes of the present optimization, the genome $\mathbf{Z}$ consisted of $N$ genes, $\mathbf{Z} = [\mathbf{Z}_1, \mathbf{Z}_2, ..., \mathbf{Z}_N]$. Each gene, $\mathbf{Z}_i$, was the ordering of the substituents at scaffold site $i$. The smoothness of the total ordering $J(\mathbf{Z})$ was used as the objective function. Individuals with lower values of $J(\mathbf{Z})$ were given a greater chance for survival and reproduction. Thus, after a sufficient number of generations, the genetic algorithm aims to find a minimal value of $J(\mathbf{Z})$.

The steady-state genetic algorithm implementation introduced by Goldberg was used to find an optimal total ordering.[11,15] This algorithm proceeds as follows: First, an initial population of 100 individuals was generated with random total orderings. Then the objective function value for each individual in the population was calculated. After each generation, the 10 worst individuals were removed from the population. These individuals were replaced by generating 10 new individuals which were the offspring of the 90 remaining individuals. Individuals with lower objective function scores were more likely to survive and reproduce.

Different total orderings were sampled through genetic mutation and crossover, which occurred during the mating process. Mating was accomplished through sexual selection which favored individuals with lower objective function scores. To generate each offspring, two parents were selected with a probability proportional to the inverse of their objective function scores. If neither crossover nor mutation occurred, then the offspring was a genetic replica of one of the parents. Genetic crossover occurred with a probability of 0.8. If crossover occurred, then the genetic code of the offspring was generated by selecting each gene from either parent with equal probability. Thus, the product of crossover was a single offspring which incorporated genes from each parent. Mutation occurred separately for each gene with a probability of 0.2 and could occur along with crossover. If mutation occurred, then a segment of random length was removed from the mutated gene and reinserted into a random position in the gene. Figure 1 shows

**TABLE 1: Reordering of Substituents for the Linear Model Function in Equation 11: Inversion of Order for $Z_1$ and $Z_2$ Is Irrelevant, since Only Relative Ordering Is Significant**

| | $\mathbf{Z}^{seq}$ | | | | | | | | | | $\mathbf{Z}^*$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\mathbf{Z}_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 9 | 10 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| $\mathbf{Z}_2$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 1 | 2 |
| $\mathbf{Z}_3$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

a schematic illustration of crossover and mutation. The frequent occurrence of crossover and mutation ensured that the genetic algorithm sampled a diverse selection of orderings.

The definitions for mutation and crossover were based on the logical behavior of the objective function $J(\mathbf{Z})$. For instance, it is likely that an individual possessing a gene for the optimal ordering of a single scaffold site will have a low objective function score. Thus, crossover preserves the entire ordering of a single scaffold site (i.e., a gene of one parent) in the genetic code of the offspring (cf., Figure 1). As each scaffold site will likely have a unique chemical/physical role upon the observable, it generally makes no sense to consider crossover between different sites (i.e., different genes). Similarly, low objective function scores may be the result of large segments in a gene which are optimally ordered. Mutation attempts to correct the poorly ordered segments of individual genes through rearrangement.

In the limit of a sufficient number of generations, genetic algorithms will explore the entire substituent ordering space $\mathcal{Z}$. However, such a thorough exploration would be prohibitively time-consuming and generally unnecessary, as any good solution should suffice for molecular property interpolation. The efficiency and accuracy of the genetic ordering algorithm were investigated for several test cases based on a fixed number, $M$, of data samples. The results of these experiments are given in section III. No attempt was made to explore an iterative algorithm that would seek a minimal number of samples for a specified level of smoothness.

## III. Illustrations

Several numerical experiments were performed to test the utility of the ordering algorithm. First, the algorithm was tested by using simulated data generated by a linear, separable property function. The performance of the algorithm was then tested using a more complicated quadratic model function with cooperative coupling. Finally, the algorithm was used to predict the glass transition temperatures of a library of copolymers studied in a previous paper.[7]

For the first example, a library with three substituent sites ($N = 3$) and 10 substituents per site ($S_1 = S_2 = S_3 = 10$) was considered. Thus, the complete library had 1000 compounds. The property function was

$$g^Z(\mathbf{X}) = \mathrm{X}_1 + \mathrm{X}_2 + \mathrm{X}_3 \tag{11}$$

where $\mathbf{Z}$ refers to the total ordering of the substituents. A reference case was chosen as $\mathbf{Z}^{seq}$ with sequential ordering (see Table 1). Each $X_i$ ($i = 1, 2, 3$) took integer values on the range.[1,10] We define

$$J^{seq} \equiv J(\mathbf{Z}^{seq}) \tag{12}$$

to be the smoothness of the sequential total ordering for a given data set as defined by eq 9. A *smooth total ordering* is defined to be any total ordering $\mathbf{Z}^*$ such that $J(\mathbf{Z}^*) \leq J^{seq}$. In general, a finite sample size can effect the distribution of smooth total orderings over the ordering space $\mathcal{Z}$ and $\mathbf{Z}^{seq}$ may not actually be the smoothest ordering as defined by eqs 5 and 10. It is also not guaranteed that there is a unique smooth total ordering for any function; many total orderings may be display good local optimality, as defined in eq 10.

