Uniform Coverage Designs for Molecule Selection

Raymond L.H. Lam, William J. Welch & S. Stanley Young

IIQP Research Report
RR-01-06

July 2001

Uniform Coverage Designs for Molecule Selection

Raymond L.H. Lam

Biomedical Data Sciences
GlaxoSmithKline Inc
Mississauga, Ontario L5N 6L4
Canada

William J. Welch

Department of Statistics & Actuarial Science
University of Waterloo
Waterloo, Ontario N2L 3G1
Canada

S. Stanley Young

Statistical Research Unit
GlaxoSmithKline Inc
Research Triangle Park,
North Carolina 27709-3398
USA

Monday, March 19, 2001

In screening for drug discovery, chemists often select a large subset of molecules from a very large database (e.g., select 1,000 molecules from 100,000). To generate diverse leads for drug optimization, highly active compounds in several structurally different chemical classes are sought. Molecules can be characterized by numerical descriptors, and the chosen subset should cover the descriptor space, or subspaces formed by several descriptors. We propose a method that concentrates on low-dimensional subspaces, a criterion for uniformity of coverage, and a fast exchange algorithm to optimize the criterion. These methods are illustrated using a National Cancer Institute database.

KEY WORDS: Drug discovery; High throughput screening; Space-filling design; Projection; Binning; Exchange algorithm.

1. INTRODUCTION

The use of robotics and miniaturization is now allowing researchers to quickly screen thousands of chemical compounds (molecules) for biological activity. Combinatorial chemistry provides the logistics of mass production of compounds and a wide range of molecular diversity for drug discovery. The automation of biological assays, High Throughput Screening (HTS), allows for investigation of thousands of chemical compounds against biological targets per week. While this brute-force approach to lead generation certainly has its place in the field of drug discovery, it is not practical, given the size of today's chemical libraries (e.g., hundreds of thousands to millions of compounds), to test every available compound for every new target of potential importance.

Various molecular descriptors (explanatory variables) can be readily computed to describe the chemical properties of every molecule in the database. When there is no prior model relating biological response to these descriptors, the generally accepted procedure is to screen (test) a diverse subset of the overall database to find active compounds of several structurally different chemical classes, and then examine further compounds that are structurally similar to any promising leads. If multiple chemical classes can

be found, they provide optional starting points for further optimization of activity, physical properties, tissue distribution, plasma half-life, toxicity, etc. Ideally, selected objects should be as dissimilar as possible and any candidate not selected should be near a molecule in the experimental design. Measures of "diversity" and "similarity" are based on the numerical descriptors. The assumption here is that similar chemical objects are more likely to have similar biological responses. Thus, if an initial subset is to be selected, the subset should "fill" or "cover" the numerical space. In high dimensional space, nearly all data sets are sparse, and it is not possible to densely cover a high-dimensional space with thousands of design points. Therefore, we focus on filling or covering low-dimensional projections of the space instead.

To measure the "coverage" of a descriptor space, we will be dividing the space into cells. In a conventional cell-based method, each of k numerical descriptors is subdivided into m bins of equal size, yielding m^k cells or hypercubes, and the experimental design chooses at least one molecule from every cell. A good experimental design will ideally have at least one molecule in every cell. If so, we say the space is covered. Cummins et al. (1996) and Menard et al. (1998) used cell-based methods to compare the relative diversity of molecular databases and to select diverse subsets of molecules.

Such cell-based methods are attractive for several reasons. It is easy to divide the descriptor space into cells, and allocating even a very large dataset to these cells is straightforward. Choosing a design by random sampling is also easy. Missing diversity (i.e. empty cells) can easily be identified.

The key problem with the conventional cell-based method is that a high-dimensional space will have too many cells to be covered by a modest number of compounds (design points). As two molecules must have fairly close values of all critical descriptors for similar biological activity (McFarland and Gans 1986), the number of bins, m, should be relatively large. If k=6 and m=10, say, we have one million cells, which cannot be covered by only thousands of design points. This is just the curse of dimensionality.

To reduce the number of cells, a common approach is to use fewer, wider bins in each dimension, even though these bins may include rather dissimilar compounds. For example, Cummins et al. (1996) and Menard et al. (1998) restricted the number of descriptors and the number of bins per descriptor. They also excluded hundreds to thousands of outlying candidate points (as outliers lead to an artificially large space). Even with these compromises, they reported a large proportion of empty cells, many compounds densely clustered in a few cells, and many cells being singleton. Indeed, a very low cell occupancy (i.e., at least one compound) rate is expected by Menard et al. (1998) — they recommended a target occupancy of 12-15%. If most cells are empty and hence most of the space is ignored, however, the utility of covering the remaining space is questionable, calling for new methods of binning and creating cells.

If only a few descriptors are responsible for the particular biological activity, however, it is possible to densely cover their low-dimensional subspace with just thousands of design points. A subspace is simply a subset of the descriptor variables, ignoring the remaining descriptors. Different sets of critical descriptors may be relevant to structurally different chemical classes, but hopefully only a few variables are involved at a time. If we knew, in advance, that certain subsets of descriptors were critical we could choose design points to give good coverage of the relevant subspaces. At the outset, we will probably not know which descriptors are critical, and we therefore aim for uniform coverage in every low-dimensional projection. With m=10 bins per descriptor, for example, it is theoretically feasible to cover all 10^3 cells in any three-dimensional subspace with about 1000 points. This is analogous to a fractional-factorial design projecting down to a full factorial in a few critical variables.

Thus, because of the practical difficulty of covering the numerical space of all descriptors, and the belief that probably relatively few descriptors are active for any given mechanism, we will concentrate on low-dimensional subspaces throughout this article, typically involving one, two, or three descriptors.

Designs with good coverage of low-dimensional subspaces have been suggested in many other contexts. For example, Dalal and Mallows (1998) proposed plans for testing software such that for any f input factors, all combinations of their levels occur at least once. Typically, f is 2, 3, or 4. Thus, these designs exhaustively cover the input-factor space when projected down onto f-dimensional subspaces. Although the objectives are similar, these plans cannot be directly applied to molecule selection. Suppose we grouped each descriptor's values into a moderate number of bins to generate "levels". For an experimental run, the Dalal and Mallows (1998) designs can choose any combination of levels (bins) over all factors (descriptors). Unfortunately, a set of candidate molecules will typically have some bin combinations that are empty. We start with a candidate set of molecules, and we cannot necessarily select an arbitrary combination of descriptor values and place a design point there. The haphazard combinations of descriptor values similarly rule out plans based on Latin hypercubes and orthogonal arrays with good projective properties (Owen 1992 and Tang 1993) that have been proposed for computer experiments. The same difficulty arises with many other designs aiming for uniform space-filling properties, for example, the uniform shell designs of Doehlert (1970) or number-theoretic methods for generating representative points motivated by discrepancy measures (e.g., Fang et al. 1994).

Algorithmic, rather than combinatorial, methods can generate space-filling designs from any given set of candidate points. They typically optimize some function of the inter-point distances. Johnson et al. (1990) proposed two classes of designs, based on either minimax or maximin distance criteria. Maximin designs maximize the minimum distance between design points. By making the design points maximally dissimilar they spread throughout the space; the algorithm of Kennard and Stone (1969) has this underlying objective. Alternatively, minimax distance designs minimize the maximum distance between candidate points and the design points. This criterion tries to make every candidate close to a design point and hence the design covers the candidate space. Similarly, Zemroch (1986) clustered the candidate points and chose a member of each cluster to cover or represent the entire set. Thus, distance-based

algorithms appear to be useful for molecule selection and have been applied in this context (Higgs et al. 1997) and are readily available in SAS (Tobias 1995, pp. 657-728).

