Published in final edited form as:
J Chem Inf Model. 2023 March 27; 63(6): 1776–1793. doi:10.1021/acs.jcim.2c01579.

To Design Scalable Free Energy Perturbation Networks, Optimal Is Not Enough

Mary Pitman[†], **David F. Hahn**[‡], **Gary Tresadern**[‡], **David L. Mobley**[*,†,¶]

[†]Department of Pharmacy & Pharmaceutical Sciences, University of California, Irvine, CA 92697, USA

[‡]Computational Chemistry, Janssen Research & Development, Turnhoutseweg 30, Beerse B-2340, Belgium

[¶]Department of Chemistry, University of California, Irvine, CA 92697, USA

Abstract

Drug discovery is accelerated with computational methods such as alchemical simulations to estimate ligand affinities. In particular, relative binding free energy (RBFE) simulations are beneficial for lead optimization. To use RBFE simulations to compare prospective ligands *in silico*, researchers first plan the simulation experiment, represented by graphs where nodes represent ligands and graph edges represent alchemical transformations between ligands. Recent work demonstrated that optimizing the statistical architecture of these perturbation graphs improves the accuracy of the predicted changes in the free energy of ligand binding. Therefore, to improve the success rate of computational drug discovery, we present the open-source software package High Information Mapper (HiMap) — a new take on its predecessor, Lead Optimization Mapper (LOMAP). HiMap removes heuristics decisions from design selection and instead finds statistically optimal graphs over ligands clustered with machine learning. Beyond optimal design generation, we present theoretical insights for designing alchemical perturbation maps. Some of these results include that for n number of nodes, the precision of perturbation maps is stable

[*] dmobley@uci.edu .
Article and Author Information
Mary Pitman
Contribution: Conceptualization, Software, Investigation, Data curation, Formal analysis, Investigation, Visualization, Methodology, Writing - original draft, Writing - review and editing.
David F. Hahn
Contribution: Conceptualization, Data curation, Writing - review
Gary Tresadern
Contribution: Conceptualization, Writing - review
David L. Mobley
Contribution: Conceptualization, Supervision, Funding acquisition, Writing - original draft, Writing - review and editing.

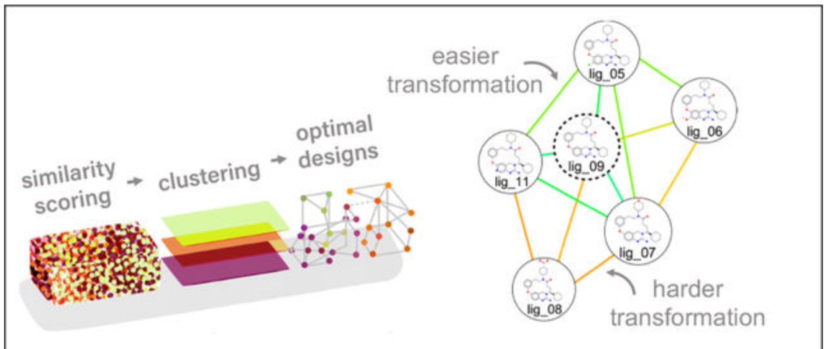
Competing Interests
DLM serves on the scientific advisory boards of OpenEye Scientific Software and Anagenex, and is an Open Science Fellow with Roivant Therapeutics. Aside from that, the authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Supporting Information Available
The following files are available free of charge.

- Figure S1–3 for analysis of seed type for optimization, HiMap clustering analysis, and BACE1 ligand optimal perturbation maps. Tables S1–3 for efficiency data and to compare optimal maps of BACE1 ligands (PDF).
- SMILES strings of BACE1 ligands used for optimal map planning are available in SMILES.csv (CVS).

at $n \cdot \ln(n)$ edges. This result indicates that even an 'optimal' graph can result in unexpectedly high errors if a plan includes too few alchemical transformations for the given number of ligands and edges. And, as a study compares more ligands, the performance of even optimal graphs will deteriorate with linear scaling of the edge count. In this sense, ensuring an A or D-optimal topology is not enough to produce robust errors. We additionally find that optimal designs will converge more rapidly than radial and LOMAP designs. Moreover, we derive bounds for how clustering reduces cost for designs with a constant expected relative error per cluster, invariant of the size of the design. These results inform how to best design perturbation maps for computational drug discovery and have broader implications for experimental design.

Graphical Abstract



High Information Mapper (HiMap) plans optimal ligand binding free energy perturbations.

Introduction

Accurate computational methods can substantially accelerate early-stage drug discovery. However, drugs commonly have dissociation rate constants of 10^{-1} to $10^{-6} s^{-1}$,[1] making it impractical to directly simulate binding and unbinding events with conventional molecular dynamics (MD). These slow unbinding events make alternative routes for affinity estimation appealing. In particular, so-called alchemical simulations provide a method to compute free energy differences between the bound and unbound states. Since free energy is a state function, we can measure changes in free energy with alternative binding and unbinding pathways via non-physical ("alchemical") intermediates. Ultimately, suitable alchemical simulations measure the change in Gibbs free energy, ΔG, which for binding is related to ligand affinity. Alchemical simulations can then avoid some of the slow sampling associated with direct MD, making alchemical methods an attractive tool for drug discovery.

While alchemical simulations bypass slow binding and unbinding events, they still can encounter other sampling limitations. For example, one type of alchemical transformation, done using what are called absolute binding free energy (ABFE) calculations, directly computes ΔG using the reversible work of decoupling a ligand from its binding site and then recoupling it in bulk solvent.[2] A sampling limitation of ABFE methods is that the simulation of the bound and unbound states requires the target to relax to its preferred state both in the presence and absence of the ligand, which can be slow. Alternatively, relative binding free energy (RBFE) simulations compare pairs of ligands, A and B, to measure $\Delta\Delta G_{A \to B}$. In the RBFE method, ligand A is progressively morphed into B in both the binding pocket and solvent, reducing the time required for macromolecular rearrangements since the receptor may always stay in a conformation consistent with ligand binding. However, larger RBFE transformations may be difficult and less reliable due to sampling problems such as charge changes,[3] ring breaking,[4] and atom insertions,[5,6] while more minor perturbations improve convergence speed.[7,8]

To compare a series of ligands using RBFE, researchers typically make a map or graph of planned transformations, where nodes are ligands and connecting edges are alchemical transformations between nodes. For the highest possible confidence when accounting for statistical errors, the resulting map would include all possible edges in the limit where all transformations are equally alchemically feasible. Graph redundancy can then be used to detect and correct for statistical error. However, the computational cost of highly redundant designs could become prohibitive as the number of nodes, n, increases since each RBFE calculation typically incurs substantial costs.

Here, we focus on planning RBFE calculations, an area where the field typically uses one of several tools to plan maps. Typical tools use a heuristic decision-making process. For example, the tool Lead Optimization Mapper (LOMAP) tries to find the minimal number of edges where every node is in at least one cycle and edges are removed from regions of the graph with many connections.[8] This design strategy results in maps with an edge count typically between $(1n, 2n)$. The OpenEye tool, OE Mapper, also follows this design goal and minimizes the edge number while maintaining cycles.[9]

Some design approaches entirely avoid solving the challenge of design selection by choosing maps with a fixed topology,[10] sacrificing potential accuracy in favor of simplicity. One such design scheme employs a radial design where one central ligand, often with a reference affinity, is connected to each other ligand in the set by a single edge. These minimally spanning radial designs have $n - 1$ edges. While radial designs have the shortest distance between all ligands and the reference ligand to decrease computational cost, there is no redundancy in the design and no possibility of error correction. Radial designs also assume that transformations from all ligands to the reference should be equally represented in the overall topology or that all transformations are equally challenging to perform, and can deliver poor results if the chosen reference proves problematic.

Overall, while both LOMAP and the simple radial approach yield maps of planned simulations, these approaches do little to quantify anticipated performance and may yield higher error predictions in the free energy change relative to more statistically optimal designs.

Planning an ideal map of RBFE calculations is challenging because the number of possible designs grows rapidly. We will denote a design with n number of nodes and k number of edges as $\mathcal{G}(n, k)$. The number of possible edges, k_{full}, that we pick between to form a design is

$$k_{full} = \binom{n}{2} = \frac{n(n-1)}{2}\,. \tag{1}$$

The number of designs within the set, $|\mathscr{G}(n,k)|$, is the number of unique combinations of edges,

$$|\mathscr{G}(n,k)| = \binom{k_{full}}{k} = \frac{k_{full}!}{k!(k_{full}-k)!}\,. \tag{2}$$

To illustrate the scaling of the design space, for a map that connects 20 ligands ($n = 20$) by 95 edges ($k = k_{full}/2$), there are $\approx 10^{55}$ designs to select from. These values apply to an undirected graph, meaning that the calculation is equivalent in each direction. Heuristics and intuition are unlikely to produce statistically optimal designs at this scale; we instead need an information-driven model to steer the design process. Thus, to decide what edges to simulate, we should design experiments that logically allocate resources to preserve precision.

To improve perturbation map planning, we present the High Information Mapper, HiMap,[11] a software package that generates optimal perturbation maps and clusters ligands into similar groups via unsupervised machine learning. We use the term *optimal* to mean the mathematical process of finding the solution that optimizes a statistical design criterion (further described in Background), rather than to mean that these graphs are *best*, colloquially. The optimization process can steer designs toward solutions that aid in the reduction of error. Indeed, optimizing the statistical architecture of RBFE perturbation maps dramatically improves the accuracy of free energy estimates.[12,13]

We will also present our theoretical findings on the planning of designs. These results are integral to the features included in HiMap and should inform design decisions. For example, we show how the number of edges in a design should scale with the number of ligands to avoid accumulation of errors. To our knowledge, the question of how k should vary with n for alchemical designs is largely unexplored. So, we aim to provide both an accessible tool to users and theoretical insights that researchers can apply beyond the scope of HiMap.

Background

A free energy calculation approximates the unknown but true changes in free energy, $\mathbf{\Delta G^{True}}$ (with boldface indicating an array), of n ligands that occur upon some physical process such as binding or solvation. In our particular case, we simulate k transformations to measure relative changes in binding free energy, $\mathbf{\Delta\Delta G^{obs}}$, and collect known reference free energy values for one or more of these ligands, $\mathbf{\Delta G^{ref}}$, from experimental work. Our goal is to use the simulation results, reference values, and subsequent analysis to approximate $\mathbf{\Delta G^{True}}$ as accurately as possible.

To approximate the unknown but true changes in free energy, we need a tool to map $\mathbf{\Delta G^{True}}$ onto our observations. The mapping is similar to how a slope, m, relates x and y in the equation for a line, $y = mx + e$, where e is the error. Our multidimensional analogy to m that maps $\mathbf{\Delta G^{True}}$ onto our collection of observations and relative changes in free energy is a design matrix, $\mathbf{A}$. Each value in the block array of υ observations or calculated values, $\mathbf{\Delta G^{obs}} = \left[\mathbf{\Delta G^{ref}} \mathbf{\Delta\Delta G^{obs}}\right]^{\mathsf{T}}$, has an associated, unknown error propagated from multiple sources, $\mathbf{e_{\Delta G}}$. We assume this error vector is multivariate normal so that each observation is drawn from a normal distribution centered on the true change in free energy. If ΔG_i^{obs} is equal to the true but unknown ΔG_i^{True}, the value of $e_{\Delta G,i}$ is equal to zero. Using block matrix notation, the mapping is

$$\underset{\upsilon \times 1}{\begin{bmatrix} \mathbf{\Delta G^{ref}} \\ \mathbf{\Delta\Delta G^{obs}} \end{bmatrix}} = \underset{\upsilon \times n}{\begin{bmatrix} \mathbf{A^{\Delta G}} \\ \mathbf{A^{\Delta\Delta G}} \end{bmatrix}} \underset{n \times 1}{\mathbf{\Delta G^{True}}} + \underset{\upsilon \times 1}{\mathbf{e_{\Delta G}}}\,. \tag{3}$$

The elements of the design matrix block that defines the reference ligands, $A_{ij}^{\Delta G}$, are either 0 or a value of 1 at the column of the reference ligand row in $\mathbf{\Delta G^{True}}$. The elements of the design matrix block for the relative free energy calculations, $A_{ij}^{\Delta\Delta G}$, has a value of 1 to specify a perturbation's final state, ΔG_f, and −1 for the initial state, ΔG_i, so that $\Delta\Delta G = \Delta G_f - \Delta G_i$; unconnected ligands have a value of 0. The mapping performed by $\mathbf{A}$ in eq 3 is therefore a graph that defines what ligands (nodes) that transformations (edges) connect. Moreover, $\mathbf{A}$ is a matrix representation of an unweighted perturbation map. Solving for the error vector, eq 3 can be rewritten in the form $e = y - mx$ as

$$\mathbf{e_{\Delta G}} = \mathbf{\Delta G^{obs}} - \mathbf{A\Delta G^{True}}\,. \tag{4}$$

With some caveats, if $\mathbf{A}$ has at least one reference value and $n - 1$ perturbations, there exists a unique regression solution that minimizes the total length of $\mathbf{e_{\Delta G}}$.

We must first decide how to model the data to find the regression that minimizes the error vector in eq 4. The error vector after regression will have a residual variance of σ^2. For an unweighted graph, we could solve eq 4 with linear regression provided that the values in $\mathbf{\Delta G^{obs}}$ meet a few conditions. For example, the data must have linear correlation meaning that there is a linear statistical association between the dependent variable and the parameters of the regression.[14] Linear regression also assumes normal or Gaussian distributed errors.[14,15] In addition, there must be low expected covariance magnitudes for our predictions of the changes in free energy.[16] our case, the covariance criterion does not hold because the final changes in free energy depend on a series of pairwise differences, so the final ΔG values that we will solve for depend on one another. Linear regression also requires *homoscedasticity*, meaning that the variances for our predictions are constant.[17] Recall that some alchemical transformations perform better than others.[3–8] As a consequence, our data does not satisfy the homoscedasticity condition for unweighted linear regression. Accordingly, we assign edge weights with the matrix $\mathbf{W}$ based on a transformation's anticipated complexity and chemical similarity (Software Methods). Entries in $\mathbf{W}$ corresponding to edges range from 0 to 1, and additionally include weights for the reference ligand which are assigned a weight of 2.[13] This choice of a weighted graph to model the experimental design means we use weighted linear regression to minimize $\mathbf{e}_{\Delta\mathbf{G}}$.

