Data-Driven Insights into Ligands, Proteins, and Genetic Mutations

by

Jing Lu

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Bioinformatics)
in the University of Michigan
2016

Doctoral Committee:

Professor Heather A. Carlson, Chair
Professor Charles L. Brooks III
Assistant Professor Barry Grant
Professor David S. Sept
Professor Kerby A. Shedden

Acknowledgements

I would like to thank my advisor, Dr. Heather Carlson, for years of patient guidance, teaching, and support through the course of my PhD. I have learnt how to think critically and be rigorous in every step of research. I also want to express gratitude to my committee: Professor Charles L. Brooks III, Assistant Professor Barry Grant, Professor David S. Sept, Professor Kerby A. Shedden. Their advising is insightful and deepens my understanding of my research projects.

I would like to thank Dr. Richard Smith for timely support for both my writing and research. For many Saturdays and Sundays, he promptly responds my requests for proofreading. Much of my work is built on his code in protein and ligand analysis.

I would like to thank other members in Dr. Carlson's lab for helping me with my work. Through the discussion with Dr. Jim Dunbar, I have learnt many critical ideas in Cheminformatics. Also, thank you to Sarah Graham and Jordan Clark for their tremendous friendship and willing to help with my writing. I would also thank previous members in Dr. Carlson's lab. I would thank Dr. Phani Ghanakota for many late-night discussions and Dr. Aqeel Ahmed for collaborations on projects.

I would also like to thank Department of Computational Medicine and Bioinformatics for supporting me to finish my work, especially Dr. Margit Burmeister. She has been a great friend and mentor for helping me survive my PhD.

Last but not least, I would thank my dear life-long partner, Kuangli Xie. She makes me feel calm during hard time. Her encouragement gives me the strength through difficulties. She gave me the feeling of completeness and makes me a better person every single day. I feel so lucky to have her in my life to write our stories together.

Table of Contents:

List of Tables

List of Figures

List of Appendices

Abstract

Structure-based drug design (SBDD) is an essential component of many drug discovery programs. In SBDD, a large number of potential molecules are virtually screened against the known three-dimensional protein structure. Proper creation of ligand libraries and selection of target binding sites are critical for SBDD. This thesis focuses on knowledge-based approaches to improve SBDD in two aspects, the construction of the ligand libraries and analysis of the target binding sites used.

First, this thesis presents ChemTreeMap, a visualization tool to explore structurally diverse molecules and mine for correlation between chemical structure and biological data. The visualization tool is applicable to a wide range of questions involving small molecule/drug binding and exploration and construction of ligand libraries. Experimental data and molecular properties can be interactively visualized with graph properties. With the help of this powerful tool, this thesis reports the findings on discriminating physicochemical properties between allosteric and orthosteric competitive molecules. It is observed that allosteric ligands are more hydrophobic, aromatic, and rigid. The result is useful to guide building new chemical libraries biased towards allosteric regulators. Thirdly, the selection of target binding sites of drug candidates needs to take account for possible interruption due to mutations which

occur from non-synonymous single nucleotide polymorphisms (nsSNPs). Disease nsSNPs occur more frequently in a protein core or binding site, rather than the rest of the protein surface. The result can be used to imply the probability and consequence of nsSNP on new target binding sites.

Chapter 1. Introduction

1.1 Overview:

Proteins are composed of one or multiple chains of amino acids, performing various biochemical functions in a living organism. Proteins perform a wide variety of functions including DNA replication, responding to stimuli, transporting small molecules, and catalyzing biochemical reactions [1]. Functions are accomplished by interacting with other components in the cellular environment, including other proteins, DNA, RNA, or smaller molecules like metabolites, substrates, nucleotides among others. Proteins are common biological targets for drugs [2]–[4], which are typically small molecules that can be used to diagnose, prevent, treat, or cure a disease or enhance well-being[5]. Understanding protein small molecule interactions is essential to comprehend the biological process and facilitate the design of a small molecule or potential drug to modulate protein function by mimicking these interactions.