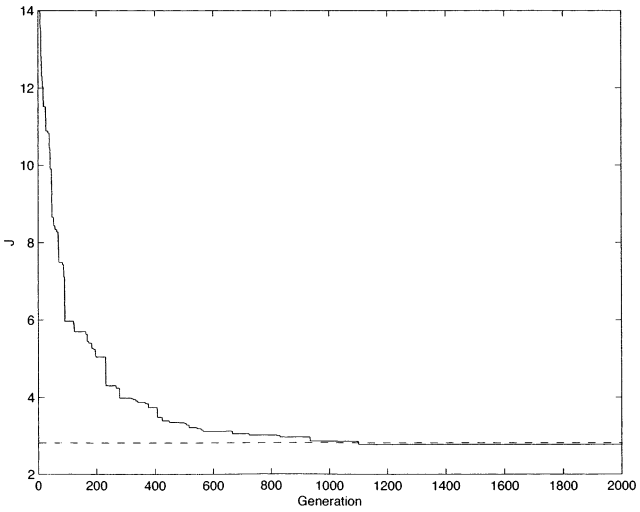


**Figure 2.** Smoothness of the best solution versus genetic algorithm generation for reordering of the linear model function. The dashed line is the smoothness $J^{seq}$ of the sequential ordering.

In this first example, the cut-HDMR approach based on eq 2 would immediately lead to the solution $J(\mathbf{Z}^{seq})$ as the function $g(\mathbf{X})$ in eq 11 is separable. Thus, as a more general test of the reordering procedure, a data set was generated by randomly sampling $M = 100$ of the library compounds, with the constraints that every substituent was used in at least one sampled compound and that no compounds were duplicated. Using this data set, the genetic ordering algorithm was run for 2000 generations. Figure 2 shows a plot of the best solution versus generation. The genetic algorithm converged to a solution, $\mathbf{Z}^*$, with objective function value $J(\mathbf{Z}^*) = 2.78$ after 1100 generations. Since 10 new individuals were created each generation to replace the 10 worst individuals, each generation required the sampling of 10 new total orderings. Thus, the genetic algorithm located an optimal solution $\mathbf{Z}^*$ by sampling $1.11 \times 10^4$ total orderings on the space $\mathcal{Z}$. The objective function value of the sequential total ordering was $J^{seq} = 2.82$. Hence, the genetic algorithm converged to an optimal total ordering. Furthermore, this optimal total ordering recovered the relative ordering of $\mathbf{Z}^{seq}$ for almost all of the substituents (see Table 1). A radial basis function regression using the $M = 100$ sampled points and the optimal ordering $\mathbf{Z}^*$ was used to predict the model function values for the unsampled points. The average RMS error of the regression on the 900 unsampled points was 0.65. Figure 3 shows a plot of the predicted value of the model function versus the actual value of the model function. For comparison, a radial basis function regression using the same $M = 100$ sampled points and a random (i.e., nonsmooth) ordering gave an average RMS error of 5.0. Figure 4 shows the predicted versus the actual value of the model function for the random ordering. Comparing Figure 3 to Figure 4 shows that substantial gains in accuracy were made by determining a smooth ordering for the substituents prior to regression.

The efficiency of the genetic algorithm was then compared to a brute force implementation by sampling the objective function values of $1.11 \times 10^4$ randomly selected total orderings on the ordering space $\mathcal{Z}$. None of these total orderings displayed an objective function value of less than 9. Figure 5 shows a histogram of the smoothness values. The distribution of smoothness values had a mean of $J = 18.6$ and a standard deviation of

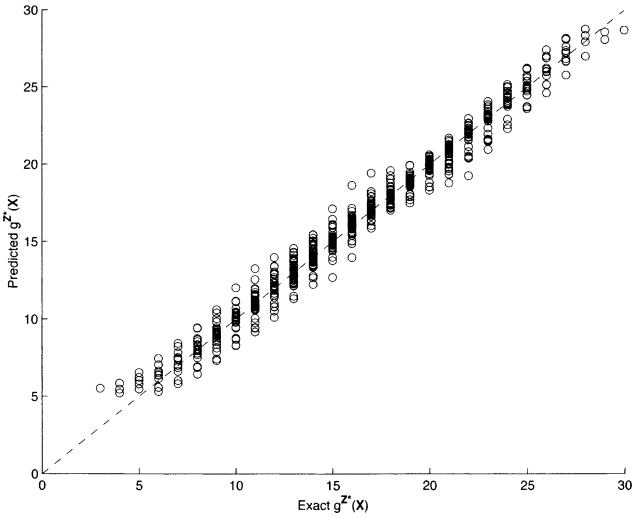


**Figure 3.** Predicted values of the linear model function using the identified smooth ordering **Z*** versus the actual values for the 900 unsampled data points.