Finding the minimum possible $\mathbf{e}_{\Delta\mathbf{G}}$ with weighted regression is not as simple as our $y = mx + e$ analogy. We can instead use a derived result from maximum likelihood estimation[12,18] to solve for the changes in free energy, $\Delta\hat{\mathbf{G}}$, that best estimate $\Delta\mathbf{G}^{\mathbf{True}}$. The weighted linear regression solution is[13]

$$\Delta\hat{\mathbf{G}} = \left(\mathbf{A}^{\top}\mathbf{W}^{-1}\mathbf{A}\right)^{-1}\mathbf{A}^{\top}\mathbf{W}^{-1}\Delta\mathbf{G}^{\mathbf{obs}}. \tag{5}$$

Notice that eq 5 no longer uses $\Delta\mathbf{G}^{\mathbf{True}}$ as input, since the true free energies are not known. We can fully describe the most likely solution by weights, the design topology, and observations.

The design information and weights can also help predict how an edge will impact the solutions for $\Delta\hat{\mathbf{G}}$ in eq 5. For weighted regression, we can describe the impact of an edge on the regression using the hat matrix, $\mathbf{H}$,[19] where

$$\mathbf{H} = \mathbf{A}\left(\mathbf{A}^{\top}\mathbf{W}^{-1}\mathbf{A}\right)^{-1}\mathbf{A}^{\top}\mathbf{W}^{-1}. \tag{6}$$

Using the cyclic property of trace[20] and from 6,

$$\operatorname{Tr}\mathbf{H} = \operatorname{Tr}\left[\left(\mathbf{A}^{\top}\mathbf{W}^{-1}\mathbf{A}\right)^{-1}\mathbf{A}^{\top}\mathbf{W}^{-1}\mathbf{A}\right]. \tag{7}$$

Since $\mathbf{A}^{\top}\mathbf{W}^{-1}\mathbf{A}$ produces an $n \times n$ matrix where the inverse exists, 7 simplifies to summing the diagonal elements of the n identity matrix, $\mathbf{I}_n$, expressed as $\operatorname{Tr}\mathbf{I}_n$. By definition, the n diagonal elements of $\mathbf{I}_n$ are 1,[20] and thus

$$\operatorname{Tr}\mathbf{H} = n \tag{8}$$

The expression in 8 sums the diagonal values of the hat matrix, H_{ii}, called the leverages of the edges. Leverage values are positive with a maximum value of 1.[16] We focus on the diagonal elements of $\mathbf{H}$ here because these values are commonly used for outlier detection for robust regression analysis.[16] A leverage value of 1 is a potentially influential point on the regression and may be an x-axis outlier.[16] Further, as the leverage of a point approaches 1, the residual of that point decreases. A high-leverage point is problematic if the exclusion of the point dramatically changes the the linear model. Using eqs 5 and 6, the fitted relative changes in free energy, $\Delta\hat{\mathbf{G}}^{\mathbf{obs}}$, are $\Delta\hat{\mathbf{G}}^{\mathbf{obs}} = \mathbf{H}\Delta\mathbf{G}^{\mathbf{obs}}$. Consequently, leverages describe the potential for a particular observation to influence its prediction in $\Delta\hat{\mathbf{G}}^{\mathbf{obs}}$.

While we can apply eq 5 after simulation to solve for the changes in free energy, we can also use the information about uncertainties to engage in design and find a set of transformations that are likely to decrease error. The added information we need about potential designs is the covariance matrix of the parameter errors, **C**, which describes the free energy changes' variances, C_{ii}, and covariances, C_{ij} where $i \neq j$. To find **C**, we use

$$\mathbf{C} = \left(\mathbf{A}^{\top}\mathbf{W}^{-1}\mathbf{A}\right)^{-1}\sigma^{2}. \tag{9}$$

where σ is a constant standard error that scales the varied weight values in **W** and σ^2 is proportional to the variance of the error in $\mathbf{e}_{\Delta G}$ in eq 3 and 4.[13] The variance of the error in $\mathbf{e}_{\Delta G}$ on average decreases as simulation time increases. The variances and covariances predict *a priori* the best the experimental design can estimate $\mathbf{\Delta G}^{\mathbf{True}}$, given the model assumptions. See the Cramér-Rao bound[15] for expanded details. We can also describe the design information by the Fisher information matrix, $\mathscr{F}$, which under standard assumptions is the inverse of eq 9.[21] Ultimately, with $\mathscr{F}$ or **C**, we can forecast results, model precision versus cost, and design better experiments.

We use the framework described above to iteratively find a weighted graph that reduces the uncertainty in simulation predictions. For the optimization, we hold the edge count fixed and sample connected graphs formed from edges in k_{full} (eq 1). For each iteration at constant node, n, and edge count, k, if numeric criteria of **C** decrease, we accept the design $\mathscr{G}_i(n, k)$ (Software Methods). The criteria we use to find an optimal graph, $\mathscr{G}*$, that minimizes the statistical error for any given simulation time are

$$\mathscr{G}* = \begin{cases} \min\left(\det \mathbf{C}\right) = \min \prod_{i=1}^{n} \lambda_i\left(\mathbf{C}\right) & \text{if D-optimal} \\ \min\left(\operatorname{Tr} \mathbf{C}\right) = \min \sum_{i=1}^{n} \lambda_i\left(\mathbf{C}\right) & \text{if A-optimal} \end{cases}, \tag{10}$$

where $\lambda_i(\mathbf{C})$ are the eigenvalues of **C** with multiplicity. The optimality criteria depend only on $\lambda_i(\mathbf{C})$ because of the orthogonal invariance of the 2-norm of **C**.[22] From eq 10, the A-optimal design has the lowest sum of variances and is responsive to a reduction in the arithmetic mean of the parameter uncertainties. For A-optimal designs, the total variance is reduced by more connections to the highest similarity ligands (Results), often resulting in a branched topology.[13] The D-optimal design responds to a reduction in the geometric mean of the parameter uncertainties. This promotes less extreme values of $\lambda_i(\mathbf{C})$ and increases cycles (Results). In some cases, multiple degenerate designs could satisfy the conditions in eq 10. So, we use $\mathscr{G}*$ to mean the solution found through optimization. Optimization can be repeated from unique seed designs to select the optimal solution. With repeated runs from random seed designs, the Fedorov Exchange algorithm typically finds the global optimum for A and D-optimal designs.[23]

Software Methods

HiMap has three main parts: similarity scoring, clustering, and design optimization (Figure 1A).

Similarity Scoring

First, HiMap finds designs where edges are weighted based on a generalized similarity metric. Within HiMap, an edge with a high weight represents a relatively small perturbation between a ligand pair. Weights range from 0 to 1, where a value of 1 means the ligands are identical. To illustrate this, in Figure 1B, the transformation between ligands 7 and 8 is anticipated to be larger, or more complex, than a transformation from 8 to 9.

The similarity scores used by HiMap are customizable. As the default behavior, HiMap uses the similarity scores calculated by LOMAP based on chemical and methodological metrics.[8] HiMap can also read alternative metrics such as 3D shape information, Tanimoto scores, or ligand binding mode information. For example, a user performing RBFE ligand binding simulations could incorporate 3D binding mode information.

Alternatively, a user could use a solute surface area metric to plan solvation free energy simulations. While HiMap can accommodate numerous metrics, the chosen weighting metric should be related to the transformation error of the individual application.[17] By customization of the weighting metric, HiMap can be used for a broad array of optimization problems.

Clustering

The next step in HiMap is to perform clustering with density-based spatial clustering, DBSCAN (Figure 1A).[24] The clustering feature of HiMap automates the grouping of similar ligands to produce more efficient maps and improves software performance (see Results). DBSCAN is an unsupervised machine learning algorithm and is an optional step in the optimization workflow. We use DBSCAN instead of methods like k-means and hierarchical clustering because DBSCAN can identify noise data, and divides ligands into groups of arbitrary shape instead of assuming clusters are circular. Our input for DBSCAN is a 2D matrix of similarity scores, which is an array of n^2 values. Clustering in HiMap is proportionally very fast compared to design optimization. For more background on how DBSCAN compares to other clustering algorithms, refer to.[25] Clustering improves the automation of HiMap so that a user can plan mathematically justified perturbation maps with minimal manual curation.

While clustering is optional in HiMap, there are benefits to including this feature in the workflow. First, clustering improves optimization speed for designs with large n. In addition, clustering allows optimization to proceed without failure in some cases. For instance, one ligand might have no similarity to any other ligand in a set of ligands, which makes it impossible to numerically optimize the graph design with the methods used here. This failure point is a natural feature of the A and D-optimal design search: matrices must be invertible during optimization and must have a nonzero determinant (eq 10). Clustering prevents this failure point by removing dissimilar ligands from the set included in a perturbation map. Briefly, clustering results in multiple design matrices, which must only individually be invertible. In our example of an isolated dissimilar ligand, this ligand is disconnected from the other design matrices and optimization can proceed, albeit with this ligand left as an orphan. Other advantages of clustering for network planning can include: sorting ligands into congeneric series, reduction of perturbation map sizes, and cost savings. We will progressively expand upon and demonstrate these points throughout this work.

Clustering in HiMap requires a few inputs. One required input is the distance matrix which is calculated by HiMap using distance = 1 − similarity. Clustering by DBSCAN also depends on two additional parameters, the neighbor distance, ϵ, and the minimal number of samples that can define a cluster, min_s. HiMap automates the calculation of ϵ and reports a default value and a suggested reasonable range as described below (Figure 2A). For the min_s input parameter, the default value in HiMap is 2, so that clusters necessarily include an edge. A user can control both input parameters with keyword parameters.

Neighbor Distance Calculation

HiMap calculates ϵ values (illustrated in Figure 2B) via an automated scheme based on maximum curvature points in the distribution of distances. The points of maximum curvature are called ‘knees’ or ‘elbows,’ named based on the concavity at the extrema. For the remainder of this text, we will refer to these points as knees. Points of high curvature, or possible knees, are shown as plateaus in the difference curves in Figure 2A. In Figure 2A, y-values of the normalized curve correspond to potential ϵ values, and so we find the y-values of the normalized curve at the x-value of extrema of the difference curves. To calculate ϵ values for a unique distribution of distances, we use the Kneedle algorithm,[27] a general approach to ‘online’ and ‘offline’ knee detection. The online method searches through the whole array to find the global extrema, whereas the offline method returns the first extrema. To find the point of maximum curvature, we use the Kneedle method in the online mode[27] to find the global extremum of a polynomially fit curve (to reduce the effect of noise). This global extremum corresponds to the default ϵ calculated by HiMap, shown as the y-value where the black dotted line on Figure 2A, right, intersects the normalized curve.

The 2D array of distances for a set of ligands may have multiple points of high curvature or extrema in the difference curve. For instance, in Figure 2A, the left panel shows raw distance data with multiple possible knee points. One could imagine a case where a user wishes to subdivide a data set at a higher resolution — corresponding to a lower ϵ value — than the suggested default. Moreover, the polynomially fit curve does not resolve the local extrema seen in the raw data. To account for this, we also detect the first local extremum of the difference curve in the raw distance data; see the vertical dotted line in Figure 2A, left. Then, HiMap reports a range of ϵ values from the first local extremum in the raw data to the global extremum (given as the default value).

Neighbor Distance Sensitivity

The selection of the neighbor distance, ϵ, must be tailored to the individual dataset. A good choice of the numeric value of ϵ sets the clustering resolution and finds the point of diminishing returns for further clustering based on the distribution of distances.[27] We call the different natural breaks in the similarity matrix — which can serve to divide ligands into groups — *similarity substructures*. Similarity substructures can become visually apparent in a heatmap of distances, such as that shown in Figure 2C. The visual representation of distance data in the heatmap (Figure 2C) reveals diagonal block structures that could form groups of ligands. We would like to automatically cluster the groups of similar ligands indicated by these similarity substructures.

Indeed, we analyzed if similarity substructures were found by HiMap during clustering and found that the default ϵ produced distinct chemical groups. By plotting the distributions of similarity scores within clusters and between clusters (Methods and Materials), we found that ligands within clusters have higher similarity than between clusters (Figure S2A, B). And, further connecting clusters would require transformations between highly dissimilar ligands (Figure S2C).

Similar ligands can occur in arbitrary order in a dataset, so we developed additional visualization tools to aid in the selection of ϵ. When similar ligands are far apart in the heatmap, bands far away from the diagonal will be present, as seen in Figure 2C, from about mol_0 to mol_119. HiMap depicts the clustered regions calculated by DBSCAN at the chosen ϵ. In the cluster region plots in Figures 2D1–3, each color corresponds to a separate cluster, and black regions are noise points. These cluster region plots are output by HiMap with a horizontal alignment with the distance heatmap, as depicted in Figures 2C and D1–3. Since the number of clusters and noise points depends on ϵ, the cluster region plots provide a visual aid to adjust ϵ as needed.

There are practical benefits to the HiMap clustering algorithm. By selecting ϵ at knees, as seen with the default ϵ of 0.95 in Figure 2D3, we both avoid optimizing designs that include highly dissimilar nodes (which could break optimization, explained below) and preserve a low cluster number. For instance, a user could generate designs at an ϵ of 0.88 (Figure 2D2). However, the study could require added resources through ABFE simulations to connect designs or more experimental ligand data if a reference ligand with experimental data does not exist for a cluster.

Further, the automation mechanism in HiMap reduces the number of ligands treated as noise. The effect of reduced noise points with increasing ϵ is qualitatively shown in Figures 2D1–3 by the decrease in black colored regions. We further quantified the effect of varying ϵ in Figures 2E1–3. First, the calculated default ϵ is where the difference curve slope crosses the x-axis in Figure 2E1. As the neighbor distance decreases, the ratio of noise points becomes increasingly sensitive to ϵ until DBSCAN measures most of the ligands as noise (Figure 2E2). Specifically, the number of noise points shown in Figure 2E2 dramatically decreases as ϵ approaches the recommended default value. We also find that the cluster number increases with decreasing neighbor distance (Figure 2E3) until DBSCAN treats a high proportion of ligands as noise. So, the default value of ϵ in HiMap reduces the number of noise points in the dataset. Since each noise point will essentially be an orphaned ligand, this is a relevant design consideration.

After clustering, the user can select to optimize all clusters, only those containing a reference ligand, or any combination of specified clusters. Optimization is run either after clustering or as a stand-alone function of HiMap.