There are several difficulties, however, with distance-based design criteria. All of the methods mentioned above are based on distance metrics calculated from all descriptors. As we have already noted, it is not possible to densely cover a high-dimensional space with only thousands of points. Low-dimensional coverage, which is more relevant if few descriptors are critical, is not directly considered and could be quite uneven (some results will be presented in Section 6.4). Moreover, the definition of an appropriate metric is problematic for molecular descriptors. Two molecules with fairly close values of all critical descriptors are likely to have similar biological activity (McFarland and Gans 1986), but beyond some (unknown) threshold, there may be little relationship between distance and similarity of activity. Finally, the presence of relatively few outlying observations leads to large, dominating inter-point distances. Very often this requires removal of many molecules to generate a sensible design.

The simplest designs are based on random sampling. In fact, most new leads have been discovered through random screening, in which large numbers of compounds are tested for a specific biological activity, and the active compounds are then selected for optimization. Young et al. (1996) used a constant radius hypersphere around each randomly selected compound to measure the coverage of the descriptor space. They concluded that, unless a very large number of compounds are used to fill space, randomly selected compounds will cover as much space as carefully selected compounds. Again, however, if relatively few descriptors are important, then a rational selection should be more effective than a random design. In Section 6.2 we examine the coverage of designs generated by simple random sampling and stratified random sampling.

The approach proposed in this article is to divide all low-dimensional subspaces into small cells and attempt to find a design that has one point in every cell of every subspace, so covering every low-

dimensional subspace. In Section 2 we describe a National Cancer Institute (NCI) database that we will use to motivate and illustrate our methodology and notation for the general case. Section 3 discusses a data-adaptive descriptor binning method that leads to two- and three-dimensional cells such that only a small proportion are empty with respect to the candidates. To guide the choice of the design points from the candidates, we develop a uniform cell coverage (UCC) criterion in Section 4, and Section 5 describes a fast exchange algorithm to implement it. In Section 6 we apply the UCC criterion to the NCI data and compare computational time and quality of coverage relative to other methods. Finally, Section 7 provides some conclusions and discussion of further work.

2. CHEMICAL DATABASES AND DESCRIPTORS

2.1. The NCI Candidate Set

We illustrate our methods with the NCI AIDS antiviral screen database, because it is a large database in the public domain and represents a problem of practical importance. The activity data can be obtained from the web site http://dtp.nci.nih.gov/docs/aids/aids_data.html. When we downloaded the data in May 1999, there were 32,110 compounds in the database. Some were removed, because their descriptors could not be computed, leaving 29,812 molecules.

We use six continuous BCUT variables as descriptors. They are based on the work by Burden (1989), who found that structurally similar compounds have similar BCUT values. They tend to characterize molecular bonding patterns and atomic properties such as surface area, charge, hydrogen-bond donor and acceptor ability. The BCUT values for the 29,812 molecules are available from the first author.

Figure 1 shows the univariate distributions of the six descriptors for the NCI candidate molecules. The distributions exhibit multimodality and outlying values. The pairwise plots in Figure 2 show that the two-dimensional projections are complex, with much empty space. Either the collection is missing chemicals

or it is not possible to make compounds with certain combinations of descriptors. In more than two dimensions this problem will be even worse.

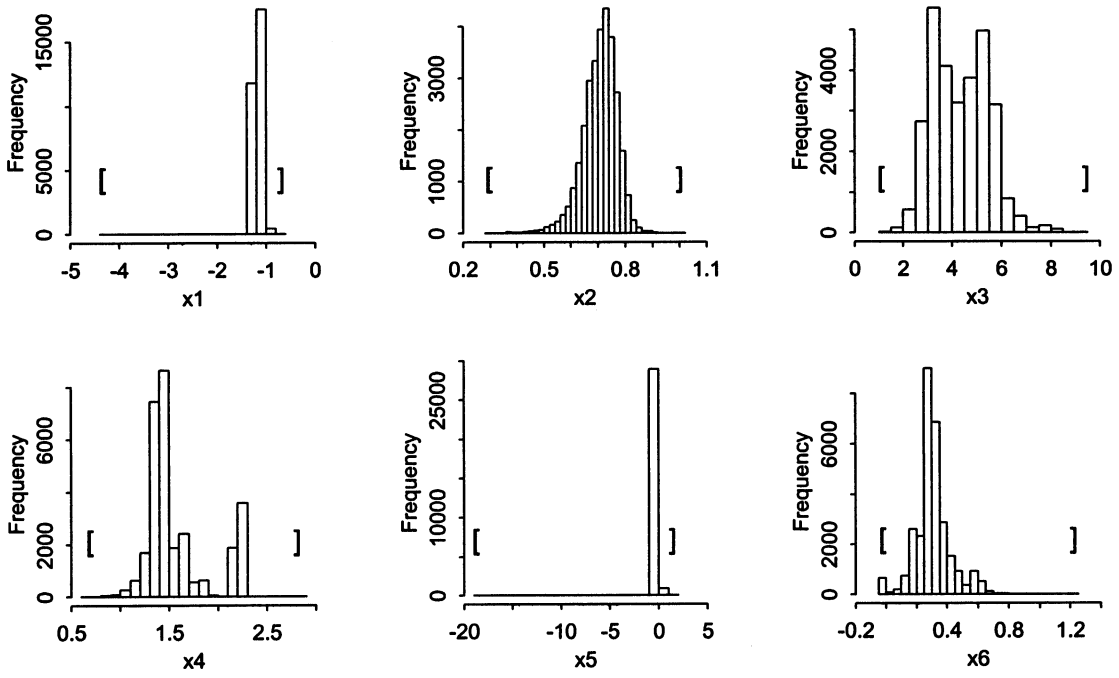

Figure 1. Univariate Distributions of the Descriptors in the NCI Data. The "[" and "]" symbols denote a descriptor's range.

2.2. Notation

In general, denote the k continuous descriptors by $x_1, x_2, \ldots, x_k$, and let X_c be a candidate set of compounds with N points. The objective is to choose a representative set of n design points, X_d, to cover the descriptor space occupied by the candidate set.

Within the full k-dimensional descriptor space, a p-dimensional (p-D) subspace is defined by p of the k descriptors ($1 \leq p \leq k$). For convenience, Xi will denote the 1-D subspace involving only x_i. Similarly, Xij, $Xijl$, etc. will represent 2-D, 3-D and higher-dimensional subspaces. For example, $X1$ is a 1-D

subspace defined by x_1, and $X12$ is a 2-D subspace formed by x_1 and x_2. A subspace, then, is simply a subset of the descriptor variables, ignoring the remaining descriptors.