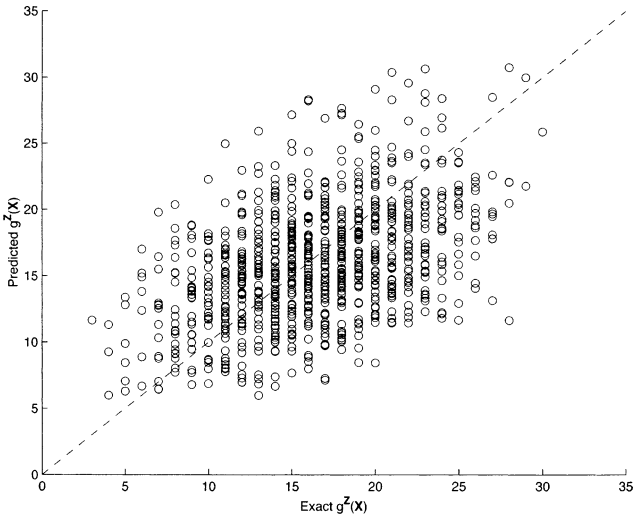


**Figure 4.** Predicted values of the linear model function using a random (i.e., nonsmooth) ordering **Z** versus the actual values for the 900 unsampled compounds. This behavior should be compared to that found in Figure 3 using the identified optimal ordering **Z***.

2.3. If we estimate that this distribution is approximately Gaussian, then a brute force implementation would have required more than $\sim 10^{11}$ samplings to find an optimal total ordering with a smoothness value of less than $J^{seq}$.

The next example tested the algorithm using a nonlinear model function,

$$g^Z(\mathbf{X}) = {X_1}^2 - (X_2 - X_3)^2 \qquad (13)$$

We again considered a scaffold with three substituent sites and 10 substituents per site, giving a complete library of 1000 compounds. Thus, each variable $X_i$ again took integer values on the range [1, 10], as in the first example. The observed values of $M = 100$ randomly selected compounds were used as the data set. The objective function value of the sequential total ordering $\mathbf{Z}^{seq}$ in Table 2 was $J^{seq} = 236$.

The genetic ordering algorithm was run for 1000 generations and was again able to converge to a smooth total ordering despite the increased complexity of the model function. Figure 6 shows a plot of the best solution versus generation. The genetic algorithm converged to a solution **Z*** with objective function

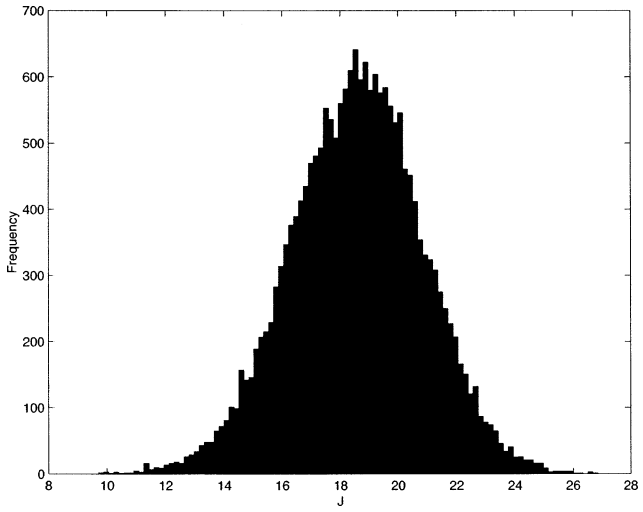


**Figure 5.** Histogram of smoothness values for $1.1 \times 10^4$ randomly sampled orderings of the linear model function. Random sampling of the orderings is shown to be very inefficient compared to using the genetic algorithm in Figure 2.

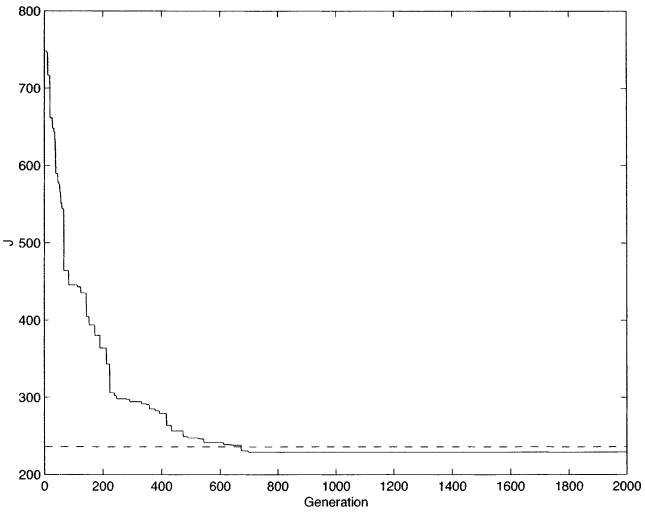


**Figure 6.** Smoothness of best solution versus genetic algorithm generation for the reordering of the quadratic model function. The dashed line is the smoothness $J^{seq}$ of the sequential ordering.