Optimization

Several tuneable input parameters control optimization. For example, the user may select the number of edges, k, for the designs. The minimum k is $n - 1$ to allow for a minimally connected design and the maximum number of edges is $(n - 1)/2$, k_{full}. If the requested edge number is outside the allowed range, HiMap sets k to the nearest allowed number.

The list of reference ligands also serves as an additional input for optimization. An input list of reference ligands is used to define **A** in eq 3. If no reference ligands are specified, the algorithm selects the ligand with the highest sum similarity to the rest of the set or cluster. The one or more reference ligands that are either specified or calculated by HiMap for each cluster are assigned a weight of 2 in **W** (eq 9), similar to the prior implementation in.[13] This higher weight increases the number of connections formed to the reference ligands.

Given the complexity of the procedure for setting up a graph optimization, HiMap has an interactive mode to walk the user through the required inputs.

Optimization Algorithm

We use the classical Fedorov exchange algorithm[28,29] for optimization, implemented in R and adapted from prior work.[13] To begin the Fedorov exchange process, we first generate a random design which is a seed design suitable for optimization. The random seed design, or what we will refer to as a 'random' design, is randomly generated until it passes a few tests. The random generation occurs by using the sample function from R[30] to randomly select k rows from an adjacency matrix of all possible edges, k_{full}. Next, the seed design's covariance matrix, **C**, and by extension the Fisher information matrix, $\mathscr{F}$, must be nondegenerate, meaning that no eigenvalues are equal to zero (see eq 10). Hence, we test if

$$\log(\det \mathscr{F}) \neq -\infty \tag{11}$$

to floating-point precision. Next, we test if $\mathscr{F}$ of the potential starting design is invertible. The seed design's $\mathscr{F}$ is not invertible if a ligand has zero or near zero similarity to the set. HiMap preserves the zero point score of ligands so that transformations that are not tractable for an alchemical method are never planned. However, dissimilar ligands that result in matrices that are not invertible can be filtered with clustering.

If HiMap does not find a random invertible seed design after 1000 iterations, a deterministic, minimally connected seed design is created. This minimally connected seed is found by first generating a radial graph, where all ligands excluding the reference are connected with $n - 1$ edges to a reference ligand. If $k > n - 1$, we calculate the number of edges to add, k_{add}. Then, the top k_{add} edge weights, excluding weights of radial edges, are added to the radial design. For a test set, we found that a random seed versus this nonrandom seed produced similar optimal designs both numerically and topologically (Figure S1).

With the data about designs collected and a seed design generated, the optimization proceeds via the mathematical framework presented above (Background). At the end of the optimization, HiMap outputs the final optimization criteria (eq 10), the leverages of the edges (eq 6), images of the final optimal designs, CSV files that specify the final edges and similarities, and JSON files of the clusters. Finally, the data frame of the optimal design and associated data can be imported from R into the python API for further manipulation.

Results

As designs become sparse, they become less suitable for optimization

While perturbation maps are ubiquitous for planning RBFE simulations, we still need to explore how the number of ligands and transformations (edges) in a study affect errors. Automated planners tend to minimize the number of edges, partially due to computational cost (Introduction). Consequently, to improve methods and develop flexible tools, we must answer the question: how does error change in response to the number of edges, k, and nodes, n? This line of reasoning will provide a model for the suggested use of HiMap and, more broadly, the design of scalable perturbation maps.

We begin the investigation into the relationship between n and k by examining design connectivity. First, we find that as designs become sparse, random seed designs more often have zero or near zero values of the determinant of the Fisher information matrix, $\mathscr{F}$. These seed designs are unsuitable as a starting point for optimization due to being weakly connected or disconnected. For a disconnected graph, at least one λ_i is precisely zero. Alternatively, in a connected graph, λ_i values and hence eq 10 may approach zero and result in a starting point for optimization that is not accurately invertible.[20] Recall that $\det(\mathscr{F})$ is the product of the inverse of $\lambda_i(\mathbf{C})$ (see eq 10). Since $\mathscr{F}$ is a positive semi-definite matrix for our designs, $\lambda_i \geq 0$. In addition, numerical methods on matrices required for optimization, such as inversion and Gaussian elimination, require a nonzero matrix determinant. Seed designs with low values of k relative to n are more often unsuitable for optimization because λ_i values become very small or are precisely zero.

To further quantify the effect of low values of k, we inspected the eingenspectra (arrays of eigenvalues) of $\mathscr{F}$. To perform this analysis, we studied a set of 20 ligands, taking the first 20 ordered ligands from the representative series analyzed in Figure 2. We then generated random designs until we found a solution with a nonzero determinant to floating-point precision. If we did not find a suitable design after 3×10^5 iterations, we terminated the search with the random design at the final iteration and then repeated this procedure three times. From this data, we find a progressive consolidation of $\lambda_i(\mathscr{F})$ towards zero as k decreases (Figure 3A). Interestingly, as $\lambda_i(\mathscr{F})$ becomes less balanced and more skewed towards zero, optimization becomes more difficult.[31] The progressive skew we find in Figure 3A could have implications for the optimization fidelity when there are relatively few edges.

Additionally, we find that if we increase k, the number of trials required to find a suitable seed design decreases (for constant n). Particularly, for 20 ligands, as k increases from 20 to 40, 60, and 80 edges, finding a suitable design takes an average of 3×10^5, 3×10^5, 72, and 7 trials, respectively. For 20 edges, two out of three of our trials resulted in a $\det(\mathcal{F})$ of precisely zero, meaning that the generated graph is disconnected. This observation leads us to ask next: given fixed n, for what values of k is a random design, $\mathcal{G}(n,k)$, likely to be connected?

The connectivity of designs has a logarithmic dependence on edge number

The literature previously addressed this question of connectivity for random graphs,[32,33] and we will provide the reader with a brief explanation to motivate our subsequent findings. We will use the phrase 'minimally connected' to refer to a design where a path exists between any pair of nodes. A connected component in a graph is a node or set of nodes that are linked to each other by a path. So, the number of connected components, c, for a minimally connected design is 1 and for a design without edges, c is equal to n. We provide an illustration of c in Figure 3B. To examine how connectivity changes with k for fixed n, we will consider the Erdős–Rényi model for random graph generation,[32,34] where edges are independently and randomly selected. The probability of randomly generating a design, $\mathcal{G}_i(n,k)$, for fixed n and k and with edge probability, p, is the geometric distribution:

$$P(\mathcal{G}_i(n,k)) = p^k(1-p)^{\binom{n}{2}-k}. \tag{12}$$

We will consider the case where we study the number of connected components, c, while adding edges to a previously empty design. Upon addition of one edge and then a second edge, c becomes $n-1$ and then $n-2$. Beyond this point, additional edges may or may not decrease c. The number of edges required to decrease c, k_c, can be summed as

$$k_{min} = \sum_{c=2}^{n} k_c, \tag{13}$$

where k_{min} is the number of edges added to form a minimally connected design. The sum in equation 13 starts at an index of 2 because k_c is the number of edges to transition from $c \to c-1$ and 1 is the lower bound for c. For a geometric distribution such as the one in eq 12, the expectation value for k_c is the inverse of the probability[35] that adding an edge decreases c. By linearity of expectation, it was then previously shown[36] that

$$\langle k_{min} \rangle \le \sum_{c=2}^{n} \frac{n-1}{c-1} = (n-1)H_{n-1}. \tag{14}$$

Equation 14 sums the reciprocals of the first $n-1$ positive integers and is the $(n-1)^{st}$ harmonic number, H_{n-1}. Harmonic numbers approximate the natural logarithm, meaning that the expected number of edges for a connected design scales as

$$\langle k_{min} \rangle \le n \cdot \ln(n). \tag{15}$$

From eq 15, the expected number of edges for a minimally connected design is less than or equal to $n \cdot \ln(n)$. Since this relationship is a lower bound, adding more edges than $n \cdot \ln(n)$ is unlikely to increase c. Below the lower bound in equation 15, as we observed in our experiment on random graph generation (Figure 3A), the probability of generating disconnected seed designs increases.

The derivation leading to eq 15 clarifies what number of edges are likely to produce minimally connected designs, but it does not yet address how connectivity changes for minimally connected designs. The gradual shift in eigenspectra of $\mathscr{F}$ for minimally connected designs (Figure 3A) is related to shifts in connectivity.[37,38] An example of a weakly connected design could be a design with only one perturbation per ligand, no cycles, and more edges on average separating ligands. For a weakly connected, weighted design, edges could also have low weights.

In the limit where it is feasible to perform all alchemical transformations accurately, we would like to plan our perturbation map so that k approximates the full design with all possible edges, $\mathscr{G}(n, k_{full})$. Two objects could be considered similar if they are close in distance to each other. If entries in arrays are farther in magnitude, two arrays are distant from each other and *vice versa*. But, objects such as vectors and matrices must have their lengths defined using a *norm*. To measure how different a design is from the the full design, we therefore calculate the matrix norm distance between the covariance matrix of the full design, $\mathbf{C}_{full}$, and the covariance matrices of randomly generated graphs at varied k, $\mathbf{C}_i$. We use the Frobenius norm to calculate distances (Methods and Materials, eq 26). To perform this analysis, we measure the distance to $\mathbf{C}_{full}$ for designs with varied edge count and with 10, 20, and 40 ligands. We find that as k decreases, $\mathbf{C}_i$ becomes increasingly further from $\mathbf{C}_{full}$ (Figure 4). The magnitude of the difference increases for designs with more ligands (Figure 4). We also find that the first and second moments in the matrix norm distance monotonically converge towards $\mathbf{C}_{full}$ as k approaches k_{full} (Figure 4). The rapid and monotonic decrease in the second moment of the matrix norm distance with increasing k means that the expected variability is inversely related to k. Further, once designs have on average $n \ln(n)$ edges, $\mathbf{C}_i$ starts to plateau towards $\mathbf{C}_{full}$ (Figure 4), suggesting that designs with $n \ln(n)$ edges reasonably approximate $\mathscr{G}(n, k_{full})$.

Perturbation maps have stable precision with logarithmic scaling of edges

We next hypothesize that increased design connectivity will improve ΔG precision and that the expected number of edges for a connected design (eq 15) may be a tipping point for how k should vary with n to control error. To test this hypothesis, we ran numerical optimization simulations (see Methods and Materials) for A-optimal, D-optimal, and 'random' designs over varied parameter sets of (n, k) with 200 replicates each (Figure 5A). Here, we label designs we randomly generate until we find a sufficiently connected solution *random designs* (defined in eq 11). We performed simulations for 10, 15, 20, and 30 nodes and edge numbers ranging from $n - 1$ to k_{full}. For varied n and k, the computational cost is not held fixed. From this analysis, we found a rapid increase in the expected mean squared error of ΔG, $\langle \text{MSE}(\Delta G) \rangle$, with decreasing edge count (Figure 5A). In Figure 5A we show the

$\langle \mathrm{MSE}(\Delta G) \rangle$ values over the 200 simulations as points along curves at constant value of n, or 'isonode' curves, plotted with interpolation. Importantly, $\langle \mathrm{MSE}(\Delta G) \rangle$ values worsen with increasing n when linearly scaling the edge count at $1n$ or $2n$ edges. This means that as n grows, a proportional increase in k is insufficient to maintain the precision of free energy estimates.

To further inspect the effect of edge number, we next calculated the error associated with designs at varied k with respect to the full design with all edges, $\mathscr{G}(n, k_{full})$, using the data in Figure 5A. We calculate the relative $\langle \mathrm{MSE}(\Delta G) \rangle$ for each value of k using

$$\langle \mathrm{MSE}(\Delta G) \rangle_{rel} = \langle \mathrm{MSE}(\Delta G) \rangle_{k} - \langle \mathrm{MSE}(\Delta G) \rangle_{k_{full}}. \tag{16}$$

From equation 16, we found a degradation in the precision of ΔG values with respect to that of $\mathscr{G}(n, k_{full})$ when testing edge counts linearly related to the number of edges – $1n$, $1.5n$, $2n$, and $3n$ (Figure 5B). Further, we found high and stable relative precision in the predicted changes in free energy for varied n at $k = n\ln(n)$ (Figure 5B).

With these results taken together, we conclude that as n increases, linear scaling of k causes an increasingly damaging effect on overall design precision as the resulting graphs grow more and more sparsely connected. So, for designs where $n \approx e^2$, traditional software and methods that choose edge counts near $2n$ do not suffer from a large loss in precision. However, it becomes increasingly important for larger designs $\left(n > > e^2\right)$ to consider the edge count and try to reach an edge number approaching $n\ln(n)$. If costs are prohibitive, each additional edge gained below $n\ln(n)$ edges is likely to improve the precision because k is less than that corresponding to the plateau regions in Figures 4 and 5A, B.

D-optimal **designs have the greatest number of cycles**

Over a closed thermodynamic cycle, free energies should sum to 0. However, over a closed cycle in FEP (illustrated in Figure 1B), errors may result in a nonzero sum, referred to as the hysteresis of the cycle. The theoretical cycle closure condition can be used as an informative constraint to solve for changes in free energy over the design.[12,39,40] A node may also be included in multiple cycles, allowing for further redundancy in free energy solutions. To analyze this, we quantified how A or D optimization changes the number of cycles in designs. A study may then select the optimization type that produces the most cycles if cycle closure corrections will be applied to solve for ΔG values.

The calculation of the number of cycles is an NP-complete problem, meaning that in practical terms, as n or k increase, the computational difficulty of finding the number of cycles becomes markedly higher or eventually intractable. So, to calculate the number of cycles in A-, D-, and random designs, we ran design optimizations over tractable combinations of (n, k), with five replicates per (n, k) (as compared to the 200 trials performed for MSE calculations). We ran simulations for the same set of (n, k) for each design type. In Figure 5C, we present the results for the number of cycles using surfaces plotted with triangular mesh interpolation. We found that D-optimization results in an increased number of cycles compared to A-optimal and random designs (Figure 5C). This trend that D-optimal designs produce more cycles was previously qualitatively noted.[13] Here, we provide a quantitative analysis in agreement with prior observed behavior.

Optimal designs are predicted to converge more rapidly than the reference designs

We now present the efficiency gains that HiMap achieves at constant n and k compared to radial designs and LOMAP 2.0. Since both LOMAP and radial designs have inflexible edge counts, we set k for the optimal designs to that of the reference design. For reference, LOMAP designs contained $1.2n$ to $1.5n$ edges compared to the $n - 1$ edge count of radial designs (Table S1).