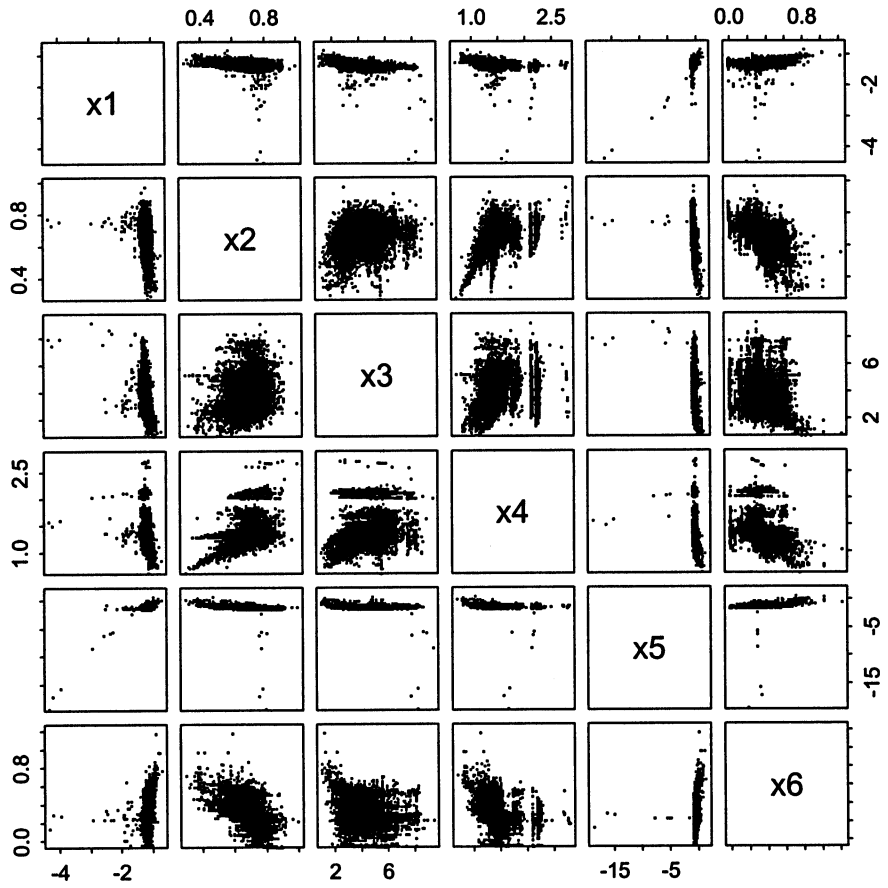

Figure 2. Pairwise Plots of the Descriptor Values in the NCI data.

3. CELL-BASED APPROACH

We use a number of techniques to keep the cells small, yet limit their number, and to ensure that relatively few cells are empty in the candidate set. First, when we bin each descriptor, we adopt a data-driven hybrid binning method that makes bins larger towards the extremes. This avoids empty bins towards the limits of a descriptor's range, where molecules tend to be sparse. Second, we focus attention on low-dimensional subspaces, typically all 1-D, 2-D, and 3-D subspaces. By considering no more than

three variables at a time, fewer cells are required to represent a subspace. Selecting a design with good coverage of all low-dimensional subspaces is analogous to a two-level fractional factorial design of Resolution IV. Such a design is a complete factorial for any subset of three or fewer variables (Box, Hunter, and Hunter 1978, p. 388) and can estimate all interaction effects if only three factors are found to be important. Third, every subspace considered has the same number of cells, avoiding the exponential increase with dimension.

3.1. Data-Driven Binning

For each descriptor, we first divide its range into mutually exclusive and exhaustive sub-ranges or bins (e.g., we use 729 bins in Section 6 for the NCI data). The bins for descriptor x_i immediately become the cells for the 1-D subspace *Xi*. For subspaces of higher dimension, cells will be formed from the bins of the descriptors forming the subspace (Section 3.2).

To construct bins, we use a hybrid of two simple-to-implement methods: equal width (EW) and equal frequency (EF). The EW method simply divides a descriptor's range into equal-width intervals. Alternatively, EF bins have their cut-points chosen to make the frequency of candidate molecules approximately equal in each bin.

In regions where there is a reasonable density of descriptor values, EW bins are compelling. When a molecule is chosen to represent a bin (and hence a cell), it is the size of the bin that determines the quality of coverage in the descriptor space, not the number of molecules in a bin. Another way of looking at this is that EF bins are very small where there is a high density of candidate molecules. Such regions will be over-represented in an experimental design, to the detriment of coverage in regions where candidates are sparse and bins are wide.

On the other hand, outlying or extreme descriptor values may inflate a descriptor's range, making many EW bins empty towards the extremes. This problem is compounded when we form cells in multiple dimensions (Section 3.2). To avoid empty bins, extreme candidates are sometimes removed from consideration (Cummins et al. 1996 and Menard et al. 1998). By definition, the EF method has candidate points in every bin and hence none are empty. Empty cells in 2-D or 3-D subspaces can still arise, but EF bins will tend to have fewer empty cells.

To combine the best features of EW and EF bins, we use a data-driven, hybrid method. EF bins are constructed for the extreme values. For example, the first percent of a descriptor's values can be placed in one bin, with a similar bin for the last one percent. EW bins are then used between these extreme bins. Thus, EW bins predominate, while the EF method for the extreme values avoids empty bins.

Figure 3 illustrates the advantage of this hybrid binning strategy, applying it to x_1 from the NCI data. Here, to keep the demonstration of binning and cell construction simple, we use 64 bins. (When we apply these methods to a realistic sized design in Section 6 we will use 729 bins.) With EW bins the frequencies shown in Figure 3(a) are very uneven: Of the 64 bins, 32 in the long tail to the left are empty. In contrast, all of the 64 hybrid bins shown in Figure 3(b) are occupied. The 62 equal-width bins in between the two 1% end bins are much narrower on the original x_1 scale. Compounds within these narrower bins are more likely to have similar activity.

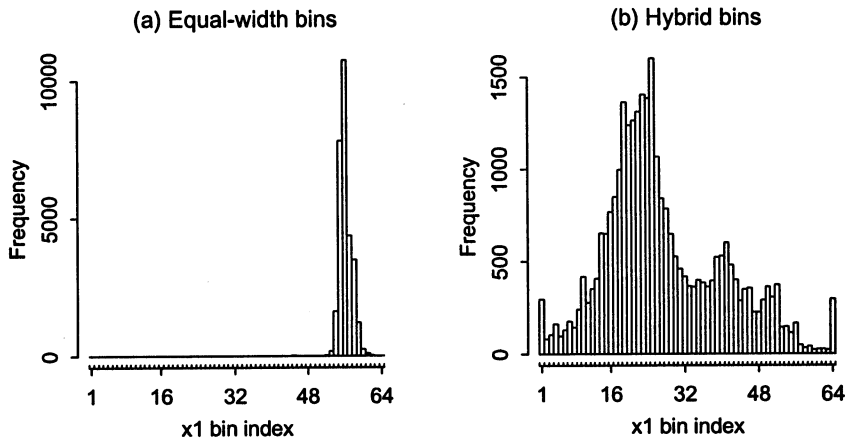

Figure 3. Candidate-Point Bin Frequencies for Descriptor x_1 in the NCI data (64 Bins).

After binning, we make no further use of the raw x_i values; our uniform-coverage algorithm only uses the hybrid-bin index to characterize a molecule according to x_i. The data-adaptive binning procedure is repeated for each descriptor.

3.2. Forming Cells

To form cells for, say, a 2-D subspace, we combine the 1-D bins for each of its two descriptors. We keep the number of cells constant over subspaces, however, and hence avoid the curse of dimensionality. This is effected by amalgamating 1-D bins when working in a higher-dimensional subspace. For example, if we have used 64 bins for each 1-D subspace, as in Figure 3(b), we divide the 2-D subspace X12 for descriptors x_1 and x_2 into $8 \times 8 = 64$ cells, as illustrated in Figure 4. The first cell, in the lower left corner, for instance, is formed from the first eight 1-D bins for X1 and the first eight 1-D bins for X2. This gives the 64 2-D cells shown. Similarly, we form 3-D subspaces of 4 x 4 x 4 = 64 cells by amalgamating 1-D bins 16 at a time for each descriptor.