**TABLE 2: Reordering of Substituents for Quadratic Model Function in Equation 13: Inversion of Order for $Z_1$ Is Irrelevant since Only Relative Ordering Is Significant**

| | $\mathbf{Z}^{seq}$ | | | | | | | | | | $\mathbf{Z}^*$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\mathbf{Z}_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 1 | 2 |
| $\mathbf{Z}_2$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 1 | 2 | 3 | 4 | 6 | 5 | 7 | 10 | 9 | 8 |
| $\mathbf{Z}_3$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 2 | 4 | 1 | 3 | 5 | 7 | 6 | 8 | 9 | 10 |

value $J(\mathbf{Z}^*) = 229$ after 699 generations and a total of 7090 objective function evaluations. The optimal total ordering located by the genetic algorithm also displayed good agreement with the sequential total ordering of the model function (see Table 2). A radial basis function regression using the $M = 100$ sampled points and the optimal ordering **Z*** was able to predict the model function values for the 900 unsampled points with an average RMS error of 11.6. Figure 7 shows a plot of the predicted versus the actual value of the model function for the optimal ordering. For comparison, a radial basis function regression using the same $M = 100$ sampled points and a random (i.e., nonsmooth) ordering gave an average root-mean square (rms) error of 33.3. Figure 8 shows the predicted versus the actual value of the model function for the random ordering. Comparing Figure 7

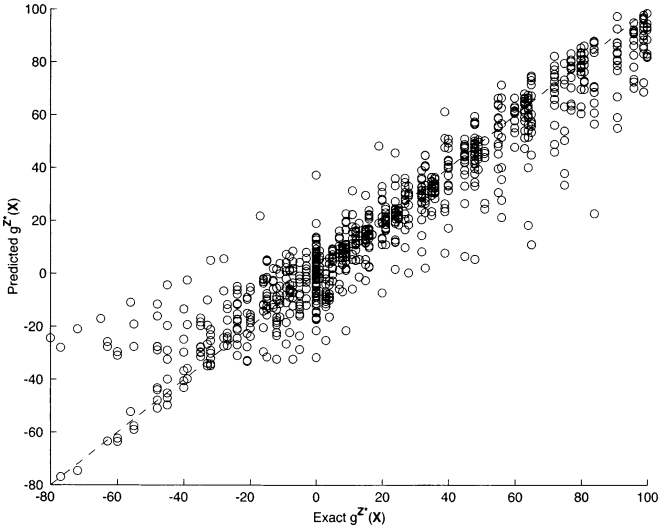


**Figure 7.** Predicted values of the quadratic model function using the smooth ordering **Z*** versus the actual values for the 900 unsampled data points.

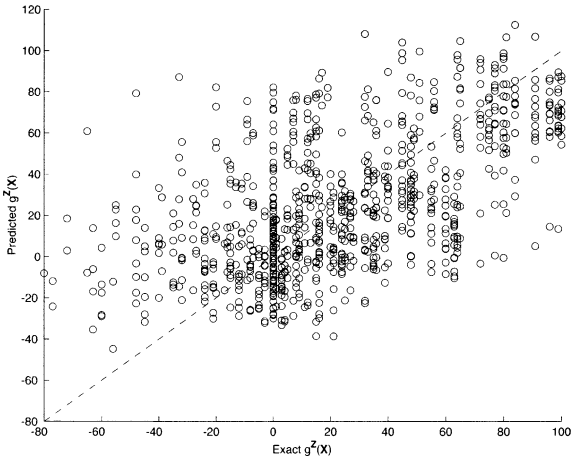


**Figure 8.** Predicted values of the quadratic model function using a random (i.e., nonsmooth) ordering **Z** versus the actual values for the 900 unsampled compounds. This behavior should be compared to that found in Figure 7 using the identified optimal ordering **Z***.

to Figure 8 again shows that significant gains in accuracy were made by determining a smooth ordering for the substituents prior to regression.

Having verified the behavior of the ordering method to find a smooth ordering using simulated data, the algorithm was applied to experimental data from a polymer library. A previous study by Reynolds et al. used Quantitative Structure−Property Relationships (QSPR) to predict the properties of a 112-compound polymer library from a representative 17-polymer data set.[7] The library considered by Reynolds et al. consisted of two substituent sites: a diol substituent and diacid substituent. A total of 14 diols and 8 diacids were used to synthesize a complete library of (14)(8) = 112 compounds. Property prediction was accomplished using two-dimensional topological descriptors to characterize the substituent groups. The glass transition temperature property $T_g$ of the polymers in this library was found to be highly dependent on the chain length of the substituent diacids and diols.[6] Once suitable molecular descriptors were identified, QSPR models were then derived from the experimental $T_g$ values for the representative 17-polymer data set using linear regression. These models were shown to have good agreement with the experimental values of $T_g$.