For efficiency calculations, we compare the optimality criteria (eq 10) of reference designs and A and D-optimal designs. We calculate the weighted covariance matrices of the reference LOMAP and radial designs, $\mathbf{C}_{ref}$; A-optimal designs, $\mathbf{C}_A$; and D-optimal designs, $\mathbf{C}_D$, using eq 9. To calculate the relative efficiency of the reference designs to D-optimal designs, we take the ratio of the determinants of $\mathbf{C}$, where if

$$\det\left(\frac{\mathbf{C}_D}{\mathbf{C}_{ref}}\right) < 1 \tag{17}$$

the D-optimal design is measured as more efficient than the reference design (radial or LOMAP). To compare the A-optimal designs to the reference, if

$$\frac{\mathrm{Tr}\left(\mathbf{C}_A\right)}{\mathrm{Tr}\left(\mathbf{C}_{ref}\right)} < 1 \tag{18}$$

the A-optimal design is more efficient than the reference design. Conversely, if the ratios in eqs 17 and 18 are greater than 1, the reference design is measured as more efficient. We performed optimization in triplicate (Methods and Materials, Table S1).

Our results show that optimization provides a consistent improvement in the efficiency of designs compared to radial and LOMAP designs (Figure 6A). Indeed, the relative efficiencies we obtain are less than 1 for all five ligand sets (Figure 6A). These efficiency results indicate that optimal designs will converge more rapidly than the reference designs studied. And therefore, by investing additional effort into perturbation map planning, we can achieve computational cost savings due to improved efficiency even without increasing the number of edges in designs.

Clustering reduces cost at constant expected relative error per cluster

Clustering affects the overall cost of high-precision designs. With linear scaling of k, dividing nodes into smaller graphs does not result in cost benefits. In contrast, clustering does reduce costs for high precision designs because $n \cdot \ln(n)$ is *subadditive*. By subadditive, we mean that if we divide a group of ligands of size n into smaller groups and then plan designs for each cluster, the total number of edges for the whole set decreases.

However, the cost after clustering depends on how HiMap divides n into the set of clusters, $\{n_i\}$. We then solve for the total number of edges, k, over the number of clusters, d, such that

$$f(n_i) = \sum_{i=1}^{d} n_i \ln(n_i) = k\,. \tag{19}$$

We use Lagrange multipliers to find the extrema of eq 19 over continuous n and k under the constraints that the total number of nodes, n, is equal to

$$g(n_i) = \sum_{i=1}^{d} n_i = n \tag{20}$$

and that the minimum cluster size is

$$n_i \geq 2\,. \tag{21}$$

We find that the minimum cost and number of total edges occurs at

$$n_i = \frac{n}{d} \tag{22}$$

where the number of nodes in clusters are evenly balanced. Combining eqs 19 and 22, and defining the cost per edge as γ_{edge}, the minimum cost, γ_{min}, is

$$\gamma_{min} = \gamma_{edge} n \ln\left(\frac{n}{d}\right). \tag{23}$$

For the highest possible costs with clustering, the maximum number of edges, k_{max}, is equal to the sum of n_i in eq 19 where

$$n_i = \begin{cases} 2 & \text{if } 1 \leq i \leq d-1 \\ n-2d-2 & \text{if } i = d \end{cases}. \tag{24}$$

We provide this worst-case cost with clustering in the domain where the constraints in eqs 20 and 21 hold with eq 24. By combining eqs 19 and 24, the maximum cost, γ_{max}, is

$$\gamma_{max} = \gamma_{edge}\left[(d-1)\ln 4 + (n-2d+2)\ln(n-2d+2)\right]. \tag{25}$$

Thus, we can preemptively estimate the cost incurred with $n_i \ln(n_i)$ transformations per cluster to achieve high precision. The possible range of cost is from γ_{min} to γ_{max} as defined in eqs 23 and 25.

To graphically present our results in eqs 23 and 25, we plot the distributions of cost over continuous n and d. As a reference cost per edge, we use data from HPC benchmarking studies, which found that each RBFE simulation cost about $24/edge (Methods and Materials). For γ_{min} (eq 23) over varied n and d, we find a steep reduction in cost compared to unclustered designs even with minimal clustering (Figure 6B, top). On the other hand, for the most poorly balanced clusters (eq 24), clustering reduces cost but to a smaller extent than balanced clustering (eq 23) resulting in a more gradual decrease in cost in γ_{max} (Figure 6B).

These results highlight the practical cost benefit of either clustering ligands or subdividing ligands into smaller sets for RBFE studies. The cost bounds from eqs 23 and 25 can also be used to assess the financial risk of increasing the precision of designs.

D-optimal designs produce the most balanced leverages

We next inspect how individual transformations impact the accuracy of free energy predictions made using LOMAP 2.0 versus HiMap designs. We will use leverages to perform this analysis (Background). Briefly, leverages are the extent to which a particular observation influences its prediction in $\mathbf{\Delta\hat{G}^{obs}}$ (eq 6). Large and outlier leverage values can result in misleading binding free energy predictions. As a statistical test, leverage values two times larger than the average can be considered outliers.[16,19]

High leverage values can occur for a few reasons. For example, for our designs, edges to reference ligands have high leverage due to higher weighting (eq 6). Another example of a high leverage edge could occur if a node is connected via many transformations to the rest of the network and there is minimal redundancy or cycles; this would mean a single node serves as a critical link in the overall graph and any edge to that node might substantially affect overall predictions.

Initially, we compare A-optimal, D-optimal, and LOMAP designs generated with fixed n and k for each ligand set. The value of k that we use for each design is the number of edges in the LOMAP design. We find that the distribution of leverages, H_{ii}, of D-optimal designs is more peaked than A-optimal designs in all tested ligand sets (Figure 7A).

A-optimal designs favor high-leverage connections to the reference ligand, as seen in the higher density of leverage points near 1 (Figure 7A). From eq 8, the high leverage edges come at a cost where ligands far from the reference ligand are low leverage (Figure 7A) and present in few cycles (Figure 5C). As a consequence of the branched topology of A-optimal designs compared to the cyclic topology of D-optimal designs, the leverages of A-optimal designs are more varied — some edges are high-leverage, while other transformations weakly influence the regression. In contrast, D-optimal designs produce the lowest variance in H_{ii} meaning that each edge exerts a more even influence on the regression and the residuals are more uniform (see Background, eq 6). As a result, D-optimal designs may compensate for the anticipated varied errors of transformations and provide more even coverage of the experimental design space.

We find that LOMAP designs can achieve a similarly peaked distribution of leverages compared to D-optimal designs. We observe this similarity in the leverage distributions in Pfkfb3, Ptp1b, and Cdk2 ligands where k was restricted to the number of edges chosen by

LOMAP (Figure 7A). Both LOMAP and D-optimal designs favor cycles (Figure 5C,[8]). As a consequence, the leverages are balanced so that no subset of transformations exerts undue influence on the predictions for the changes in free energy.

We next compare D-optimal and A-optimal designs at varied k. We find a decrease in the probability of high leverage points and a trend towards reduced variance with increasing k (Figure 7B, C). Here, the control edge number is the number of edges output by LOMAP, shown in Figure 7A. Therefore, as k increases, any single edge is less likely to sway the solution of free energy values.

Example application of HiMap

We will now present a demonstration of HiMap clustering and design optimization using inhibitory ligands of the enzyme β-secretase 1, BACE1. The first step in HiMap is to calculate the similarity scores of the BACE1 ligands, where $n = 75$ (Figure 8A). For similarity scores, we used LOMAP 2.0 scoring which is the default in HiMap (Software Methods). Subsequently, to cluster the ligands, we used a neighbor distance cutoff of 0.4 which produced three distinct clusters, $\{d_i \mid 0 \leq i \leq 2\}$. HiMap clustering results in the division of ligands into logical groupings, based on difficulty of alchemical transformations. To illustrate this, macrocycle ligands are in cluster d_0, ligands with a six-member ring attached to a N-Phenylpyridine-2-carboxamide scaffold are in cluster d_1, and ligands with an additional five-member ring fused to the d_1 scaffold are in cluster d_2 (Figure 8A). The chemical structures of the ligands which are most similar to each cluster — lig_09, lig_22, and lig_76 — highlight the differences between the clusters detected by HiMap (Figure 8A). Due to ring breaking and opening, transformations between the clusters of BACE1 ligands would be challenging to perform with some RBFE methods.[10] For example, the open-source software PMX does not allow ring breaking.[41] Another study discussed ring breaking as highly challenging, but developed a protocol for ring breaking with AMBER.[42]

Upon further inspection, HiMap automatically divided the ligands into groups based on the original scaffolds from which the ligands were created. That is, d_0 ligands were derived from PDB ID: 2Q15.[43] The d_1 ligands were derived from ligands in PDB IDs: 3U6A,[44] 5CLM,[45] and 7D2X[46] which share a similar scaffold without insertions or ring size changes; and d_2 ligands were derived from PDB ID: 6OD66[47] with a heteroaryl-fused inhibitor. This division of the ligands supports the notion that HiMap rationally subdivides ligands into groups without manual editing by the user.

We next optimize designs for each cluster. For the designs, the ligand with the greatest sum similarity to the cluster is the reference ligand. We generated perturbation maps with $2n_i$ and $n_i \ln(n_i)$ edges for A and D-optimal designs, where n_i is the variable number of nodes for clusters in the set $\{d_i\}$. The A and D-optimal maps for d_0 at $2n$ edges are shown in Figure 8B, where the edge colors illustrate the edge weights. The perturbation maps for each cluster with $n \ln(n)$ edges are shown in Figure S3A. For the smaller ligand groups, d_0 with six ligands and d_2 with nine ligands, the A-optimal and D-optimal designs vary by only one and a few edges, respectively (Tables S2 and S3). For cluster d_2, A-optimal and D-optimal designs share a common core of 15 out of 19 edges. Notably, three edges exclusive to the

A-optimal design connect to the reference ligand, lig_76. While the D-optimal design lacks these connections to the reference, the edges exclusive to the D-optimal designs are of higher weight (Table S3).

The difference between A-optimal and D-optimal designs becomes more apparent for the largest cluster, d_1 (Figure S3A2). For d_1, the A-optimal design has 30 more edges than the D-optimal design to the reference ligand. However, the edges exclusive to the D-optimal design are overall higher weight with a gain in the average weight from 0.30 ± 0.09 for A-optimal to 0.5 ± 0.1 for D-optimal (Figure S3B2). The increased connections to the reference reduce the theoretical MSE of A-optimal designs compared to D-optimal designs (Figure 5A). But, the overall transformations that one must perform in A-optimal designs could be more challenging.

Discussion

Considerations when planning perturbation maps

In this work, we presented a new software package, HiMap, to plan alchemical perturbation maps and provided a theoretical framework to plan maps with improved precision and cost. Based on our findings, we propose the following practices when using HiMap or planning perturbation maps:

1. Increase the number of edges in designs to as close to $n \ln(n)$ as computational resources allow, since the total computational cost will increase. The map's number of edges, k, becomes increasingly important as the number of ligands, n, increases, in particular, where $\ln(n) > > 2$.
2. To compare alchemical predictions and methods, consider how to control for variable n and k in a computational study or experimental design. If the design size is left as an uncontrolled variable, changes in precision between maps may alter decisions on which ligands bind better or which methods perform better.
3. Use similarity score clustering to filter which ligands to group into maps. Alternatively, rationally subdivide ligands into smaller sets where possible.
4. To select for graphs with more cycles, use D-optimal designs. Use A-optimal designs to favor connections to reference ligands.

As a practical consideration for our first proposal, the subdivision of ligands into smaller groups, each with $n \ln(n)$ edges, reduces costs while maintaining precision (Figure 6B). There are additional costs if clusters do not have a suitable reference ligand. If a reference does not exist, either more experimental reference values need to be collected or an ABFE simulation performed. In addition, to aid users in increasing the number of edges per design, HiMap has a fully automated clustering algorithm (Software Methods). Clustering may be helpful because resources may limit the number of edges per design.

The ideal edge count we provide to control precision can be considered a lower limit to estimate the design with all edges (Figure 4). In other words, each added edge below $n \ln(n)$

is likely to improve overall results. Studies can be improved simply by increasing the edge count appropriately without adding more advanced simulation methods.

The second proposal has implications for methods development and application studies. Our results demonstrate that the relationship between n and k is an essential scientific control variable to eliminate confounding factors for a study. One way to control for the design size would be to keep n and k consistent across designs while using a similar planning method. Conversely, a study could account for differences in statistical certainty using $n\ln(n)$ scaling of the edge count. This control allows a researcher or reviewer to fairly compare the variabilities between methods, designs, or ligands. As noise increases, the effect of randomly varied n and k may not be detectable. Even in such a case, this proposal will help design experiments with fewer chances for variations in the scale of the perturbation map to contaminate scientific conclusions.

We note that edge count is one parameter of a study that can alter the error of predictions. There are additional variables in a study that could change prediction error. These factors include simulation length (though in one study only modest improvements in accuracy were reported from extending and even doubling simulation time[41]), the choice of lambda intermediates,[48] and structural features such as correct starting pose.[49] Therefore, edge count alone will not guarantee high precision and we make these conclusions in the limit of sufficient sampling and simulation protocol.

We suggest utilizing the clustering feature, our third proposal, in HiMap for two main reasons. Firstly, a study can run fewer transformations overall for high precision results when using clustering (Figure 6B) in the limit where reference ligands exist for each of the clusters chosen. Secondly, the rules that compose the overall similarity score in LOMAP also have veto power, meaning that a rule can result in a similarity score of zero. Without clustering or filtering before optimization, edges without similarity to the entire set cause optimization failure. For example, consider a group of ligands that can be divided into sets of alchemically similar ligands. We called these groupings similarity substructures, which are visible in the heatmaps output in HiMap (Figures 2C and 8A). Without clustering, a design may only weakly connect the similarity substructures via one or a few error-prone edges. The few edges connecting similarity substructures could require complex transformations such as a charge change, ring breaking, or a different binding mode. With clustering, the overall complexity of planned transformations is reduced. Alternatively, HiMap can optimize the design including all similarity substructures without clustering as long as sufficiently similar transformations exist so that eq 11 is satisfied.