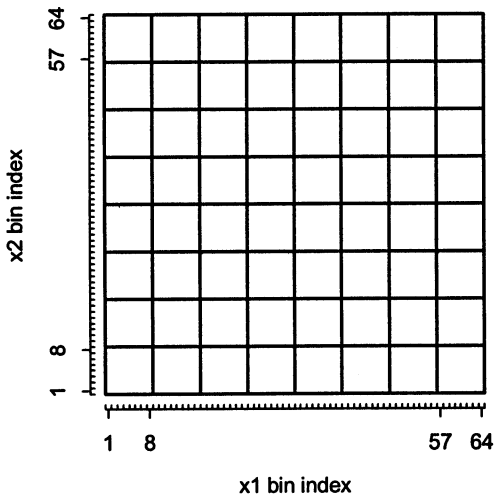

Figure 4. Construction of 64 2-D Cells From Descriptors With 64 Bins.

In general, suppose we consider 1-D, 2-D, and 3-D subspaces and want m cells per subspace. For 2-D subspaces, analogously to Figure 4, cells are formed in an $m^{1/2} \times m^{1/2}$ array, and for 3-D subspaces there is an $m^{1/3} \times m^{1/3} \times m^{1/3}$ array of cells. Thus, convenient values of m have integer square roots and cube roots: $2^{2x3}=64$, or $3^{2x3}=729$, or $4^{2x3}=4096$, etc.

With k descriptors, there are

$$\binom{k}{1}+\binom{k}{2}+\binom{k}{3}=\frac{5}{6}k+\frac{1}{6}k^3$$

1-D, 2-D, and 3-D subspaces in total. When k=3, for example, there are 7 subspaces: $X1$, $X2$, $X3$, $X12$, $X13$, $X23$ and $X123$. When k=6, there are 41 subspaces and when k=10, there are 175 subspaces. For larger k, it might be necessary for computational reasons to reduce the number of subspaces by focusing on only 1-D and 2-D subspaces.

Including subspaces of 4-D and higher will usually not be practical. Chemists believe that two molecules must have fairly close values of all critical descriptors for similar biological activity (McFarland and Gans 1986). This means that bins have to be small if one molecule from a bin is to represent the rest. Yet, even with 10 bins per dimension, which is probably too few, there are 10,000 cells per 4-D subspace. Clearly, we would need to choose at least this many molecules if the experimental design is to cover every cell. Thus, it is not possible to give dense coverage of a 4-D subspace with a modest subset of molecules. For analysis, this implies that interaction effects are hopefully limited to no more than three factors.

How big should m be? Even with the data-driven binning method in Section 3.1, there will be some multi-dimensional cells with no molecules. The proportion of empty cells, which varies from subspace to subspace, will tend to increase with m. In addition, if n design points are to be selected, we would like n nonempty cells per subspace, so the space-filling design can cover distinct nonempty cells. These two considerations suggest that m should be approximately equal to n or a little larger.

4. CRITERIA FOR EVALUATING COVERAGE

In a conventional cell-based design (Section 1), there is one set of cells based on all k descriptors. Simply picking a point from each occupied cell would guarantee a good coverage design. As already noted, however, this approach often generates many more cells than the number of design points, making good coverage impossible. In Section 3.2 we defined cells based on low-dimensional subspaces to overcome this problem. With more than one subspace, it is no longer straightforward to select a set of candidate points to give good coverage simultaneously in many subspaces. If, say, one point is chosen from each cell in a particular subspace, these points may be unevenly distributed in other subspaces. We now describe two measures of the quality of coverage; the second will be used in Section 5 as an optimization criterion to drive the numerical search for a good experimental design.

We first need some definitions and notation. Let X denote a set of points (molecules) in the descriptor space; X will typically be the entire set of candidate points, X_c, or a trial experimental design, X_d. The set X is said to *cover* cell i in subspace s if at least one of the points falls in that cell. Mathematically, we set up indicator variables $c_{si}(X)$ taking the value 1 if cell i in subspace s is covered and 0 otherwise.

4.1. Average Percentage of Cells Covered

The first experimental design criterion simply computes the percentage of cells that are covered by a design, averaged over all subspaces. Some cells are not covered by the candidate set, X_c, and so cannot be covered by any choice of design; these cells are eliminated from consideration when computing the criterion.

In subspace s, the percentage of cells covered by a design X_d is defined to be

$$P_s = \frac{\sum_i c_{si}(X_d)}{\sum_i c_{si}(X_c)} \times 100\%,$$

where the summation is over all cells in the subspace (i.e., i=1, ... , m). We can then define the average percentage coverage over, say, all 1-D subspaces as

$$P_{1-D} = \frac{\sum_{s \in S_1} P_s}{|S_1|},$$

where S_1 is the set of all 1-D subspaces and $|S_1|$ is the number of such subspaces. For 2-D subspaces we define $P_{2\text{-}D}$ analogously, and so on.

We can then obtain the average percentage coverage, P. For example, if 1-D, 2-D, and 3-D subspaces are being considered, we have

$$P = \frac{P_{1-D} + P_{2-D} + P_{3-D}}{3}. \qquad (1)$$

The average could also be weighted, for example giving more weight to 1-D subspaces.

One deficiency of this criterion is that it ignores the distribution of design points in the covered cells. For instance, consider two very different designs: one has two points in each of 50 cells and the other has 1 point in each of 49 of these cells and 51 points in the remaining cell. With respect to these 50 cells, the coverage is 100% for both designs, yet we would prefer the first as the distribution of points is more uniform. Thus we report the criterion P in Section 6, but the selection of a design is a based on a modification that takes the uniformity of coverage into consideration.

4.2. Uniform Cell Coverage (UCC)

Suppose design X_d places $n_{si}(X_d)$ points in cell i of subspace s. If the candidate set X_c does not cover this cell, i.e., $c_{si}(X_c) = 0$, then $n_{si}(X_d)$ also has to be 0. For cells that are covered by X_c, i.e., $c_{si}(X_c) = 1$, we want the $n_{si}(X_d)$ counts to be approximately 1. Thus, ideally, $n_{si}(X_d) = c_{si}(X_c)$ for every cell. In subspace s, then, a measure of lack of uniformity is

$$U_s = \sum_i \left[n_{si}(X_d) - c_{si}(X_c) \right]^2 . \qquad (2)$$

Again, we can average these quantities over subspaces. The total lack of uniformity for 1-D subspaces, for example, is

$$U_{1\text{-}D} = \frac{\sum_{s \in S_1} U_s}{|S_1|},$$

and analogously for $U_{2\text{-}D}$, etc. Averaging with weights across, say, the 1-D, 2-D, and 3-D subspaces, we have the uniform cell coverage (UCC) criterion:

$$U = \frac{w_1 U_{1\text{-}D} + w_2 U_{2\text{-}D} + w_3 U_{3\text{-}D}}{w_1 + w_2 + w_3}, \qquad (3)$$

where w_1, w_2, and w_3 are user-supplied weights. A user might want to give more weight to 1-D coverage and least to 3-D coverage, for example. In all of the examples in Section 6 we use equal weights.

Minimizing U in (3) discourages uncovered cells in the design and tends to avoid having more than one design point per cell. This is the criterion used by the optimization algorithms of the next section.

The indicator variables $c_{si}(X_c)$ in (2) provide the target numbers of points per cell in the UCC criterion. With a simple modification to these targets, a generalized UCC is obtained. For example, suppose that the number of design points allows about two design points in each cell. We can set the target for a cell to 0, 1, or 2 if there are no candidate points, one point, or at least two points, respectively. In the examples of this paper, we use (2) without modification.