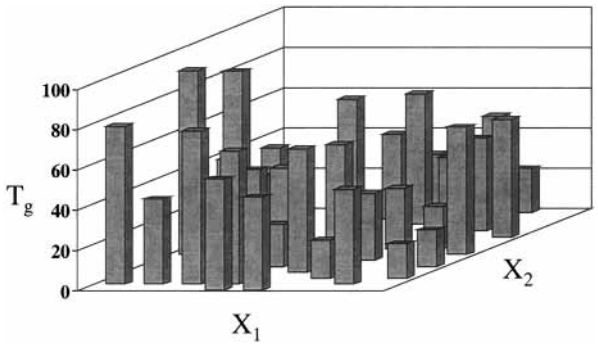


**Figure 9.** Plot the experimental $T_g$ values versus polymer substituents for the original sequential ordering, $\mathbf{Z}^{seq}$. $\mathbf{X}_1$ is the diacid substituent variable. $\mathbf{X}_2$ is the diol substituent variable. Compounds sampled in data set A are shown.

**TABLE 3: Reordering of Substituents for the Experimental $T_g$ Data Using Data Sets A and B: $Z_1$ Is the Ordering of the Diacid Substituent, and $Z_2$ Is the Ordering of the Diol Substituent.**

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | $\mathbf{Z}^A$ | | | | | | | | |
| $\mathbf{Z}_1$ | 2 | 8 | 7 | 6 | 5 | 4 | 3 | 1 | | | | | | |
| $\mathbf{Z}_2$ | 7 | 4 | 2 | 12 | 10 | 11 | 9 | 1 | 6 | 13 | 14 | 8 | 3 | 5 |
| | | | | | | $\mathbf{Z}^B$ | | | | | | | | |
| $\mathbf{Z}_1$ | 2 | 7 | 8 | 6 | 5 | 4 | 3 | 1 | | | | | | |
| $\mathbf{Z}_2$ | 12 | 11 | 4 | 7 | 10 | 2 | 9 | 1 | 6 | 13 | 14 | 8 | 3 | 5 |

Rather than using traditional descriptors as a guide, the present algorithm used only the experimental $T_g$ values of a subset of the library compounds. Thus, no additional information regarding the structure or physical properties of the substituents was incorporated into the prediction algorithm. As with the prior two examples, the primary purpose here is to test the essential reordering algorithm; later work will aim to exploit features such as simplification with HDMR, inclusion of data errors, and generalization to more complicated physical systems. To investigate the influence of structured and unstructured sampling on property prediction, we applied the ordering algorithm to two different data sets, each consisting of 35 library compounds. The first data set A consisted of 35 randomly sampled compounds. The second data set B consisted of 21 compounds selected by cut-center structured sampling and an additional 14 randomly selected compounds to account for nonseparability. The cut-center compound was arbitrarily chosen as the first compound listed in the data table provided by Reynolds et al.[7] The reference sequential total ordering, $\mathbf{Z}^{seq}$ of the substituents was the order in which the substituents were listed in this data table. However, unlike the first two test cases, the original sequential total ordering for the polymer library was not a rational, smooth ordering. Figure 9 shows a plot of $g^{Z_{seq}}(X)$ with the 35 sampled compounds in data set A highlighted, and it is evident that a scattered sampling of $M \ll 12$ compounds will produce unreliable interpolation without first seeking an optimal ordering of the substituents. To find a rational total ordering, **Z***, for which the model function was smooth, the genetic ordering algorithm was used.

The genetic algorithm converged to total orderings for both data sets within 1000 generations. Table 3 compares the total ordering $\mathbf{Z}^A$ obtained for data set A to the total ordering $\mathbf{Z}^B$ obtained for data set B. The orderings of both the diol and diacid substituents obtained for both data sets showed extensive agreement. Figure 10 shows a plot of smoothness versus generation. Figures 11 and 12 show plots of $g(\mathbf{X})$ versus $\mathbf{X}$ using $\mathbf{Z}^A$ and $\mathbf{Z}^B$, respectively. The increase in smoothness for both data sets achieved by the genetic algorithm was significant upon

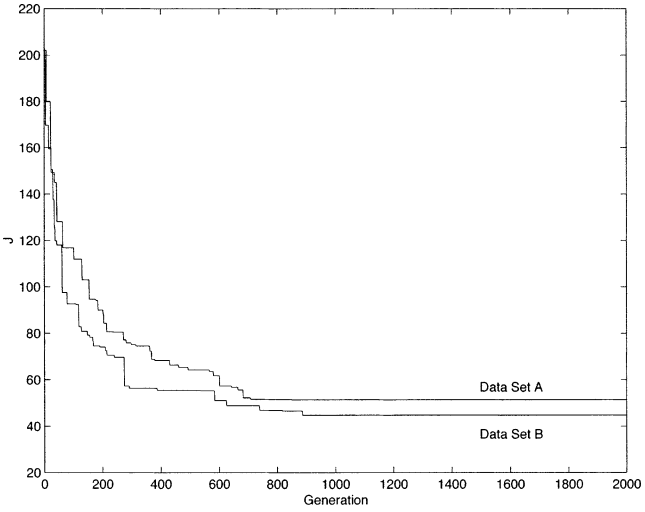


**Figure 10.** Smoothness of best solution versus genetic algorithm generation for the reordering of experimental glass transition temperature, $T_g$, data from data sets A and B. Data set A contained experimental $T_g$ values for 35 randomly sampled compounds. Data set B contained experimental $T_g$ values for 21 compounds selected by cut-center sampling and 14 additional randomly selected compounds.