Lastly, our fourth proposal helps users select what type of optimization to run. D-optimal designs provide more even coverage of the experimental design space and result in more cycles (Figure 5C). On the other hand, the metric used in A-optimization, the trace of the covariance matrix (Eq 10), is solely informed by the anticipated variances in $\mathbf{\Delta\hat{G}}$ values (Background). The minimization of variances results in A-optimal designs favoring more connections to the most similar ligands to the set and also produced the lowest average MSE of $\mathbf{\Delta\hat{G}}$ (Figure 5A, B). Since D-optimal designs produce more cycles whereas A-optimal designs favor a branched topology,[13] nodes are more likely to be present in at least one cycle in the D-optimal designs. This D-optimal design feature may allow more nodes to have corrections applied using cycle closure correction methods.[12,39,40] In addition, it has been shown that for graph optimization problems that use cycle closure constraints, subdividing nodes into more, smaller cycles reduces the accumulation of measurement error in cycles and decreases prediction errors.[50,51] So, future studies could investigate if D-optimal designs have an advantage over A-optimal designs for cycle closure correction methods.

We also note that A-optimal designs take longer than D-optimal designs to reach a solution in HiMap. Therefore, if the time for optimization is a concern, we recommend solving for the D-optimal design. Further, there are additional metrics that a study could use as criteria to perform graph optimizations.[12,22,52] We presented topological arguments to illustrate practical reasons to choose A versus D optimization, but there are also analytical arguments for criterion selection. [22,52]

Running diagnostics on designs

The optimal design, $\mathscr{G}*$, provides information on what transformations to perform but there is also additional Fisher information embedded in the design. It is this Fisher information that we used to perform many of the analyses presented here (Results). Indeed, $\mathscr{G}*$ is the solution that *a priori* maximizes information about the ligands true binding free energies, $\mathbf{\Delta G}^{\mathbf{True}}$.

After HiMap finds $\mathscr{G}*$, we can use the Fisher information for added insight. One such tool is the metric called the leverage (eq 6). A study could use the leverages output by HiMap (Figure 7) to triage potential outliers and diagnose issues that could overly influence the final solutions for the changes in free energy ($\mathbf{\Delta \hat{G}}$). For example, if an edge is high-leverage, a user could choose to simulate that edge for additional time or via additional replicates. This would reduce the effect of high-leverage edges on accuracy. Alternatively, if there are outliers in terms of leverage, the number of edges can be increased or the outlier excluded from the design.

Design Weights

One of the assumptions of simple linear regression is that the standard deviation of the error term is constant. Nevertheless, as studies have repeatedly observed, this assumption does not hold for alchemical transformations.[3–8] In this case, a weighted least squares model can maximize the efficiency of $\mathbf{\Delta \hat{G}}$ estimation (see Background). Without weights, transformations that are difficult to perform accurately would exert more influence than ideal on the regression. Likewise, ligands with known changes in free energies, such as reference ligands, would not have the desired increased connectivity in the design.

While a choice of weighting is necessary to guide HiMap designs, we leave the choice of similarity metric to the user. Depending on what type of free energy calculations are being done and how exactly they are done, different choices of similarity metric may be needed. The choice of weights are a separate question from how perturbation networks are planned. This aspect of planning is our focus here, whereas we treat edge weights and similarity metrics as a user choice. Accordingly, we suggest that users individually verify what rules or similarity metrics suit their application.

Presumably, efficient perturbation designs do not require perfect edge weights, meaning that imperfect weights are likely still useful, but how useful? Or, how random or noisy can the weights be for optimal designs to still be efficient? Upon a review of the statistical literature, statisticians take liberty in assigning weights based on perceived trust in the data. As justified by Williams, “great accuracy in the weights is not necessary.” In practice, if a practitioner knows that a transformation is error-prone in their protocol, they should reduce that transformation’s presence in the design. This is achieved by decreasing the weight of such a transformation. Luckily, exact correspondence between inverse variances of $\mathbf{\Delta\hat{G}}$ values and the weights is not necessary.[17,18,53] This is a fortunate conclusion since, otherwise, one might need to begin by performing a much larger set of free energy calculations to determine the precision of transformations. One could then use the inverse of the resulting statistical precisions as weights. Such a method would require a researcher to perform free energy calculations before planning a weighted perturbation map. Recent methods have implemented an adaptive approach by performing binding free energy calculations in iterations. The weights are then re-optimized in each iteration based on simulation statistical variances.[54] However, prior work indicates more approximate weights may suffice; as stated by Cutler and Flanagan, “it is not necessary for *a priori* information to be very detailed or restrictive in order that it has a significant effect on parameter-extraction accuracy.” Further, even weights that only approximately correlate with variables that reduce transformation error can still dramatically improve the design performance.[55]

Our focus here has been on efficient graph planning given an input set of similarity scores, but this means that similarity scoring remains for further consideration and is an active area of interest in the field. For example, recent work has made progress in predicting perturbation reliability in RBFE calculations using machine learning.[56] These results could be used to inform similarity scores in HiMap. In addition, with large-scale computing resources available, the curation of data sets to further investigate similarity metrics is becoming tractable and hopefully will be an area of future work.

Overall, HiMap will improve the planning of alchemical perturbation maps compared to predecessor tools such as LOMAP. Additionally, the design considerations and principles noted here, such as the scaling of errors as more nodes are included in a design, have broader applications in computational chemistry and experimental design. We hope the principles and tools presented here will be broadly useful, and future work can help inform and motivate the selection of appropriate similarity metrics for a given free energy method.

Methods and Materials

Analysis of similarity substructures

We analyzed similarities of clusters for the representative congeneric series shown in Figure 2 with the default $\epsilon = 0.95$. We collected similarities within six sets: the three detected clusters and the similarities of combined clusters minus the entries in the uncombined clusters. We used joy plots with normalized counts to show each distribution of scores. For the six sets, we then calculated and plotted the highest similarities for each ligand to the rest of the ligands in the set. Finally, we found the maximum similarity score within each set of ligands and plotted a heatmap of scores. Code to reproduce this analysis is available.[26]

Calculation of eigenvalues of the Fisher information matrix

The optimization scripts of HiMap were modified to print the eigenvalues of the Fisher information matrices by using the R *eigen* function. The input weights chosen that were held constant were the first twenty lexicographically ordered ligands of the representative congeneric series[26] analyzed in Figure 2. We generated designs and calculated eigenvalues in triplicate for each combination of n and k. We plotted continuous distributions based on the sampled data using kernel density estimation. The code for generating plots is *eigen_val_hists.py* and the raw data is *jdata_n_20.txt*.[26]

Calculation of Frobenius norm over varied edge number

We used matrix norm distances to measure the distance between the covariance matrix, $\mathbf{C}_i$, of a randomly generated graph at varied k and the covariance matrix of $\mathcal{G}(n, k_{full})$, $\mathbf{C}_{full}$. For ease of calculation, we used the Frobenius norm, where the distance between two matrices is

$$\| \mathbf{A} \|_F = \sqrt{Tr\left(\mathbf{A}^\top \mathbf{A}\right)}, \tag{26}$$

and $\mathbf{A} = \mathbf{C}_i - \mathbf{C}_{full}$. For 10, 20, and 40 n, we generated random similarity scores ranging from 0 to 1. We generated random similarities via the NumPy random_integers() function and then symmetrized the similarity array. We then varied k from $n - 1$ to $n(n-1)/2$ and generated random designs for each value of n. For each replicate, random designs were generated until a solution was found that satisfies (eq 11) so that the design is minimally connected and invertible. For 40 nodes, an edge count below 42 did not randomly generate an invertible solution, so we omitted designs with less than 42 edges from the analysis. We then calculated $\| \mathbf{A} \|_F$ according to eq 26 and repeated this process for 20 trials for each k and n combination. The similarity arrays were held constant for each value of n. Finally, we calculated the standard deviations in the Frobenius distances for the error in Figure 4. The standard deviations are the errors shown in Figure 4 with light smoothing using a 1-D Gaussian filter with a standard deviation for the Gaussian kernel of 1.2.

The scripts for generating the data with HiMap and plotting are respectively *frobenius_calc.py* and *fnorm_plot.py*.[26] The raw data files are named *fnorm_data_cv_n_*.txt* where *** is the number of nodes.[26]

Theoretical mean squared error (MSE) of the changes in free energy

The theoretical average MSE of ΔG values presented (Figure 5A, B) were calculated based on previously published simulation scripts[13] where expanded details are provided. Optimization in the numeric simulations occurred by finding the design that maximizes the design criteria in eq 10 and with methods explained above (Background). For varied n and k, the computational cost varies. The standard deviation of the observation error in $\Delta\hat{\mathbf{G}}$, σ, was set to 1.0kcal/mol within the simulations, and errors were assumed to be normally distributed. Briefly, these simulation scripts generate random $\mathbf{\Delta G^{True}}$ values, an experimental reference value (ΔG^{ref}), a weight matrix ($\mathbf{W}$), and a design matrix ($\mathbf{A}$) for all possible transformations (see Background). From these data, we then generated A-optimal, D-optimal, and random designs for varied n and k using the optimization algorithm describe previously (Software Methods). We solved for $\mathbf{\Delta \hat{G}}$ using weighted linear regression and calculated the variability between all pairs using the variances and covariances of $\mathbf{C}$. There was one reference ligand per design, and we varied the number of nodes and edges. Finally, we simulated each combination of n and k 200 times to obtain theoretical average MSE of ΔG values.

The average MSE points were plotted and fit with curves by polynomial interpolation. To obtain relative average MSE of ΔG values for stacked bar plots in Figure 5A, the minimum MSE for each isonode at n choose 2 edges was used as reference (see eq 16).

Code to generate plots in Figure 5A and B are available.[26] The script for running simulations is *simulations. R* and the script for performing analysis and plotting is *MSE_countours.py.* The raw data file is named *MSE_sim_data.txt.*[26]

Cycle number simulations and analysis

The numerical simulation scripts referenced in section 'Theoretical MSE of ΔG simulations,'[13] were run to generate A-optimal, D-optimal, and random designs with varied n and k. We generated random designs that satisfy eq 11. This restriction ensures minimally connected graphs for optimization and cycle number calculation. We ran simulations for five replicates at varied combinations of n and k. For each replicate, we calculated the number of unique cycles without double counting for the undirected graphs in R. The average number of cycles at $(n,\ k)$ were input as the third dimension in Figure 5C. We used triangular mesh interpolation to plot surfaces.

Code for running simulations with cycle number calculation and plotting is available.[26] The design and cycle counting script is *simulations_cycles.R*. The script for plotting 3D surfaces is *plot_3D_surface.py*. The raw data file is also deposited and is named *cycle_sim_data.txt.*[26]

Efficiency Calculations

We compared the efficiencies of D-optimal and A-optimal designs to radial and LOMAP 2.0 designs. To do so, we took the ratio of numerical optimization criteria as explained in eqs 10, 18, and 17. We randomly selected ligand sets from the Protein-Ligand-Benchmark set: Tyk2, Cdk2, Tnks2, Ptp1b, and Pfkfb3.[10] Ligand structure files are accessible.[10] Efficiencies were calculated over the whole ligand sets without clustering, except for Tnks2. For Tnks2, LOMAP did not output a fully connected graph. For consistency we used the ligands corresponding to the larger design output by LOMAP for efficiency calculations. HiMap calculated the reference ligand chosen for optimization and as the central hub for radial designs (Software Methods). Edge counts were held constant between compared design types.

We first ran LOMAP to measure how many edges it outputs. To generate the radial designs, we used the reference ligands as the central ligand and generated one edge per ligand to the reference ligand. For radial and LOMAP designs, we used the edge weights calculated by LOMAP similarity. For radial and LOMAP designs, we calculated **C** (eq 9) and then the optimization criteria defined in eq 10. Comparisons to the LOMAP designs for Tnks2 and Pfkfb3 were excluded due to numerical instability in **C**. Scripts to calculate the optimal criteria of radial and LOMAP designs are deposited.[26]

To perform optimizations to compare to LOMAP designs, we held the LOMAP edge number constant. Likewise, when we compared optimal designs to radial designs, we planned maps with $n - 1$ edges. Finally, we ran three optimization trials for each data point in Figure 6A and calculated standard deviations for error bars. We plotted the data using *relative_efficiencies.xlsx*.[26]

Plotting of cost savings with clustering

To plot cost surfaces for Figure 6B, we analytically generated surfaces using eqs 19–25. We used a cost of $24 per edge for RBFE calculations run on the high-performance cluster, UCI-HPC3. We calculated the cost of edges using 60 hours of GPU time per edge. The UCI-HPC3 cluster costs $0.40 per hour. We created visualizations of continuous surfaces with a meshgrid. Our code to generate cost surfaces is *cost_plot.py*.[26]

Calculation of leverages and plots

The leverages shown in Figure 7, were calculated using the diagonal values of the Hat matrix, **H** in eq 6. The script used to plot continuous distributions of leverages in Figure 7A is *levs_hist.py*. For varied k for A and D-optimal designs (Figure 7B, C) the plotting scripts used were *levs_A_var_k.py* and *levs_D_var_k.py* respectively. The raw data files for leverages are publicly available.[26]

Optimization of the BACE1 ligand set

We ran design optimizations of the BACE1 ligand set with $2n$ and $n \ln(n)$ edges. A neighbor distance of 0.4 was used for BACE1 ligands (Figure 2). The resulting perturbation maps, cluster regions, distance heatmaps, optimization run information, and depictions of clustered ligands are available open-source.[26]

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgement

We thank Pavan Behara for testing HiMap installation and for comments on this manuscript. We also thank Michael Gillhofer for early use of HiMap. For financial support, we thank the National Institutes of Health (GM108889), Janssen Pharmaceuticals, and Mary Pitman was supported in part by a fellowship from The Molecular Sciences Software Institute under NSF grant CHE-2136142.

Data and Software Availability

The HiMap software,[11] analysis code and input files[26] are publicly available and open-source under a MIT license. The software and data names produced to perform analyses presented here are detailed above (Methods and Materials). Original data is available for use with references provided. The unaltered scripts for numerical simulations of mean squared error are available at.[13]

References

(1). Holdgate GA In Reference Module in Chemistry, Molecular Sciences and Chemical Engineering; Chackalamannil S, Rotella D, Ward S, Eds.; Elsevier: Oxford, 2017; pp 180–211.

(2). Aldeghi M; Bluck JP; Biggin PC In Computational Drug Discovery and Design; Gore M, Jagtap UB, Eds.; Springer New York: New York, NY, 2018; pp 199–232.