5. FAST EXCHANGE ALGORITHM

5.1. Basic Exchange Algorithm

An optimization algorithm is needed to implement the minimization of the UCC criterion in (3). In other contexts, primarily efficient experiments for fitting regression models, there are many algorithms for optimizing a design criterion; see Cook and Nachtsheim (1980) and Tobias (1995, pp. 657-728) for reviews. Most of these algorithms are variants on the basic idea of an exchange. Starting with n points in a trial design, they exchange a point in the design for one in the candidate set to improve the design criterion and iterate until the criterion cannot be improved further. However, these methods were not intended for problems of the magnitude considered here (i.e., select thousands of points from hundreds of thousands) and would be far too slow.

We could modify any one of several implementations of this idea. We choose to start with the Wynn (1972) algorithm, which we call the basic exchange algorithm below, because it is fast relative to other methods (Tobias 1995, pp. 657-728) and its simplicity facilitates adaptation. The modifications greatly reduce the computational effort, especially when dealing with very large candidate sets.

The basic exchange algorithm starts with a random subset of n points (an initial design) from the N candidates. The optimization criterion is then sequentially improved by a series of exchanges. (Wynn worked with the D optimality criterion, but we will use UCC.) In each exchange, a point in the candidate set replaces a point in the current design. An exchange is broken down into two steps. First, a point in the candidate list is found to add to the current design. The point added from the candidate list is the one with the best value of the design criterion for the modified design of $n + 1$ points. Second, a point in the new design of $n + 1$ points is removed; this point is chosen to give the best criterion value for the new design of n points amongst those that are subsets of the $n + 1$ points available. These exchanges continue until the criterion cannot be improved. We now describe the adaptations to this exchange concept.

5.2. Identifying Good Candidates for Exchange

The basic exchange concept is computationally inefficient for large candidate lists. In principle, we have to loop through the whole candidate list, X_c, to find only one candidate to add. Moreover, many of the initial n points will have to be replaced, requiring many loops if n is moderately large. The adaptations we first describe are aimed at obtaining many exchanges per X_c loop, thereby reducing the number of X_c loops required. Every time a candidate is visited, we note the improvement in the criterion if it were added to the design. Hence, an approximation to the distribution of improvements can also be maintained. As we pass through the candidates, whenever a candidate's change is in the upper tail of this distribution, it is deemed "good" and considered for an exchange. (A similar process will be described in Section 5.3 to search for a design point to delete and complete the exchange.) Thus, each X_c loop might identify many "good" candidates and carry out several exchanges.

Specifically, let δ_j denote the improvement (i.e., reduction) in the UCC criterion U in (3) if candidate j were added to the current n design points to give $n + 1$ points. The algorithm for identifying good candidates, with some explanation in parentheses, is as follows:

1. Initialize the δ distribution. Randomly select 100 candidate points. Compute their δ values, and denote the sorted values by $\delta_{(1)} \geq \ldots \geq \delta_{(100)}$. Set $\lambda = n/N$ and $\delta^* = \delta_{(q)}$, where $q = \max(1, 100\lambda)$. (In Step 2, if candidate j has $\delta_j \geq \delta^*$, it will be considered for an exchange. This rule will try approximately n of the N candidates during the first X_c loop, because all n initial design points may have to be replaced.)
2. Loop through the candidates. For $j=1,\ldots, N$ do the following steps:
    - Compute δ_j and note the value for later use in updating δ^*.
    - If $\delta_j \geq \delta^*$, then:
        - Try exchanging candidate j with one of the current design points (see Section 5.3).
        - If candidate j was exchanged, then
            - Set $\delta_j = -100$. (As candidate j is now in the design, introducing it again is undesirable.)

        else
            - Replace δ^* by $\delta^* + 10\lambda$. (A failed exchange suggests that δ^* is allowing poor candidates to be considered, i.e., δ^* is too small.)
3. If there was no improvement in the criterion in the last X_c loop, then stop.
4. Update δ^* for the next X_c loop. Sort the δ_j values from the last X_c loop and denote them by $\delta_{(1)} \geq \ldots \geq \delta_{(N)}$. Set λ to half the previous value and $\delta^* = \delta_{(q)}$, where $q = \max(10, N\lambda)$. Go to Step 2. (Decreasing λ reduces the number of exchanges considered, because fewer exchanges are likely to improve the criterion with successive passes through the list. We always want to consider at least 10 promising candidates in the next X_c loop, however, to be conservative about termination.)

Note that when a good candidate is found in Step 2, we do not re-start the X_c loop at the beginning. Rather we continue with the next candidate. These "floating" loops allow many exchanges in one X_c loop.

5.3. Identifying Design Points for Exchange

Whenever a "good" candidate for inclusion in the design is identified by the rules in Section 5.2, a design point must also be removed if an exchange is to take place. We evaluate the design points and identify a "bad" point, i.e., one that should be removed, using similar rules.

Specifically, for a fixed candidate j under consideration for inclusion, let Δ_i denote the overall improvement in the UCC criterion in Equation (3) if design point i of the n current design points is replaced by candidate j. Thus, Δ_i includes the δ_j contribution from adding candidate j. A distribution of Δ_i values is maintained, and we implement an exchange as soon as a "good" Δ_i value is found, rather than search all n design points. The details are as follows:

1. Initialization of the Δ distribution. If this is the first search of the design list, then:
    - Randomly select 100 design points, compute their Δ_i values, and denote the sorted values by $\Delta_{(1)} \geq \ldots \geq \Delta_{(100)}$.
    - Set $\Delta^* = \max(0.01, \Delta_{(q)})$, where $q = \max(1, 100\lambda)$, using the λ value in effect for searching the candidate list. (Exchanges with $\Delta_i \geq \Delta^*$ will be implemented.)
    - Set i=1. (Start at the top of the design-point list.)
2. Compute Δ_i and note the value for later use in updating Δ^*.
3. If $\Delta_i \geq \Delta^*$, then
    - Implement the exchange of design point i with candidate j.

    else if all design points have been tried, then
    - Let Δ_{max} be the maximum Δ value over all the design points. If $\Delta_{max} \geq 0$, then
        - Implement the exchange of the design point giving Δ_{max} with candidate j.
4. If $i=n$, then

- Update Δ^*. Sort the Δ_i values from the last X_d loop and denote them by $\Delta_{(1)} \geq \ldots \geq \Delta_{(n)}$. Set $\Delta^* = \max(0.01, \Delta_{(q)})$, where $q = \max(1, n\lambda)$, using the λ value in effect for searching the candidate list.
- Set i=1;

else

- Set i to i+1.

5. If an exchange occurred in step 3 or all design points had been tried in step 3, then
   - Return to searching for the next "good" candidate to add. The next search for a "bad" design point to remove will start at Step 2 with the current value of i.

   else
   - Go to Step 2.

Note that in Step 3, an exchange can occur with $\Delta_{max} = 0$, i.e., it does not change the criterion. Allowing "neutral" exchanges of this type may be useful to break away from a design that is only locally optimal.

5.4. Updating the UCC criterion

Finally, we describe how the criterion can be efficiently updated when only one point is changed, either when adding a candidate or when removing a design point.