The most widely-adopted method for drug discovery is known as forward pharmacology, which only relies on testing of known chemical substances with drug-like characteristics on biological assays (i.e. proteins or cultured cells) [6] and does not necessarily utilize protein target information. In contrast, rational drug design takes the reverse approach by modulating a specific biological target with a small molecule to achieve a desired physiological response[7]. In early stages of rational drug design, the discovery of novel leads that have potential interactions with specific protein targets is of central importance [8]. This requires a large diverse library of potential substances to be tested against targets in a relatively short amount of time. One typical searching strategy for identifying novel leads is through high throughput screening (HTS), which can quickly assay the biological or biochemical activities of a large number of compounds. However, the application of HTS is limited because of its high cost and low hit rate. In order to overcome this disadvantage, the earliest efforts were made in the early 1990s to successfully design a set of molecules targeting HIV, by investigating the interaction between inhibitors and enzyme to occur and symmetry characteristics of the enzyme active site[9]–[11]. This approach, termed structure-based drug design (SBDD), is guided by knowledge of target three-dimensional structures obtained by x-ray crystallography or NMR spectroscopy[12]. SBDD has become a starting point and integral part of many drug discovery programs[13]. In SBDD, a large number of potential molecules are virtually screened against the known three dimensional protein structure computationally [8]. It can accelerate the screening process for large molecule sets. Chapter 2 and 3 for this thesis are about finding properties or developing visualization tools to help focus and explore the chemical space to potentially optimize libraries of molecules used in SBDD.

A typical flow chart (Figure 1-1) of SBDD begins with a target structure and a compound library. The target structure is either experimentally solved (using x-ray crystallography (preferred) or NMR) or computationally modeled. A compound library of drug-like molecules, for example, a database of commercially-available compounds is described computationally. Then, molecules in the library are tested against the target protein by virtually docking into a target binding site. The docking program serves to position a complete small molecule in the protein's binding site. Alternatively, one could integrate with linking and molecule building methods to generate plausible modulators by incorporating multiple functional groups. The methods of linking/building may produce compounds that are not synthesized or cannot be synthesized, but may be able to provide a framework to search for a small molecule that may have the desired affinity. The combined molecule generated by linking and building may have higher target affinity than components separately[14]. The positioned ligands are sorted by scoring functions that approximate the free energy change upon protein-ligand binding. Promising compounds with high scores are synthesized and tested in a biological array for specificity, pharmacodynamics, and toxicity. The results from testing can initiate a new round of structure-based drug design until the desired compounds are discovered to enter the next phase of drug discovery.