(3). Öhlknecht C; Perthold JW; Lier B; Oostenbrink C Charge-Changing Perturbations and Path Sampling via Classical Molecular Dynamic Simulations of Simple Guest–Host Systems. J. Chem. Theory Comput. 2020, 16, 7721–7734. [PubMed: 33136389]

(4). Liu S; Wang L; Mobley DL Is Ring Breaking Feasible in Relative Binding Free Energy Calculations? J. Chem. Inf. Model. 2015, 55, 727–735. [PubMed: 25835054]

(5). Beutler TC; Mark AE; van Schaik RC; Gerber PR; van Gunsteren WF Avoiding Singularities and Numerical Instabilities in Free Energy Calculations Based on Molecular Simulations. Chem. Phys. Lett. 1994, 222, 529–539.

(6). Boresch S; Bruckner S Avoiding the Van Der Waals Endpoint Problem Using Serial Atomic Insertion. J. Comput. Chem. 2011, 32, 2449–2458. [PubMed: 21607991]

(7). Mey ASJS; Allen BK; McDonald HEB; Chodera JD; Hahn DF; Kuhn M; Michel J; Mobley DL; Naden LN; Prasad S; Rizzi A; Scheen J; Shirts MR; Tresadern G; Xu H Best Practices for Alchemical Free Energy Calculations [Article V1.0]. Living Journal of Computational Molecular Science 2020, 2.

(8). Liu S; Wu Y; Lin T; Abel R; Redmann JP; Summa CM; Jaber VR; Lim NM; Mobley DL Lead Optimization Mapper: Automating Free Energy Calculations for Lead Optimization. J. Comput.-Aided Mol. Des. 2013, 27, 755–770. [PubMed: 24072356]

(9). OpenEye Scientific, OpenEye Toolkits 2021.2.0. 2021; http://www.eyesopen.com.

(10). Hahn D; Bayly C; Boby ML; Bruce Macdonald H; Chodera J; Gapsys V; Mey A; Mobley D; Perez Benito L; Schindler C; Tresadern G; Warren G Best Practices for Constructing, Preparing, and Evaluating Protein-Ligand Binding Affinity Benchmarks [Article V1.0]. Living Journal of Computational Molecular Science 2022, 4, 1497. [PubMed: 36382113]

(11). Pitman M; Mobley D HiMap V1.0.0. 2022; https://github.com/MobleyLab/HiMap.

(12). Xu H Optimal Measurement Network of Pairwise Differences. J. Chem. Inf. Model. 2019, 59, 4720–4728. [PubMed: 31613620]

(13). Yang Q; Burchett W; Steeno GS; Liu S; Yang M; Mobley DL; Hou X Optimal Designs for Pairwise Calculation: An Application to Free Energy Perturbation in Minimizing Prediction Variability. J. Comput. Chem. 2020, 41, 247–257. [PubMed: 31721260]

(14). Gauss CF Theoria Combinationis Observationum Erroribus Minimis Obnoxiae. Commentationes Cocietatis Regiae Scientiarum Gottingensis Recentiores 1821, 5, 6–93.

(15). Rao CR In Bulletin of the Calcutta Mathematical Society; Kotz S, Johnson NL, Eds.; Springer New York: New York, NY, 1992; pp 235–247.

(16). Rousseuw P; Leroy A Robust Regression and Outlier Detection; Wiley Series in Probability and Statistics; John Wiley & Sons, 1987; pp 216–247.

(17). Carroll R; Ruppert D Transformation and Weighting in Regression, 1st ed.; Chapman & Hall: New York, NY, 1988.

(18). Vallisneri M Use and Abuse of the Fisher Information Matrix in the Assessment of Gravitational-Wave Parameter-Estimation Prospects. Phys. Rev. D 2008, 77, 042001.

(19). Westat JL; Valliant R Survey weighted hat matrix and leverages. Qual. Eng. 2010, 55, 39–40.

(20). Greub WH, Graduate Texts in Mathematics, Linear Algebra, 4th ed.; Springer-Verlag New York, 1981.

(21). Wittman D Fisher Matrix for Beginners. 2015; https://wittman.physics.ucdavis.edu/Fisher-matrix-guide.pdf.

(22). Suzuki J Quantum-State Estimation Problem *via* Optimal Design of Experiments. Int. J. Quantum Inf. 2021, 19, 2040007.

(23). Miller AJ; Nguyen N-K Algorithm AS 295: A Fedorov Exchange Algorithm for D-optimal Design. J. R. Stat. Soc., C: Appl. Stat. 1994, 43, 669–677.

(24). Hahsler M; Piekenbrock M; Doran D Dbscan: Fast Density-Based Clustering with R. J. Stat. Softw. 2019, 91, 1–30.

(25). Bushra AA; Yi G Comparative Analysis Review of Pioneering DBSCAN and Successive Density-Based Clustering Algorithms. IEEE Access 2021, 9, 87918–87935.

(26). Pitman M HiMap Manuscript Analyses Repository. 2022; https://github.com/pitmanme/HiMap_ms_analyses.

(27). Satopaa V; Albrecht J; Irwin D; Raghavan B Finding a "Kneedle" in a Haystack: Detecting Knee Points in System Behavior. 2011 31st International Conference on Distributed Computing Systems Workshops. 2011; pp 166–171.

(28). Fedorov VV; Hackl P Optimal Experimental Design : Spatial Sampling. Calcutta Statistical Association Bulletin 1994, 44, 57–82.

(29). Fedorov V Optimal Experimental Design. WIREs Comput. Stat. 2010, 2, 581–589.

(30). Richard B; Chambers J; Wilks A The New S Language; Chapman & Hall, 1988.

(31). Abbas A; Sutter D; Zoufal C; Lucchi A; Figalli A; Woerner S The Power of Quantum Neural Networks. Nat. Comput. Sci. 2021, 1, 403–409.

(32). Rényi A; Erdős P On Random Graph. Publ. Math. 1959, 6, 290–297.

(33). Erdős P; Rényi A On the Strength of Connectedness of a Random Graph. Acta Math. Hung. 1961, 12, 261–267.

(34). Erdös P; Palmer EM; Robinson RW Local Connectivity of a Random Graph. J. Graph Theory 1983, 7, 411–417.

(35). Evans MJ; Rosenthal JS Probability and Statistics: The Science of Uncertainty; Freeman WH, 2004.

(36). Fouque P-A; Joux A; Mavromati C Multi-User Collisions: Applications to Discrete Logarithm, Even-Mansour and PRINCE BT - Advances in Cryptology – ASIACRYPT 2014. International Conference on the Theory and Application of Cryptology and Information Security. Berlin, Heidelberg, 2014; pp 420–438.

(37). Prakash BA Propagation and Immunization in Large Networks. XRDS 2012, 19, 56–59.

(38). Chen C; Tong H; Prakash BA; Eliassi-Rad T; Faloutsos M; Faloutsos C EigenOptimization on Large Graphs by Edge Manipulation. ACM Trans. Knowl. Discov. Data 2016, 10.

(39). Wang L; Deng Y; Knight JL; Wu Y; Kim B; Sherman W; Shelley JC; Lin T; Abel R Modeling Local Structural Rearrangements Using FEP/REST: Application to Relative Binding Affinity Predictions of CDK2 Inhibitors. J. Chem. Theory Comput. 2013, 9, 1282–1293. [PubMed: 26588769]

(40). Wang L; Wu Y; Deng Y; Kim B; Pierce L; Krilov G; Lupyan D; Robinson S; Dahlgren MK; Greenwood J; Romero DL; Masse C; Knight JL; Steinbrecher T; Beuming T; Damm W; Harder E; Sherman W; Brewer M; Wester R; Murcko M; Frye L; Farid R; Lin T; Mobley DL; Jorgensen WL; Berne BJ; Friesner RA; Abel R Accurate and Reliable Prediction of Relative Ligand Binding Potency in Prospective Drug Discovery by Way of a Modern Free-Energy Calculation Protocol and Force Field. J. Am. Chem. Soc. 2015, 137, 2695–2703. [PubMed: 25625324]

(41). Gapsys V; Pérez-Benito L; Aldeghi M; Seeliger D; van Vlijmen H; Tresadern G; de Groot BL Large Scale Relative Protein Ligand Binding Affinities Using Non-equilibrium Alchemy. Chem. Sci. 2020, 11, 1140–1152.

(42). Zou J; Li Z; Liu S; Peng C; Fang D; Wan X; Lin Z; Lee T-S; Raleigh DP; Yang M; Simmerling C Scaffold Hopping Transformations Using Auxiliary Restraints for Calculating Accurate Relative Binding Free Energies. J. Chem. Theory Comput. 2021, 17, 3710–3726. [PubMed: 34029468]

(43). Baxter EW; Conway KA; Kennis L; Bischoff F; Mercken MH; De Winter HL; Reynolds CH; Tounge BA; Luo C; Scott MK; Huang Y; Braeken M; Pieters SMA; Berthelot DJC; Masure S; Bruinzeel WD; Jordan AD; Parker MH; Boyd RE; Qu J; Alexander RS; Brenneman DE; Reitz AB 2-Amino-3,4-Dihydroquinazolines as Inhibitors of BACE-1 (β-Site APP Cleaving Enzyme): Use of Structure Based Design to Convert a Micromolar Hit into a Nanomolar Lead. J. Med. Chem. 2007, 50, 4261–4264. [PubMed: 17685503]

(44). Tresadern G; Delgado F; Delgado O; Gijsen H; Macdonald GJ; Moechars D; Rombouts F; Alexander R; Spurlino J; Gool MV; Vega JA; Trabanco AA Rational Design and Synthesis of Aminopiperazinones as β-Secretase (BACE) Inhibitors. Bioorganic & Medicinal Chemistry Letters 2011, 21, 7255–7260. [PubMed: 22071305]

(45). Rombouts FJR; Tresadern G; Delgado O; Martínez-Lamenca C; Van Gool M; García-Molina A; Alonso de Diego SA; Oehlrich D; Prokopcova H; Alonso JM; Austin N; Borghys H; Van Brandt S; Surkyn M; De Cleyn M; Vos A; Alexander R; Macdonald G; Moechars D; Gijsen H; Trabanco AA 1,4-Oxazine β-Secretase 1 (BACE1) Inhibitors: From Hit Generation to Orally Bioavailable Brain Penetrant Leads. J. Med. Chem. 2015, 58, 8216–8235. [PubMed: 26378740]

(46). Fujimoto K; Yoshida S; Tadano G; Asada N; Fuchino K; Suzuki S; Matsuoka E; Yamamoto T; Yamamoto S; Ando S; Kanegawa N; Tonomura Y; Ito H; Moechars D; Rombouts FJR; Gijsen HJM; Kusakabe K.-i. StructureBased Approaches to Improving Selectivity through Utilizing Explicit Water Molecules: Discovery of Selective β-Secretase (BACE1) Inhibitors over BACE2. J. Med. Chem. 2021, 64, 3075–3085. [PubMed: 33719429]

(47). Oehlrich D; Peschiulli A; Tresadern G; Van Gool M; Vega JA; De Lucas AI; Alonso de Diego SA; Prokopcova H; Austin N; Van Brandt S; Surkyn M; De Cleyn M; Vos A; Rombouts FJR; Macdonald G; Moechars D; Gijsen HJM; Trabanco AA Evaluation of a Series of β-Secretase 1 Inhibitors Containing Novel Heteroaryl-Fused-Piperazine Amidine Warheads. ACS Med. Chem. Lett. 2019, 10, 1159–1165. [PubMed: 31413800]

(48). König G; Ries B; Hünenberger PH; Riniker S Efficient Alchemical Intermediate States in Free Energy Calculations Using λ-Enveloping Distribution Sampling. J. Chem. Theory Comput. 2021, 17, 5805–5815. [PubMed: 34476947]

(49). Mobley DL; Klimovich PV Perspective: Alchemical Free Energy Calculations for Drug Discovery. J. Chem. Phys. 2012, 137.

(50). Carlone L; Censi A From Angular Manifolds to the Integer Lattice: Guaranteed Orientation Estimation With Application to Pose Graph Optimization. IEEE Transactions on Robotics 2014, 30, 475–492.

(51). Carlone L; Tron R; Daniilidis K; Dellaert F Initialization techniques for 3D SLAM: A survey on rotation estimation and its use in pose graph optimization. 2015 IEEE ICRA. 2015; pp 4597–4604.

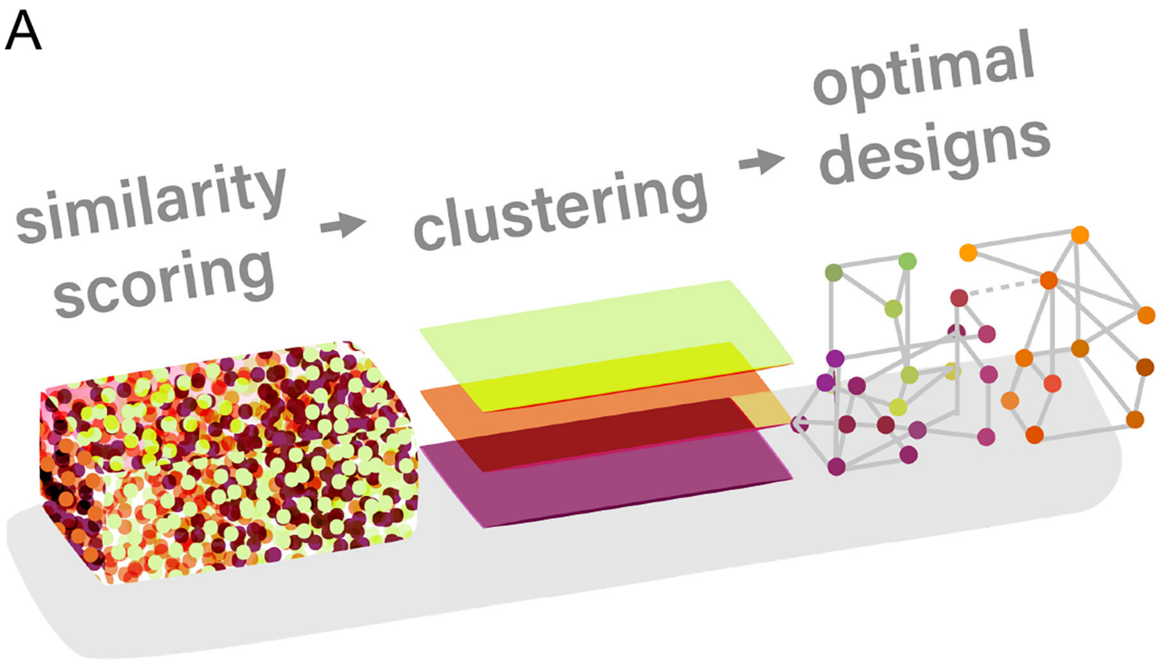
(52). Dette H Designing Experiments with Respect to ‘Standardized’ Optimality Criteria. Journal of the Royal Statistical Society. Series B (Methodological) 1997, 59, 97–110.