When a point is added to or removed from the design, it will affect only one of the m cells in each subspace. Let z_s be the number of design points in the affected cell in subspace s. If we are adding a point, then z_s becomes $z_s + 1$, and the change to U_s in (2) is

$$[(z_s+1)-1]^2-(z_s-1)^2 = 2z_s-1.$$

Note that $c_{si}(X_c)$ in (2) must equal 1, as a cell must be covered by at least one candidate if a point is to be added (or removed). Similarly, when a design point is removed, the change to U_s in (2) is

$$[(z_s - 1) - 1]^2 - (z_s - 1)^2 = 3 - 2z_s .$$

6. RESULTS

We now apply our data-driven binning method and our fast design algorithm to select 729 molecules from the 29,812 NCI molecules.

6.1. Forming Cells

The distributions of the NCI molecules in 1-D and 2-D projections for all six descriptors are shown in Figures 1 and 2. To apply the hybrid binning method described in Section 3.1, the first and last percentiles are assigned to EF bins, with EW bins between. There are six 1-D, 15 2-D, and 20 3-D subspaces, and each of these 41 subspaces is divided into 729 cells. Over the 41 subspaces, on average there are 81.4% nonempty cells in the candidate set of molecules; the worst subspace is X246 with 63.0% nonempty cells. Figure 5 shows the bin (1-D cell) counts for the 1-D subspaces. The plot for x_4 shows that adding a few extra EF bins in sparse regions could further increase the proportion of nonempty bins, but we do not pursue this. For 2-D subspaces, bins are amalgamated 27 at a time to generate $27 \times 27 = 729$ cells. Figure 6 depicts cells with at least one candidate point with a dot. It is seen that the 2-D cells are fairly well covered by the candidates. Similarly, the 3-D subspaces (not shown) have $9 \times 9 \times 9$ cells formed by amalgamating 81 bins at a time in each dimension.

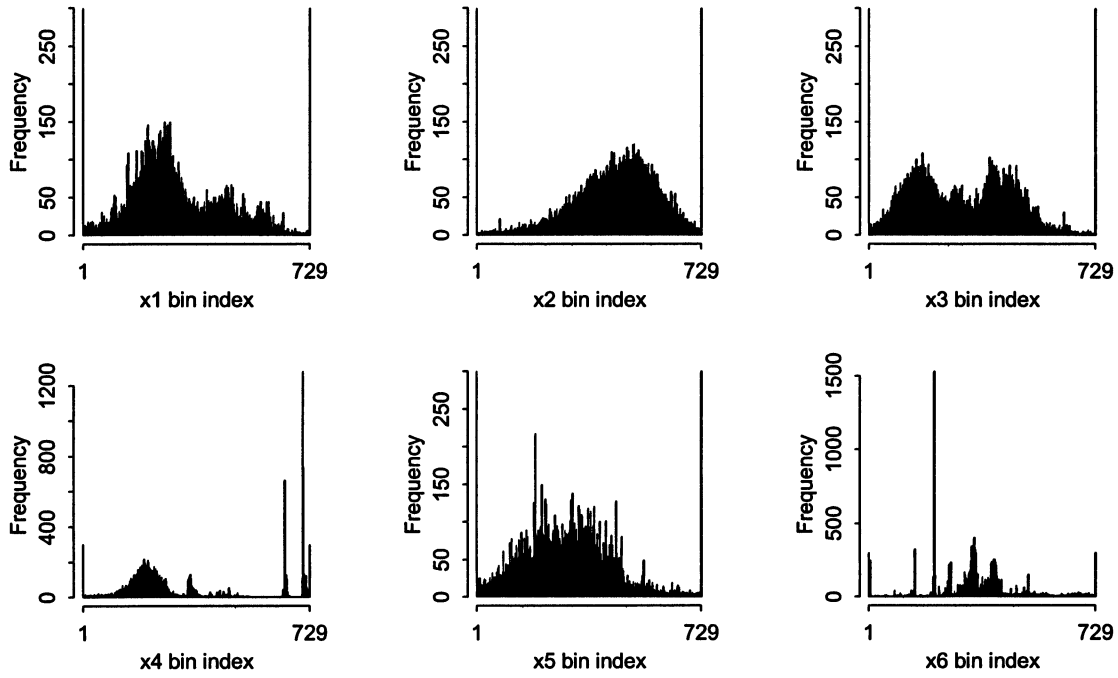

Figure 5. Candidate-Point Bin Frequencies for the NCI data (729 Hybrid Bins).

6.2. UCC Optimization Algorithm Versus Random Designs

The algorithm in Section 5 to minimize the UCC criterion in (3) gives a U value of 585. For comparison, we also generate 100 designs based on simple random sampling (SRS) of 729 points from the 29,812 candidates and compute their values of the UCC criterion. Figure 7 shows that the U value give by the UCC optimization algorithm compares very favorably with the distribution of values under SRS. As a further comparison, we use stratified simple random sampling (StratRS) to choose another 100 designs. Following conventional cell-based approaches, the entire 6-D space is divided into $3^6 = 729$ cells, one design point is randomly selected from each nonempty cell, and then further points are randomly chosen to reach 729 points. The distribution of U values under StratRS also depicted in Figure 7 indicates that StratRS is preferable to SRS according to the UCC criterion, but the UCC optimization algorithm still performs considerably better. The SRS and StratRS distributions demonstrate that the simple strategy of

randomly sampling many designs and choosing the best according to the UCC criterion is a poor substitute for the optimization algorithm.

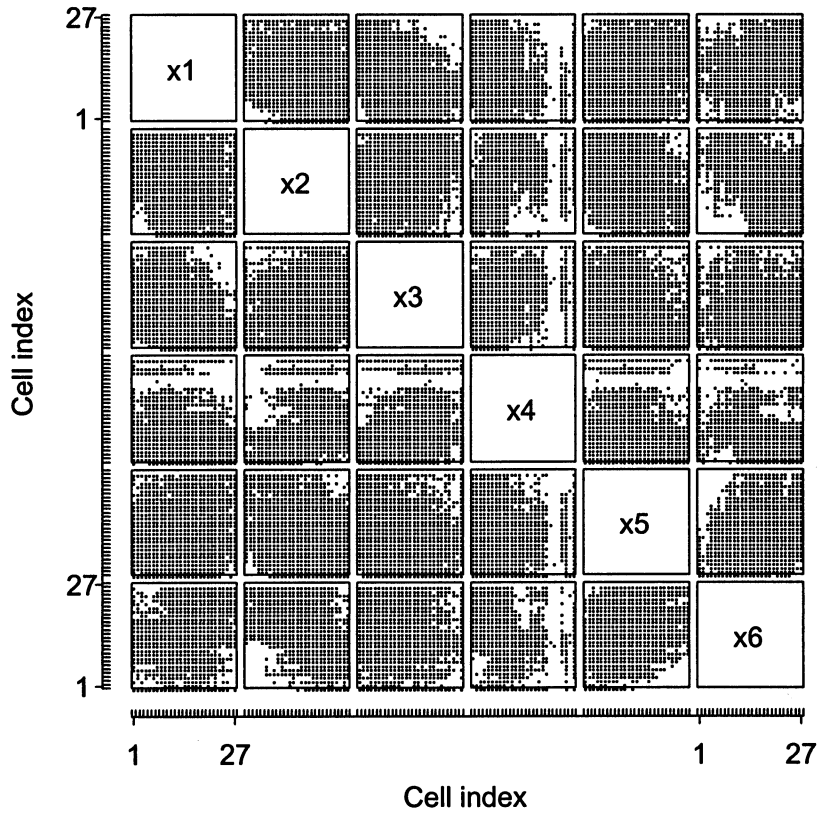

Figure 6. Candidate-Point 2-D Coverage for the NCI Data (729 Cells). A dot in a cell denotes at least one candidate point.