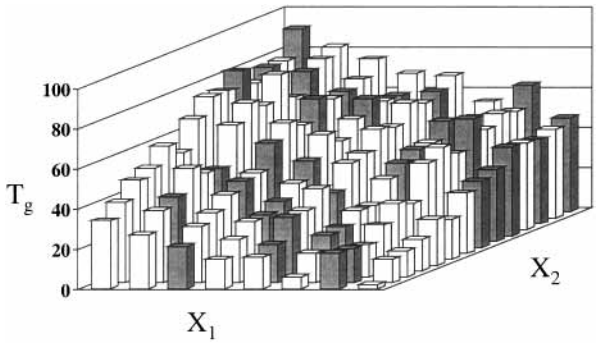


**Figure 11.** Plot of $T_g$ versus polymer substituents for the optimal ordering $\mathbf{Z}^A$ using data set A. $\mathbf{X}_1$ is the diacid substituent variable. $\mathbf{X}_2$ is the diol substituent variable. Compounds sampled in data set A are highlighted.

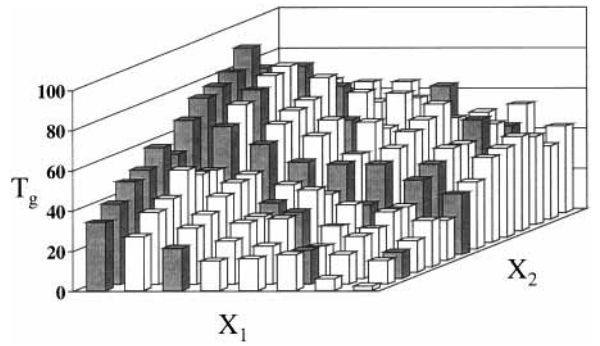


**Figure 12.** Plot of $T_g$ versus polymer substituents for the optimal ordering $\mathbf{Z}^B$ using data set B. $\mathbf{X}_1$ is the diacid substituent variable. $\mathbf{X}_2$ is the diol substituent variable. Compounds sampled in data set B are highlighted.

comparison of these figures to the sequentially ordered function shown in Figure 9. In each case, the genetic algorithm used a data set of only 35 compounds to determine a smooth total ordering, and this total ordering generated a smooth model function over all 112 synthesized compounds. The surfaces $g^{\mathbf{Z}^A}(\mathbf{X})$ and $g^{\mathbf{Z}^B}(\mathbf{X})$ in Figures 11 and 12 from data sets A and B show considerable similarity as well as some distinct differences at smaller values of $X_2$. These differences are of no relevance for property estimation, as smoothness is the only essential

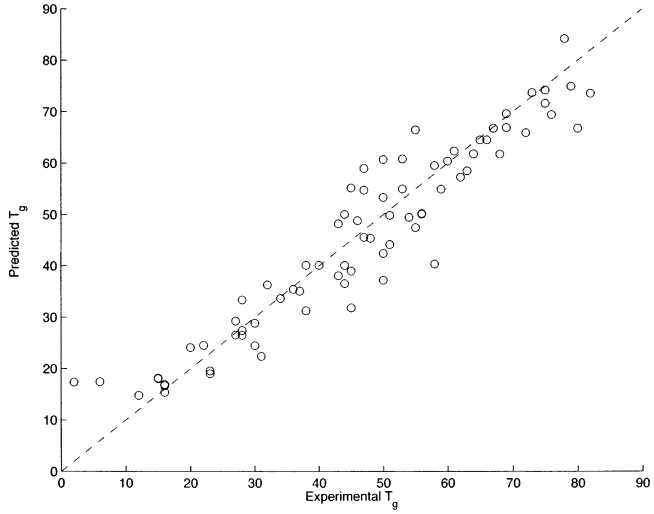


**Figure 13.** Predicted values versus experimental values for $T_g$ of 77 unsampled polymers using data set A.

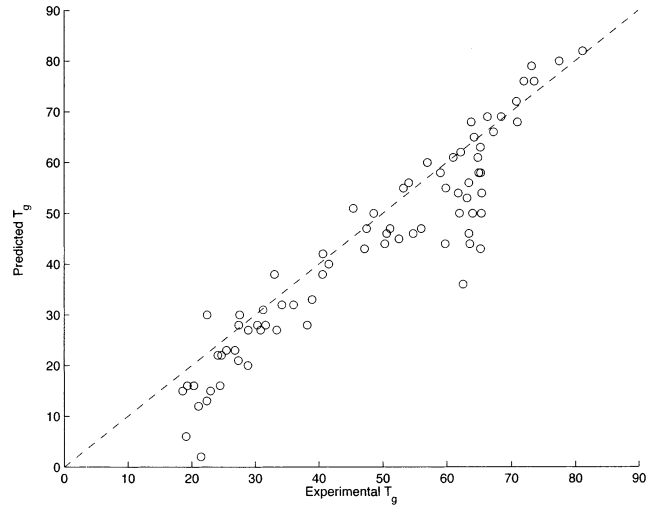


**Figure 14.** Predicted values versus experimental values for $T_g$ of 77 unsampled polymers using data set B.

characteristic needed for interpolation. Both surfaces could serve equally well for property optimization and interpolation as shown below. The precise appearance of the surfaces depends on the defined smoothness cost in eq 5 as well as the sampled data. The introduction of an additional curvature cost (i.e., $(1/R)\sum_i^R||\nabla^2 g^{\mathbf{Z}}(\mathbf{X}_i)||$) would tend to make the surfaces more similar in cases A and B.