(53). Williams E Regression Analysis; Wiley Series in Probability and Statistics: Applied Probability and Statistics Section Series; Wiley, 1959.

(54). Li P; Li Z; Wang Y; Dou H; Radak BK; Allen BK; Sherman W; Xu H Precise Binding Free Energy Calculations for Multiple Molecules Using an Optimal Measurement Network of Pairwise Differences. J. Chem. Theory Comput. 2022, 18, 650–663. [PubMed: 34871502]

(55). Cutler C; Flanagan EE Gravitational Waves from Merging Compact Binaries: How Accurately Can One Extract the Binary’s Parameters from the Inspiral Waveform? Phys. Rev. D 1994, 49, 2658–2697.

(56). Scheen J; Mackey M; Michel J Data-driven generation of perturbation networks for relative binding free energy calculations. Digital Discovery 2022, 870–885.



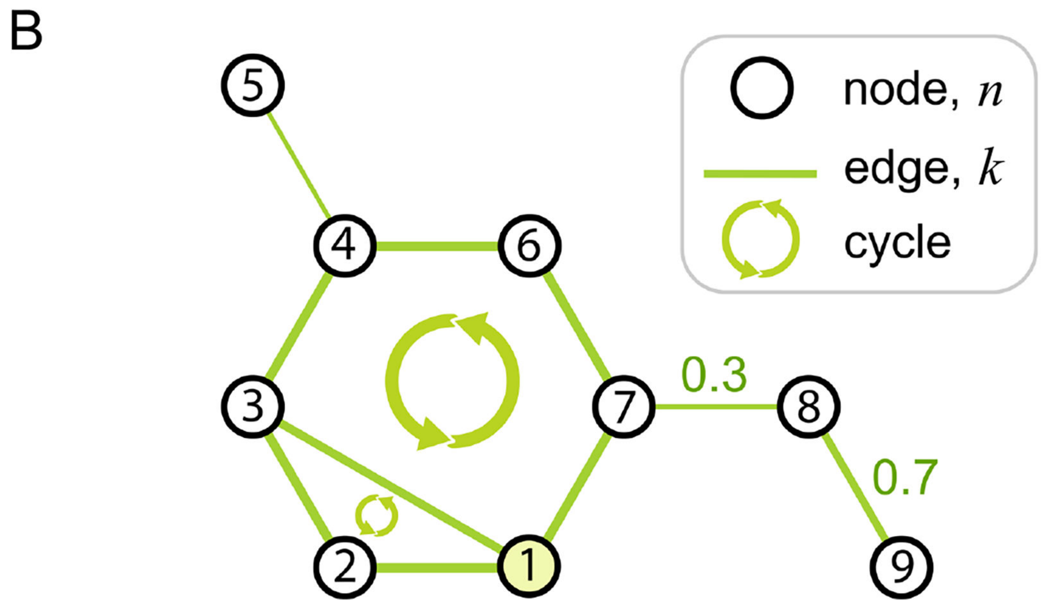


Figure 1:
(A) The HiMap algorithm. First, we score ligands by a generalized similarity metric. Then, we cluster ligands with density-based spatial clustering, DBSCAN. Finally, we find an optimal design for each cluster. The edges in 'optimal designs' that are shown as grey dotted lines illustrate optional edges. The clustering step in HiMap is optional so either separate optimal graphs can be found for each cluster, or one optimal graph can be found for the whole set of ligands. (B) An example design where nodes or ligands, n, are black circles,

numbered. The number of edges or perturbations, k, are shown as green lines. Numbered edges depict edge weights. Circular arrows show two cycles.

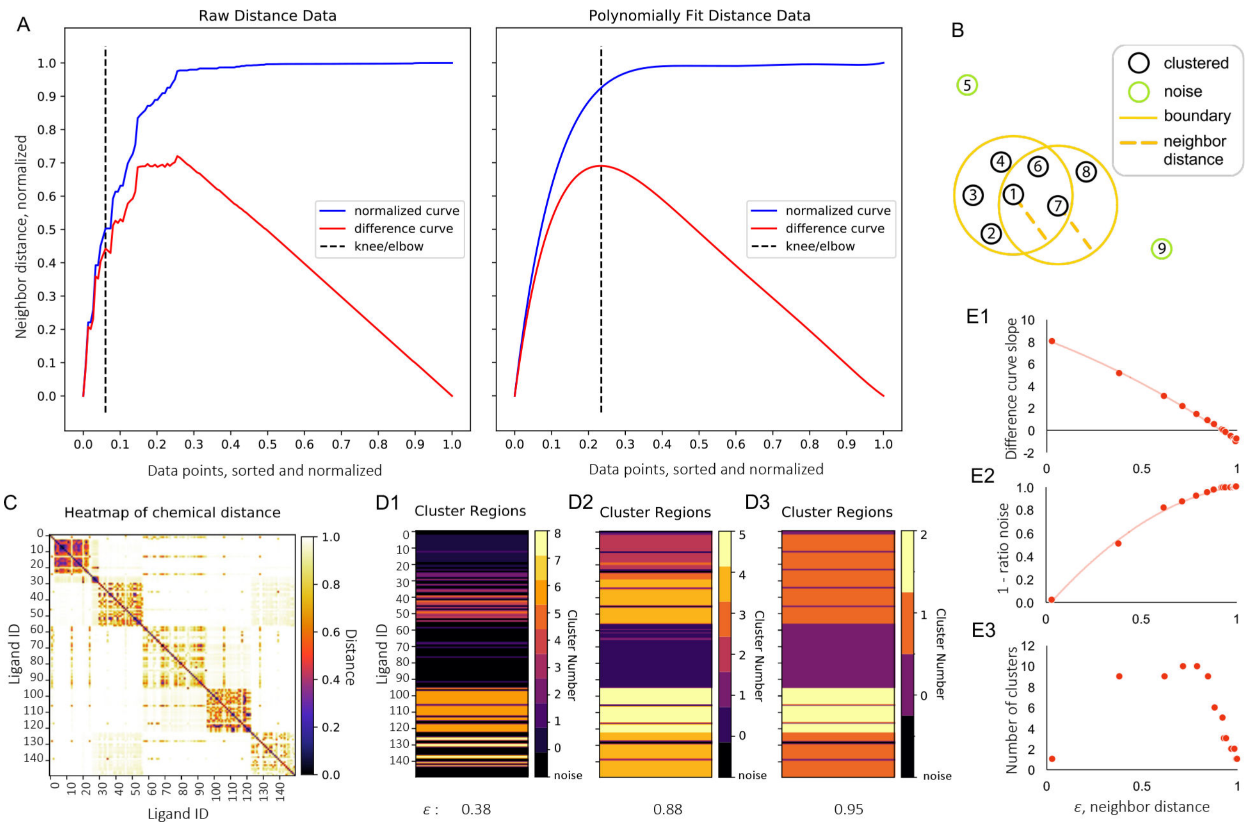
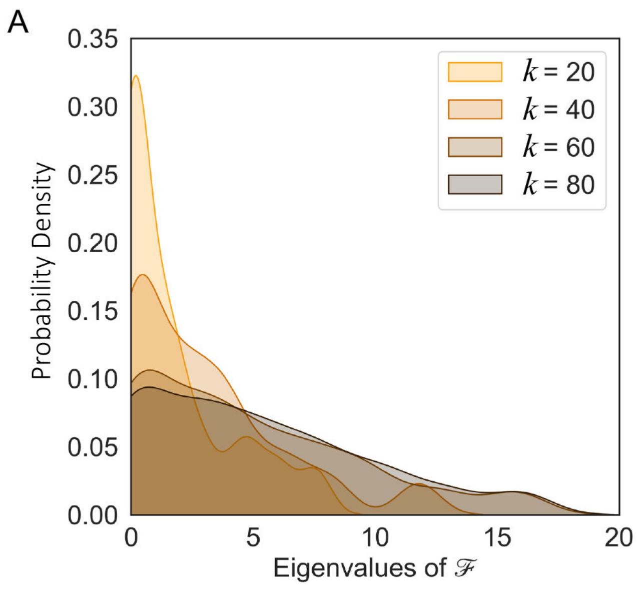


Figure 2:

Details on clustering in HiMap using a representative congeneric series with similarity data publicly available.[26] (A) HiMap outputs plots to calculate the neighbor distance using the Kneedle algorithm for detecting points of maximum curvature in discrete data. The x-axes are data points sorted by ascending distances between ligands, and the y-axes are neighbor distances. Both axes are normalized to range from 0 to 1. The difference curve, red, is the vertical distance from the normalized curve, blue, to a line spanning from (0, 0) to (1, 1). The extrema of difference curves are points of maximum curvature of the normalized curve; the corresponding y-value is the neighbor distance value, ϵ. The input data (left) is polynomially fit (right) to remove noise. HiMap calculates the default ϵ from the global maximum of the polynomially fit difference curve. (B) Definitions of parameters in density-based spatial clustering, DBSCAN. Ligands that fall within the cluster boundaries, yellow circles, are circled with black. The neighbor distance cutoff is shown as a yellow dashed line. Ligands outside the neighbor distance from the cluster are circled with green. (C) Heatmap of the ligand distances of the representative series. White or a distance of 1.0 means no measured similarity. Black, or a value of 0.0, means identity. (D1–3) Clustered regions at varied neighbor distances, ϵ, for the corresponding heatmap of distances in C. Here, ligands labeled with the same color are in the same cluster as defined by the color bars. The color black corresponds to noise points. (D1) Clusters and noise at an ϵ of 0.39. (D2) Clusters found at an ϵ of 0.88, which is within the reported range in A. (D3) Clusters for an ϵ of 0.95, which is the calculated default for this data. (E1–3) The number of clusters and noise points varies with the neighbor distance selected. (E1) The autodetected ϵ is where the difference curve slope crosses zero. (E2) As the slope and sensitivity increase, more ligands are excluded as noise. (E3) The number of clusters at a varied neighbor distance.



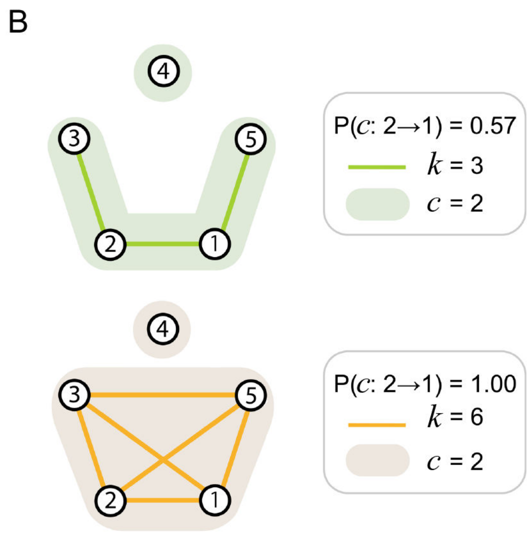


Figure 3:
The effect of adding edges to a random design. (A) The eigenspectra of the Fisher information matrices of random graphs, $\mathscr{F}$, for 20 ligands and varied edge number, k. Each distribution shown includes data from three trials. (B) Illustration of graphs with randomly added edges. In this example, the number of ligands, n, is 5, k is 3 (top) and 6 (bottom), and the number of connected components, c, is 2 for both cases. The probability of a new edge altering c is 2/3 for the top and 1 for the bottom.

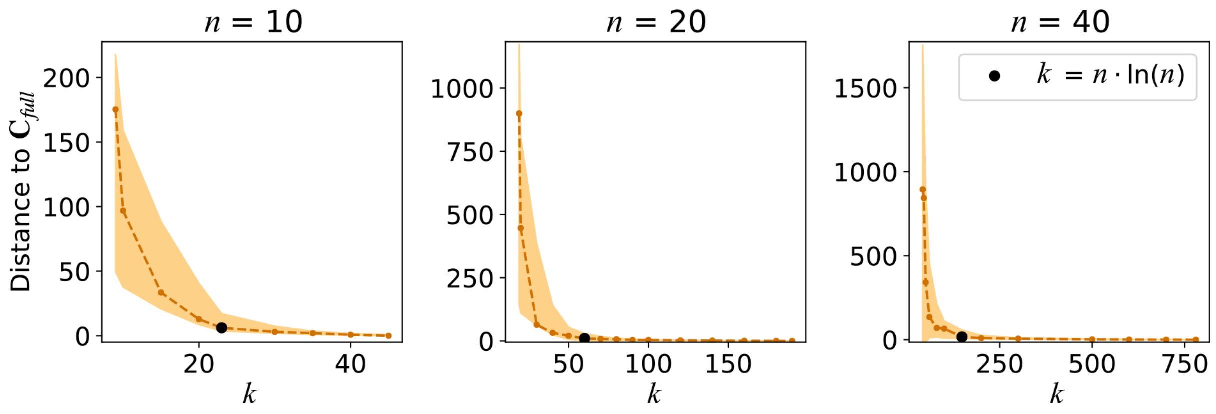


Figure 4:

The matrix norm distance (eq 26) of the covariance matrices of randomly generated, minimally connected graphs to the covariance matrix of the full design with all n choose 2 edges, $\mathbf{C}_{full}$. The weights used were randomly generated for each value of n and then held constant when varying k. We repeated data collection 20 times for each combination of n and k. The shaded regions show the standard deviation of the distances. The integer edge number, k, nearest to $n \cdot \ln(n)$ is shown with a black circular marker.