Table 1 gives some numerical summaries of these comparisons. For SRS and StratRS, the numbers given are means across the 100 random designs. We report the UCC criterion U in (3), the percent coverage criterion P in (2), and the 1-D, 2-D, and 3-D contributions to these two criteria. Note that U and its components are smaller the better measures, whereas P and its components are larger the better. In all cases, the design produced by the UCC optimization algorithm performs best. For example, P is 75% for

the design from our algorithm versus 53% on average under StratRS. Note also that $U_{3\text{-D}}$ is the largest contributor to U, probably because there are slightly more empty cells in the 3-D subspaces

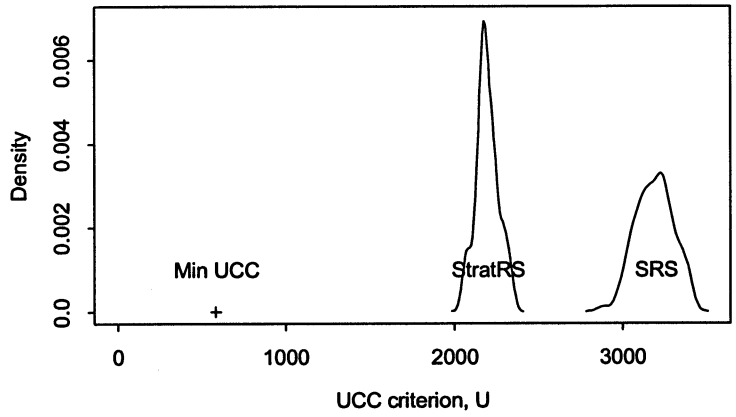

Figure 7. Distributions of UCC Values for 100 Simple Random Samples (SRS) and 100 Stratified Random Samples (StratRS), and the Value Obtained by the UCC Minimization Algorithm (+).

Design	UCC Criterion, U ($U_{1\text{-D}}$ / $U_{2\text{-D}}$ / $U_{3\text{-D}}$)	Average Percent Coverage, P ($P_{1\text{-D}}$ / $P_{2\text{-D}}$ / $P_{3\text{-D}}$)	Time (hh:mm)
UCC (Fast exchange algorithm)	585 (463/ 597/ 693)	75.1 (74.9/ 74.5/ 76.0)	0:25
Simple Random Sampling (mean over 100 designs)	3188 (2035/ 2937/ 4592)	45.5 (49.9/ 44.8/ 41.9)	0:01
Stratified Random Sampling (mean over 100 designs)	2193 (1979/ 2050/ 2551)	53.4 (53.4/ 52.5/ 54.4)	0:02
UCC (Basic exchange algorithm)	606 (489/ 615/ 714)	74.5 (73.8/ 74.2/ 75.5)	12:19
SAS PROC OPTEX Spread Design (Sequential)	3839 (7122/2365/2031)	59.4 (55.1/ 59.0/ 64.1)	2:32
SAS PROC OPTEX Uniform Coverage Design (Sequential)	3556 (2255/ 3107/ 5306)	45.4 (50.4/ 45.1/ 40.8)	133:29

Table 1. Coverage Criteria and Run Times for Various Designs

Figure 8 compares the UCC design with the first StratRS design in terms of cell frequencies in 1-D projections. Descriptor x_1's candidates are fairly well behaved after hybrid binning; in contrast x_4's distribution is more difficult to handle. In both cases the UCC design is seen to have a much more uniform distribution of design points. For all descriptors, the UCC design has one or two design points in most cells. Some analogous 2-D projections of the design points are shown in Figure 9, where a dot is plotted in a cell if there is at least one design point. It is clear that the UCC design has superior coverage of 2-D cells. Similar plots for the 3-D projections show the same pattern.

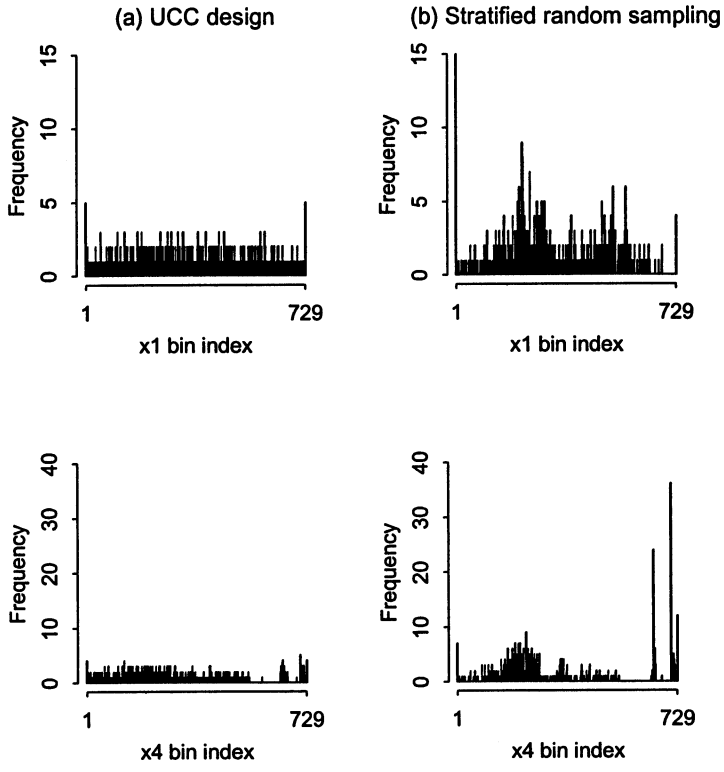

Figure 8. Design-Point Bin Frequencies for the NCI data (729 Points in 729 Hybrid Bins).

6.3. Comparison With Basic Exchange

Compared with the basic exchange algorithm, our algorithm makes about three times as many exchanges (2308 versus 754). It achieves this even though the number of passes through the candidate points is

reduced by a factor of about 50 (15.2 passes versus 754). The reduction in iterations through the candidates leads to a run time of less than half an hour, whereas the basic exchange algorithm takes more than 12 hours (with the fast UCC update described in Section 5.4). These times relate to implementations in SAS PROC IML on a Pentium III 550MHz computer with 256MB RAM. Some preliminary runs with a C++ implementation indicate that the modified algorithm runs in less than a minute for problems of this magnitude.

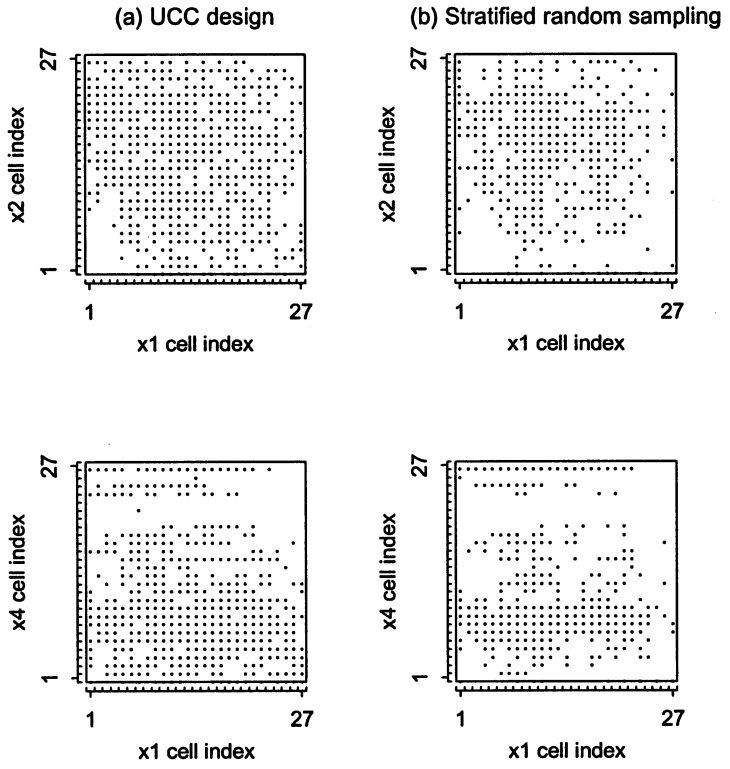

Figure 9. Design-Point 2-D Coverage for the NCI Data (729 Cells). A dot in a cell denotes at least one candidate point.