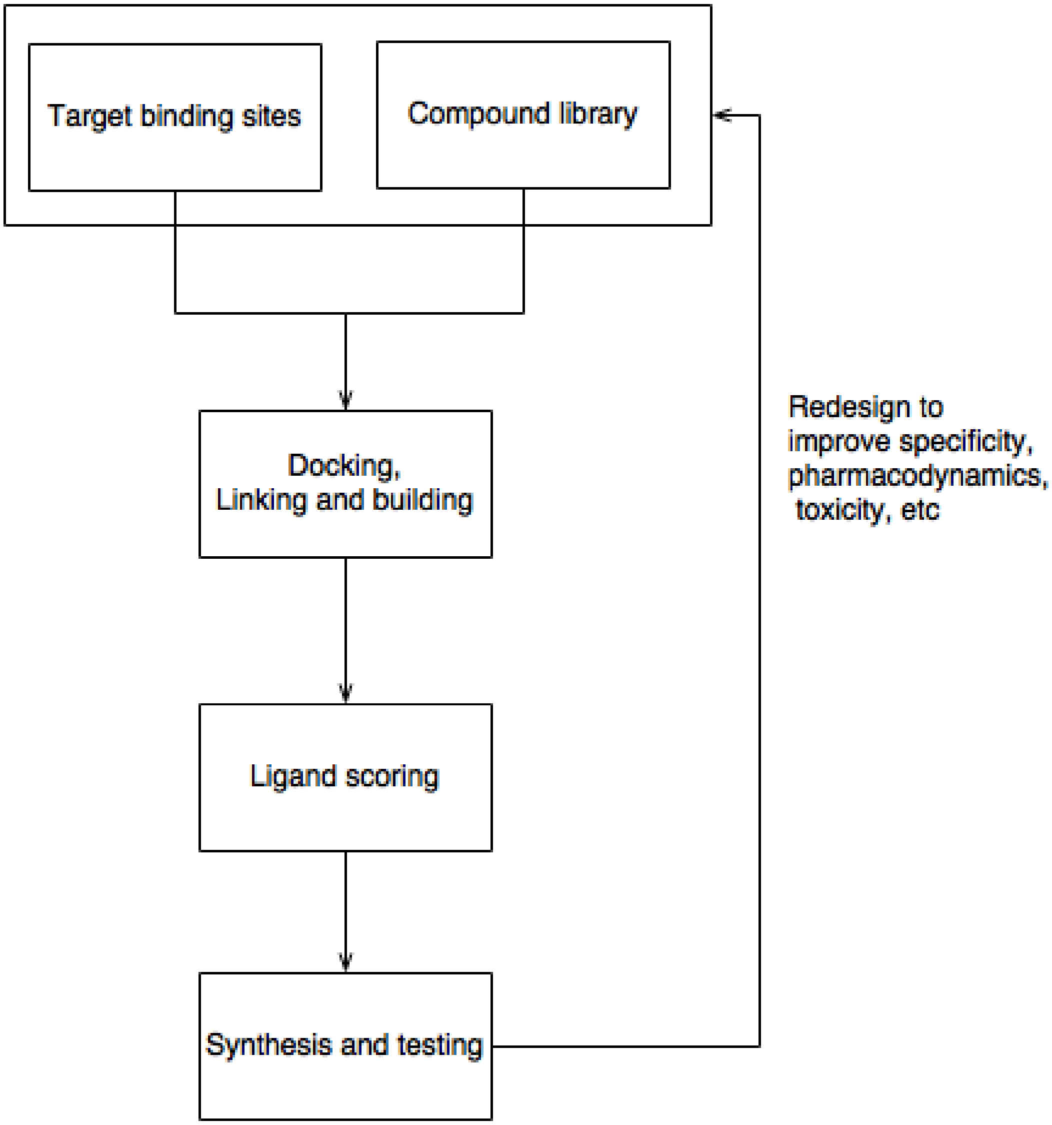


Figure 1-1. Simplified structure-based drug design workflow.

Much efforts in SBDD involved in characterizing and testing molecules in the compound libraries, which provides potential candidates in the early-stage of SBDD. The success of SBDD will be dependent on the coverage of the desired “chemical space” in the compound library. “Chemical space” is defined as “the total descriptor space covered by all the known and possible small organic compounds”[15]. However, the total number of possible molecules could be too large to proceed following analysis for a selected chemical space. For example, the total number of molecules for complete coverage is theoretically more than 10^60, for considering molecules composed by only carbon, nitrogen, oxygen, or sulfur atoms, with up to 30 atoms, 4 rings, and 10 branch points [16]. In practice, initial libraries usually are extracted from existing sources, such as corporate screening collections, purchasable compound libraries (e.g. ZINC), public chemical databases (e.g. ChEMBL), or combinatorial libraries [17]. Some widely used chemical compound repositories are shown in Table I, which covers libraries with varying numbers of protein targets. There are roughly three types of libraries based on the number of protein targets: 1) large general libraries which can be used against many targets (e.g. PubChem, ZINC, CoCoCo, eMolecules, ChEMBL); 2) focused libraries which targets a protein family or a small group of related targets (BindingDB, DrugBank); and 3) an enriched specific library on a single target (ChemDiv) [18]. After the libraries are selected, careful preparation of molecular structures has pivotal effects on the success of the docking process. It is important to ensure proper chirality, stereo chemical specifications, tautomeric state, and protonation states [17].