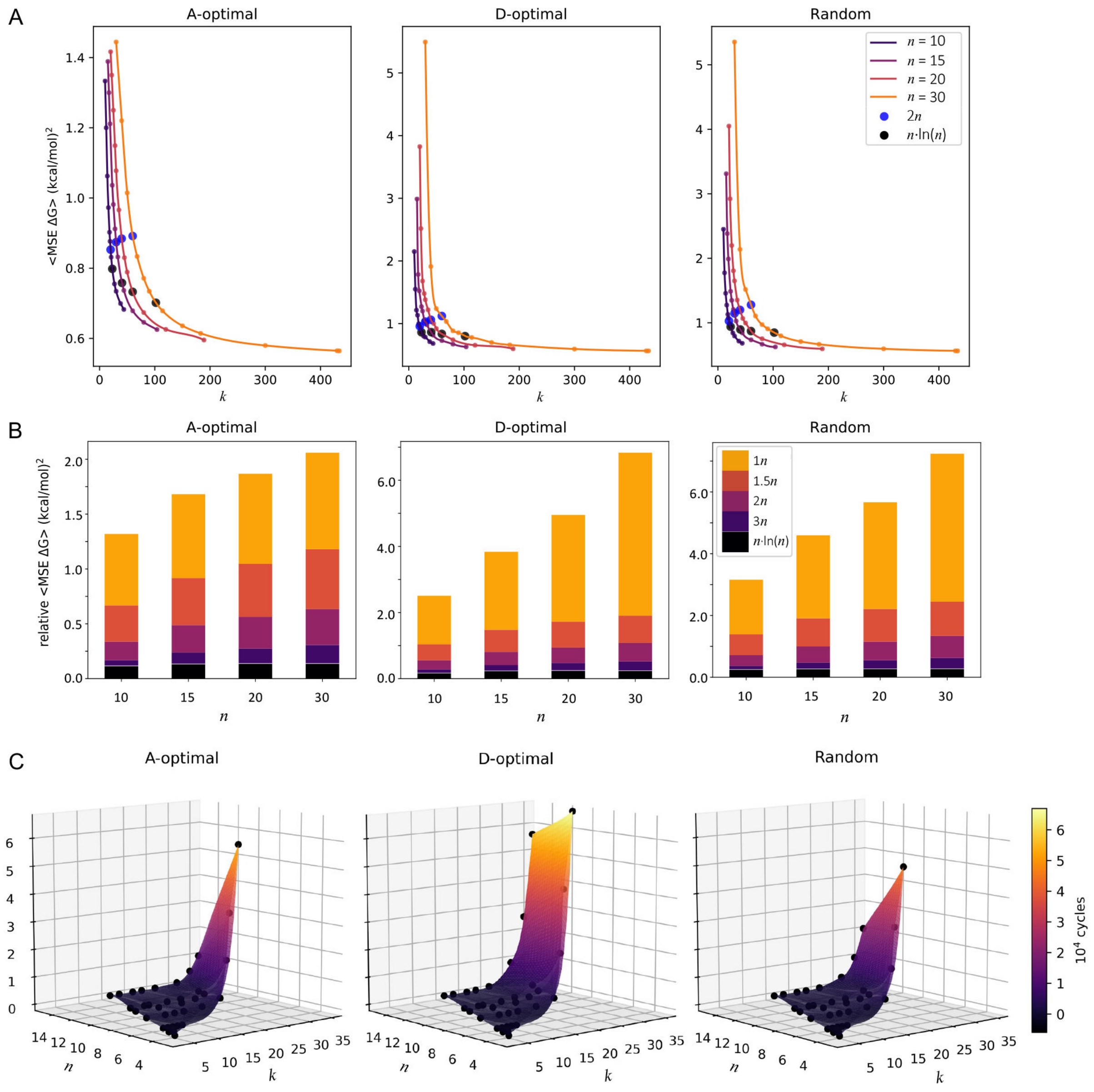


Figure 5:

Scaling effects with ligand number. (A) The mean squared error (MSE) in ΔG with varied n and k for A-optimal, D-optimal, and pseudo-random designs. The upper plots show data from 200 optimization simulations to fit free energy estimates for each data point on the curves. Each curve is an isonode at $n = 10, 15, 20$, or 30. The MSE at $k = n \cdot \ln(n)$ on each curve is shown as a black dot and at $k = 2n$ as a blue dot. (B) Plots show the MSE relative to the minimum possible error per isonode (eq 16) as stacked bar plots. (C) Design optimizations and cycle number calculations for 5 replicates are shown as black points at varied n and k. The number of cycles for each point depicted is the expectation of the number of complete cycles formed. Surfaces are plotted with triangular mesh interpolation.

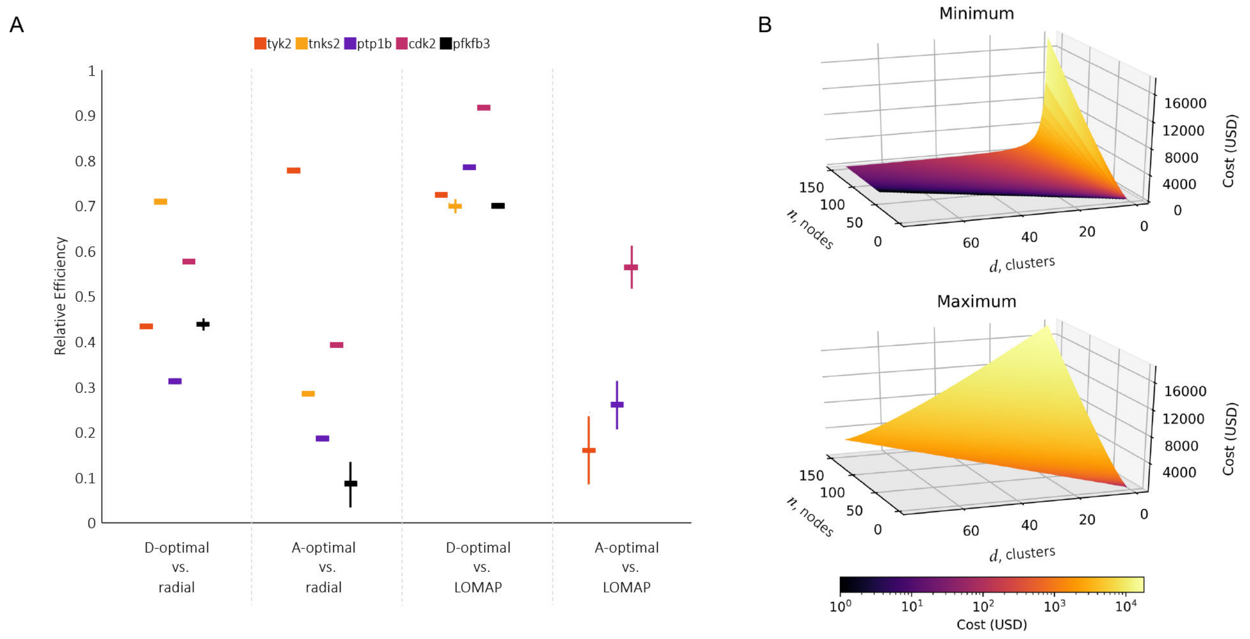


Figure 6:

Performance gains in HiMap. (A) The relative efficiencies of D-optimal and A-optimal designs to radial and LOMAP generated designs. A value less than 1 indicates efficiency gains in the optimal design relative to the reference design. To compare to radial designs, we optimized designs with $n - 1$ edges and set the reference ligand to the ligand with the highest sum similarity to the rest of the set. For the optimal designs that we compared to LOMAP designs, we set the number of edges for optimized designs to the number of edges output in the LOMAP design. Error bars are the standard deviations in efficiency after running optimization three times each. Raw data is shown in Table S1. (B) Cost with clustering at $n \cdot \ln(n)$ edges or constant expected MSE. The top plot shows the best-case savings or minimum cost where the clusters are equally balanced in size. The bottom plot shows the worst-case cost-benefit in the domain where clusters are least balanced. The minimum cluster size is defined as $n = 2$. Edge costs are calculated for the high-performance cluster, UCI-HPC3, at \$24/edge (Methods and Materials).

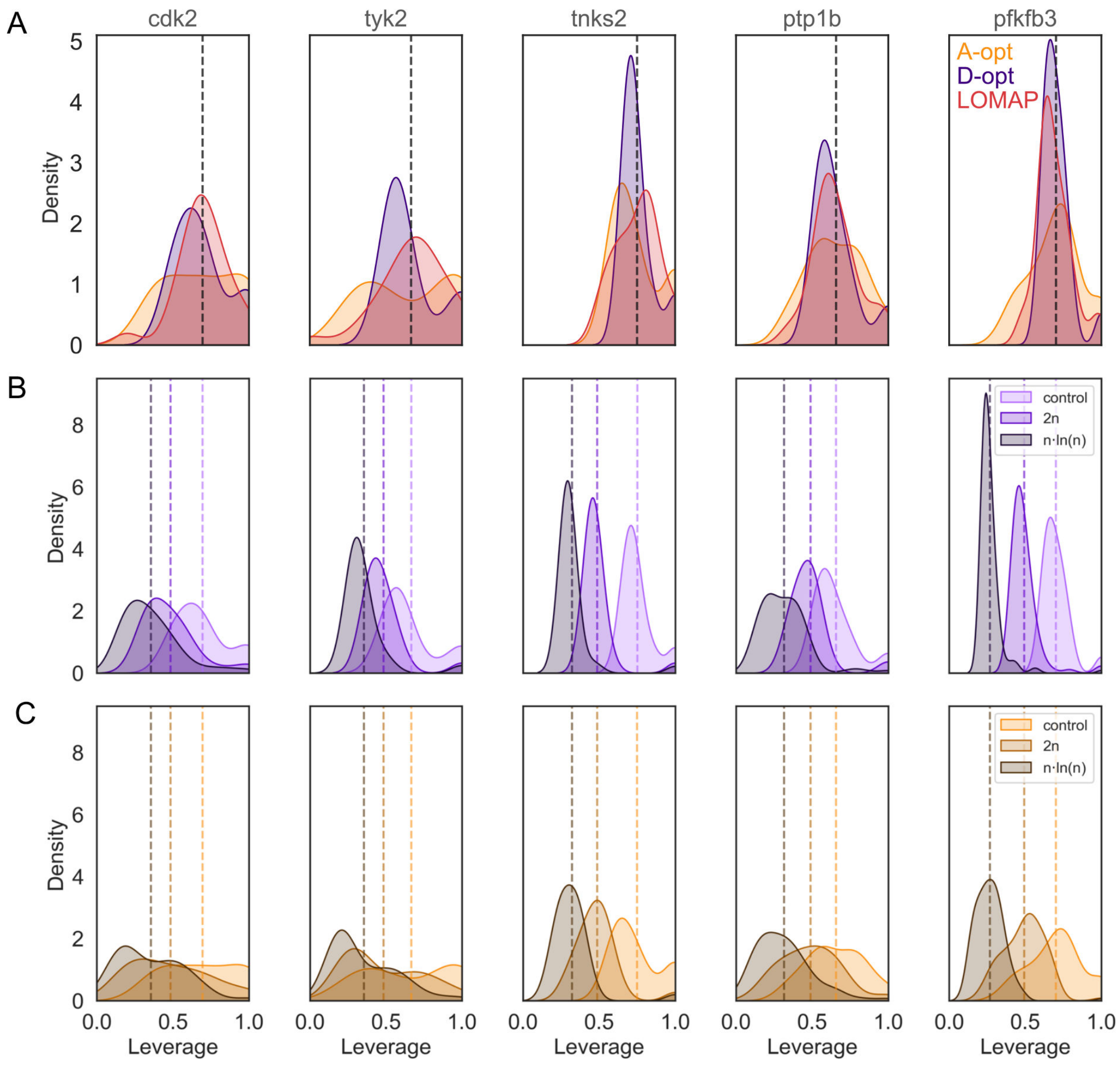


Figure 7:

Leverages of edges in designs for Protein-Ligand Benchmark ligand datasets.[10] (A) Comparison of edge leverages for A-optimal designs shown in orange (A-opt), D-optimal designs shown in purple (D-opt), and LOMAP designs shown in red. Each distribution has a constant number of edges as defined by the output edge number of LOMAP. The vertical dashed lines are the average leverage. A value of 1.0 corresponds to high leverage. (B) In purple, the leverages of D-optimal designs are shown for Protein-Ligand Benchmark ligand datasets at varied edge counts where the control value is the output edge number of LOMAP. Vertical dashed lines are the average leverage and are colored by the corresponding distribution. (C) Leverages of A-optimal designs using the same plotting methods as in panel B.

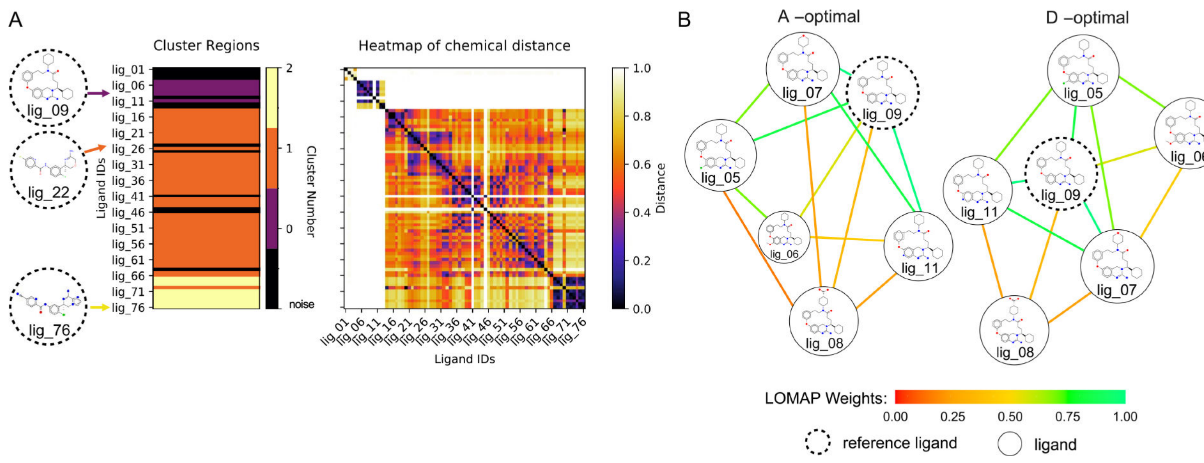


Figure 8:
Example optimization of *β*-secretase enzyme 1, BACE1, perturbation maps. (A) Cluster regions correspond to clusters d_0, d_1, and d_2 with noise points at a neighbor distance of 0.4. The ligands of greatest sum similarity to a cluster are shown on the left in a dashed circular outline, with colored arrows corresponding to the cluster region color. The cluster region plot is vertically aligned with the heatmap of chemical distance on the right. Here a value of 0.0 (colored black) means ligands are identical; a value of 1.0 (colored white) means ligands are dissimilar by LOMAP similarity score. (B) Perturbation maps for cluster d_0 at $2n$ edges. The A-optimal map is on the left, and the D-optimal map is on the right. The edge weights are colored so that red means low weight or low LOMAP similarity, and green is a high weight or high LOMAP similarity. The reference ligand is shown in a dashed circular outline.