In Table 1 we see that our fast exchange algorithm produces slightly better values of P and U here than does the basic exchange method. The increase in number of exchanges, including neutral exchanges, improves the ability to escape from local minima. Similar results were obtained with another database from GlaxoSmithKline.

6.4. Comparison With SAS PROC OPTEX

We also make comparisons with PROC OPTEX in SAS. There are two difficulties. First, our criteria focus on low-dimensional coverage whereas PROC OPTEX computes spread and coverage measures using all descriptors. Therefore, the U and P values reported in Table 1 for PROC OPTEX are unfavorable. Secondly, problems of this magnitude (729 points selected from 29,812) require substantial computing time. Even when design points are optimized one at a time in the sequential option, PROC OPTEX with the uniform coverage criterion requires over five days.

7. CONCLUSIONS AND DISCUSSION

Our design problem is somewhat special for the following reasons. The candidate set of possible explanatory variable combinations is discrete, because only certain compounds can be made. Moreover, the set of discrete points can be large and highly irregular (see, for example, Figure 2). To have similar properties, it is believed that two compounds must have very similar values of all critical descriptors. Thus, the design needs to cover the space densely. It is clearly impossible to achieve dense coverage in more than three dimensions at a time without an extraordinarily large design. Hence, we have proposed designs that aim for uniform coverage in all 1-D, 2-D, and 3-D projections.

The aim of such experimental designs is not just to discover highly active compounds but to find several structurally different chemical classes. These provide options for further optimization of activity, physical properties, distribution, half-life, toxicity, etc. By covering the descriptor space uniformly, there is more chance of discovering multiple classes.

The design algorithm proposed here can efficiently deal with tens of thousands of compounds in the candidate set. Much larger sets of compounds will be of interest as technology advances. We are currently working to implement the algorithm with multiple processors, for example.

An open question is how to analyze the data resulting from very large designs. Current practice often simply ranks the compounds by potency and selects the few top-ranking compounds for further development. One challenge in statistical modeling is that the potent molecules are likely to be acting in several different ways: Different descriptors might be critical for the various mechanisms. A single mathematical model is unlikely to work well for all mechanisms. There has been some success using partitioning methods on these problems (e.g., Hawkins et al. 1997, King et al. 1992, and Klopman 1984). In a multi-stage design strategy, the initial design should cover the descriptor space as uniformly as possible. Analysis of the resulting data would be used to directing subsequent designs to subregions of high activity in critical descriptor projections.

ACKNOWLEDGMENTS

We are grateful for the help of Eugene Stewart in providing the chemical descriptors. Welch's research was funded by NSERC of Canada. We thank the editor, associate editor, and two referees for numerous suggestions which clarified the presentation.

REFERENCES

Box, G.E.P., Hunter, W.G., and Hunter, J.S. (1978), *Statistics for Experimenters*, New York: Wiley.

Burden, F.R. (1989), "Molecular Identification Number for Substructure Searches," *Journal of Chemical Information and Computer Sciences*, 29, 225-227.

Cook, R.D., and Nachtsheim, C.J. (1980), "A Comparison of Algorithms for Constructing Exact D-Optimal Designs," *Technometrics*, 22, 315-324.

Cummins, D.J., Andrews, C.W., Bentley, J.A., and Cory M. (1996), "Molecular Diversity in Chemical Databases: Comparison of Medicinal Chemistry Knowledge Bases and Databases of Commercially Available Compounds," *Journal of Chemical Information and Computer Sciences*, 36, 750-763.

Dalal, S.R., and Mallows, C.L. (1998), "Factor-Covering Designs for Testing Software," *Technometrics*, 40, 234-243.

Doehlert, D.H. (1970), "Uniform Shell Designs," *Applied Statistics*, 19, 231-239.

Fang, K.T., Wang, Y., and Bentler, P.M. (1994), "Some Applications of Number-Theoretic Methods in Statistics," *Statistical Science*, 9, 416-428.

Hawkins, D.M., Young, S.S., and Rusinko III, A. (1997), "Analysis of a Large Structure-Activity Data Set Using Recursive Partitioning," *Quantitative Structure-Activity Relationships*, 16, 296-302.

Higgs, R.E., Bemis, K.G., Watson, I.A., and Wike, J.H. (1997), "Experimental Designs for Selecting Molecules from Large Chemical Databases," *Journal of Chemical Information and Computer Sciences*, 37, 861-870.

Johnson, M.E., Moore, L.M., and Ylvisaker, D. (1990), "Minimax and Maximin Distance Designs," *Journal of Statistical Planning and Inference*, 26, 131-148.

Kennard, R.W., and Stone, L.A. (1969), "Computer Aided Design of Experiments," *Technometrics*, 11, 137-148.

King, R.D., Muggleton, S., Lewis, R.A., and Sternberg, M.J.E., (1992), "Drug Design by Machine Learning: The Use of Inductive Logic Programming to Model the Structure-Activity Relationships of Trimethoprim Analogues Binding to Dihydrofolate Reductase," *Proceedings of the National Academy of Sciences*, 89, 11322-11326.

Klopman, G. (1984), "Artificial Intelligence Approach to Structure-Activity Studies. Computer Automated Structure Evaluation of Biological Activity of Organic Molecules," *Journal of the American Chemical Society*, 106, 7315-7321.

McFarland, J.W., and Gans, D.J. (1986), "On the Significance of Clusters in the Graphical Display of Structure-Activity Data," *Journal of Medicinal Chemistry*, 29, 505-514.

Menard, P.R., Mason, J.S., Morize, I., and Bauerschmidt, S. (1998), "Chemistry Space Metrics in Diversity Analysis, Library Design, and Compound Selection," *Journal of Chemical Information and Computer Sciences*, 38, 1204-1213.

Owen, A.B. (1992), "Orthogonal Arrays for Computer Experiments, Integration, and Visualization," *Statistica Sinica*, 2, 439-452.

Tang, B. (1993), "Orthogonal Array-Based Latin Hypercubes," *Journal of the American Statistical Association*, 88, 1392-1397.

Tobias, R. (1995), *SAS QC Software. Volume 1: Usage and Reference*, Cary, N.C.: SAS Institute.

Wynn, H.P. (1972), "Results in the Theory and Construction of D-Optimum Experimental Designs," *Journal of the Royal Statistical Society* Ser. B, 34, 133-147.

Young, S.S., Farmen, M., and Rusinko III, A. (1996), "Random Versus Rational: Which is Better for General Compound Screening?" *Network Science*, http://www.netsci.org/Science/Screening/feature09.html.

Zemroch, P.J. (1986), "Cluster Analysis as an Experimental Design Generator, With Application to Gasoline Blending Experiments," *Technometrics*, 28, 39-49.