There are multiple ways to build an enriched library, including using fast physicochemical filters or drug likeness [19], ligand structure similarity searching to known drugs [20], empirical high-throughput docking and scoring [21], genetic algorithm for focused descriptor active space (GAPDAS) [22], or geometric constraints (i.e. the location of hydrogen bonds) [23]. The most common strategy is to take advantage of the computational speed and intuitive understanding of filtering for drug-likeness based on the physicochemical properties [8]. Drug likeness is widely quantified by Lipinski rule of 5, which requires that an orally active drug should have no more than one violation of the following criteria: 1) maximum of five hydrogen bond donors, 2) less than or equal than 10 hydrogen bond acceptors, 3) molecular weight of less than 500 Da, and 4) an octanol-water partition coefficient of not greater than five [19]. If a ligand violates two or more conditions, they are expected to have poor absorption, distribution, metabolism, and excretion (ADME) properties and would not be considered suitable drugs. Some studies also show that applying physicochemical filters in enriching the compound library can improve both computational efficiency and hit rate of the docking process, including but not limit to virtual screening for G-protein coupled receptors[22], non-peptide malignant brain tumor (MBT) repeat antagonists[24], and non-nucleoside HIV-1 reverse transcriptase inhibitors[25].

An example of the use of structure-based drug design is in the design of allosteric ligands. Allosteric ligands bind at a binding site that is spatially distinct from the native ligand’s binding site. Over the past 20 years, designing and synthesizing allosteric molecules has experienced considerable growth in the pharmaceutical industry and medicine as a whole [26]. Allosteric drugs offer several advantages over the corresponding orthosteric ligands targeting the same protein, including better selectivity, favorable physicochemical properties, and improved chemical tractability for synthesis[27]. However, the general physicochemical properties for allosteric compounds are not well understood. The third chapter of this thesis focuses on elucidating the physicochemical properties which can differentiate compounds targeting allosteric sites from competitive compounds.

Due to the iterative process of SBDD, acquisition of compound collections may be needed for a new iteration. These new compounds preferably should cover new chemical space. Depending on the stage or purpose of the project, new compounds with wide chemical diversity may be desirable to enhance breadth of the library. Alternatively, compounds in nearby chemical space may be needed to increase the depth of coverage. The precise method used to quantify molecular similarity or diversity depends upon the context of the question. Examples where a new set of compounds which covered a wider range of chemical space is required is described by Eckert et al[28]. For example, finding bioactive peptide-like molecules requires exploration of highly structural diverse data set[29]. Adding new set of compounds covering a deeper depth of chemical space is necessary to extract of structure-activity relationship (SAR) information[30]. The characterization the association between structure and activity is usually non trivial and require many structure similar compounds [31]. Especially for activity cliff that is structurally similar compounds having large potency difference, many similar compounds are needed to better characterize the activity landscape around activity cliff [32]. To both increase breadth and depth of chemical space in molecular dataset, visualization of chemical similarity with associated bioactivity becomes vital to examine the relevance between compound structures and particular biological values. This is especially true now with continual expansion of biological data sets. In the second Chapter of this thesis, we developed a new ready-to-use visualization tool, ChemTreeMap to handle data mining for multidimensional, heterogeneous, biological chemical data sets.

The above discussion has focused on developing datasets of ligands for SBDD, however, the target proteins and binding sites are also of importance. The target binding sites must be annotated and have high-quality three-dimensional coordinates available. Target binding sites, the other input of SBDD, are required to have 3D structures of desired protein targets and annotations of binding site locations. In recent years, structural genomics of protein families have provided a major boost to SBDD. These projects have identified and solved new structures of known and potential drug targets [33]. To annotate the binding site location, a protein target in complex with a compound can be used to determine the possible drug interaction by molecular basis of existing protein ligand interactions. Alternatively, binding site prediction or protein structure prediction methods are necessary if there isn’t a solved complex.

Target binding sites might be altered by permanent gene mutations, which vary across individuals[34]. Most variants take the form of single nucleotide polymorphisms (SNPs). Non-synonymous SNPs (nsSNPs) cause changes to a protein through mutation of an amino acid or the introduction of a premature stop codon which affects the stability and/or function of the protein. These structural changes may have detrimental effects on drug binding, thereby inducing drug resistance which has become one of the biggest challenges in drug development in recent years [35]. The binding sites, which are easily mutated by nsSNP, are at high risk to confer drug resistance. Chapter 4 of this thesis focuses on finding the role that structure and nsSNPs play in understanding the general molecular basis of disease. Additionally, these factors may be of use to account for the possibility of drug resistance when choosing drug targets.