The values for $T_g$ predicted by the regression function $f(\mathbf{X})$ using the optimal total ordering $\mathbf{Z}^*$ were compared to the known values of $T_g$ for the unsampled library compounds. Sampling and reordering of the library compounds led to a significant increase in prediction accuracy. Using the randomly sampled data in set A and the corresponding total ordering $\mathbf{Z}^A$ obtained by the genetic algorithm, the prediction of $T_g$ for the 77 unsampled compounds showed an average RMS error of 6.1 °C. Using the structured sampling of data set B and the corresponding total ordering $\mathbf{Z}^B$, the prediction of $T_g$ for the 77 unsampled compounds showed an average error of 8.1 °C. In comparison, the best RMS error achieved by Reynolds et al. was 7.1 °C. Figure 13 shows a plot of the predicted values of $T_g$ versus the experimental values of $T_g$ for the 77 unsampled library compounds using the total ordering $\mathbf{Z}^A$, and Figure 14 shows the same plot using the total ordering $\mathbf{Z}^B$. The statistics in Figure 14 using data set B show a small but evident bias.

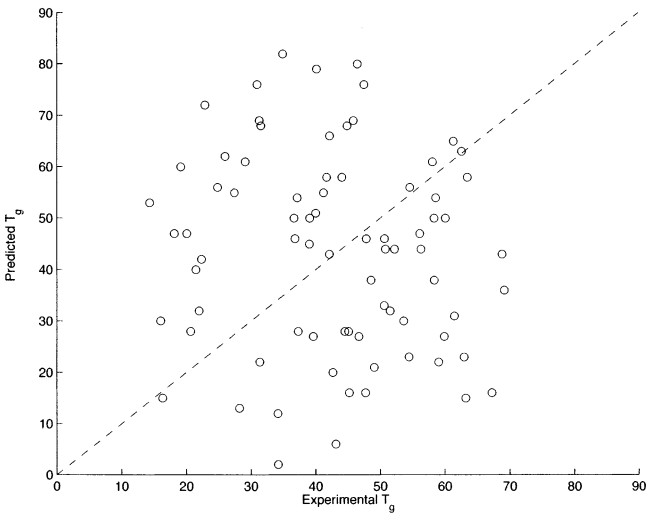


**Figure 15.** Predicted values versus experimental values for $T_g$ of 77 unsampled polymers using data set B without reordering.

This behavior likely arises from choosing a single cut-center for structured sampling, as such a choice puts undue emphasis on the reliability of that small data set.

Finally, we compared the performance of the property prediction algorithm to a straightforward regression approach which did not search for a smooth total ordering. Using the original sequential total ordering $\mathbf{Z}^{seq}$, a radial basis regression was used to fit the experimental data for the 35 sampled compounds from data set B. The prediction of this regression function for the 77 unsampled compounds showed an average RMS error of 25.1 °C. Figure 15 shows a plot of the predicted values of $T_g$ versus the experimental values of $T_g$ for the 77 unsampled compounds using the sequential total ordering. Thus, comparing Figures 13 and 14 to Figure 15 shows that substantial gains in accuracy were made by determining a smooth total ordering for the substituents prior to regression.

Although the quality of the results of Reynolds et al. and those achieved by the reordering of data sets A and B are comparable, the procedures are quite different. Working with descriptors in more complex applications, especially of increasing dimensionality, is a very difficult task. On the other hand, the simple assumption of property regularity should remain just as valid as the dimension rises, and the efficiency of this process should increase with dimension. In the third example, the laboratory synthesis of 35 polymers can be accomplished in a reasonable amount of time, but synthesizing one-third of the compounds in a complete library for the purposes of library optimization is usually unfeasible. Fortunately, the efficiency of the substituent ordering algorithm should improve significantly with increased dimensionality, i.e., more scaffold sites, and with more substituents per dimension. For instance, the ordering algorithm achieved optimal total orderings for the linear and the quadratic three-dimensional simulated model functions using only one tenth of the number of compounds in the complete library. In general, the ordering algorithm is limited by the sampling frequency of each substituent. Optimal performance for random sampling is achieved if each substituent appears in multiple sample compounds. If the library compounds are sampled using cut-center regular sampling, then the number of sample compounds scales as $O(SN)$, along with the addition of some modest number of random samples. On the other hand, the number of total library compounds grows as $O(S^N)$. In low dimensions and small values of $S$, the sampling density for reordering can be achieved only by synthesizing a large fraction of the complete library. However, even for $N = 2$ and $S = 100$, as in many pharmaceutical libraries, a reordering sample size of $M \ll 10^4$ will likely be sufficient. As the dimensionality of the library increases, it is estimated that a much smaller fraction of the complete library may be synthesized for equivalent interpolation performance.

## IV. Conclusions

As the size and complexity of compound libraries increase, new tools will be needed to analyze and utilize the data produced by screening experiments. In particular, the extensive molecular diversity of the substituents used in compound libraries makes the prediction of activity and other properties more difficult. This problem becomes increasingly vexing when multiple competitive properties must be simultaneously met. In such cases, it would be highly desirable to have as thorough a coverage as possible of the full library of compounds. However, direct synthesis of full libraries is generally not possible. The chemical properties of the substituents can also vary dramatically, thereby making it difficult to identify the proper substituent descriptors responsible for compound properties. The inadvertent omission of an important descriptor or overemphasis of a less important descriptor could lead to inaccurate and misleading property prediction. The algorithm presented in this paper does not rely on traditional descriptors. The only "descriptor" used in the ordering algorithm is a discrete numerical index label for each substituent. This simplicity makes for generic applicability of the algorithm.

Substituent ordering provides a means for the elucidation of structure−property relationships without the assumption of a prior model. The substituent ordering algorithm has been shown to be effective in identifying the underlying model structure from simulated and experimental property data. Furthermore, the algorithm can identify an optimal reordering without sampling a prohibitively large number of orderings that would be necessitated by a brute force approach. The efficiency and generality of the substituent ordering algorithm should make it a simple and powerful tool for library optimization and analysis.

Future research will explore the performance of the substituent ordering algorithm for larger compound libraries including those with higher dimensions. The inclusion of physical information regarding the model function will also be considered. For instance, the careful choice of a representative cut-center compound can sometimes improve prediction ability of structured sampling. Use of multiple cut centers and additional randomly sampled data can greatly improve property prediction. It would also be natural to build in the expectation of there being only relatively low-order substituent cooperation to fully utilize the capabilities of HDMR.[8,9] Empirical ordering of the substituents based on perceived important physical properties of the substituents may also be used to guide the ordering algorithm. The impact of input data error on the re-ordering and subsequent interpolation behavior needs to be investigated.

In addition, the use of radial basis functions, though successful, is not necessarily optimal. Other basis sets may be used for estimating the smoothness of substituent orderings and for data interpolation. It is hoped that further development of the substituent ordering algorithm will increase both the efficiency and the accuracy of property prediction in compound libraries. Finally, although this work was motivated by the desire to estimate molecular library properties as efficiently as possible, there are ancillary benefits to revealing library regularity. In particular, it may be argued that revealing regular (i.e., smooth) behavior over the full space of compounds is an essential step

toward physically understanding the origin of molecular properties, as well as molecular recognition in those cases where the properties arise from molecule−receptor interactions.

**Acknowledgment.** The authors acknowledge support from the Princeton Plasma Laboratory and the Air Force Office of Scientific Research.

## References and Notes

(1) Terrett, N. K.; Gardner, M.; Gordon, D. W.; Kobylecki, R. J.; Steele, J. *Tetrahedron* **1995**, *51*, 8135.
(2) Thompson, L. A.; Ellman, J. A. *Chem. Rev.* **1996**, *1*, 555.
(3) Cheng, S.; Comer, D. D.; Myers, P. L.; Saunders, J. *Tetrahedron Lett.* **1999**, *40*, 8975.
(4) Sutton, A. E.; Clardy, J. *Tetrahedron Lett.* **2001**, *42*, 547.
(5) Guan, Y.; Green, M. A.; Bergstrom, D. E. *J. Comb. Chem.* **2000**, *2*, 297.
(6) Brocchini, S.; James, K.; Tangpasuthadol, V.; Kohn, J. *J. Am. Chem. Soc.* **1997**, *119*, 4553.
(7) Reynolds, C. H. *J. Comb. Chem.* **1999**, *1*, 297.
(8) Li, G.; Rosenthal, C.; Rabitz, H. *J. Phys. Chem. A* **2001**, *105*, 7765−7777.
(9) Rabitz, H.; Li, B.; Pierce, L.; Carey, J. In preparation. 2002.
(10) Bard, Y. *Nonlinear Parameter Estimation*; Academic Press: New York, 1974.
(11) Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*; Addison-Wesley: Reading, MA, 1989.
(12) Rabitz, H.; Alis, Ö. F. In *Sensitivity Analysis*; John Wiley and Sons: New York, 2000; pp 199−223.
(13) Rabitz, H.; Alis, Ö. F. *J. Math. Chem.* **1999**, *25*, 197.
(14) Rabitz, H.; Alis, Ö. F.; Shorter, J.; Shim, K. *Comput. Phys. Commun.* **1998**, *117*, 11.
(15) http://lancet.mit.edu/ga/ 2001.