All of the above-mentioned challenges in structure-based drug design are related to determining the chemical space of potential drug candidates that target at protein binding sites. Solving those issues requires solid statistical analysis for assessing the similarities or differences between chemicals, exploration of the relation between biological data and chemical compounds, and understanding the impact of nsSNPs on protein structures.

This dissertation focuses on knowledge-based approaches to improve SBDD based on both the construction of the ligand libraries and analysis of the target binding sites used. First, we propose a visualization tool (ChemTreeMap) to mine for correlation between chemical structure and biological data. The performance of the visualization tool does not depend upon assumptions of activity and chemical structure and thus may be applicable to a wide range of question involving small molecule/drug binding. The tool can be used to construct and explore ligand libraries. With the help of this generic tool, this thesis reports the findings on discriminating properties between allosteric and competitive molecules. To remove ligand redundancy in the data set, clustering is conducted at two levels, protein level and ligand level. The significance of properties is assessed by Wilcoxon test and bootstrapping to obtain confidence interval. The result can be used to build chemical libraries biased towards allosteric regulators. Thirdly, the target binding sites of compounds may be vulnerable to nsSNPs. The location of nsSNP on protein structures are accessed by taking account the heterogeneous distributions of nsSNPs across protein families. The result can be used to imply the probability and consequence of nsSNP on new target binding sites. Extension and future direction for these projects are discussed in the last chapter.

Table 1-1. Some commonly used free chemical databases for screening.

Database	Description	Type of Chemicals	Last updated	No. of compounds*	Website
PubChem	PubChem contains biological activities of small molecules.	commercial, bioactive, research	3/29/2016	88 million	https://pubchem.ncbi.nlm.nih.gov
ChemSpider	ChemSpider integrates and links compounds from ~500 data sources to present a comprehensive view of freely available chemical data.	commercial, bioactive, research	3/29/2016	45 million	http://www.chemspider.com
ZINC	ZINC contains commercially-available compounds in ready-to-dock, 3D formats.	commercial, bioactive, research	2/4/2015	35 million	http://zinc.docking.org/
CoCoCo	CoCoCo collects commercial compounds from eight chemical vendors and standardizes their structural information	commercial	9/7/2012	7 million	http://cococo.isof.cnr.it/
eMolecules	eMolecules provides an online molecule search engine for public domain structures from vendors worldwide.	commercial	3/1/2016	5.0M	https://www.emolecules.com/
ChEMBL	ChEMBL is a manually curated chemical database of drug-like bioactive molecules with binding, functional and ADMET information.	bioactive	2/1/2016	2 million	https://www.ebi.ac.uk/chembl/
ChemDiv	ChemDiv provides a diverse and highly specialized compound selections, which have been extensively validated in biological assays.	bioactive, commercial	7/16/2013	1.5 M	http://www.chemdiv.com/
BindingDB	BindingDB provides binding affinities between small, drug-like molecules and drug-targets, with crystal structures.	bioactive, research	9/1/2015	0.5 million	https://www.bindingdb.org
DrugBank	Drugbank contains extensive drug information, including chemical, pharmacological and physiological data, with detailed targets information, including sequence, function, and gene location.	bioactive	2/15/2016	8,206	http://www.drugbank.ca/

* Approximate numbers

1.2 Visualization for aiding structure-based drug design

Many steps of SBDD, including construction and exploration of compound collections , involves visualization of compound structures and biological attributes, like data mining, chemical library design, acquisition of compound collections [36], prioritization of molecules in a compound library for biological evaluation [37], ADME profiling [38], multi-objective optimization (specificity, pharmacodynamics, toxicity, etc.) [39].

The visualization is based on a key concept in medicinal chemistry, that molecules with similar structures will usually have similar physicochemical properties, including biological activity [40]. Visualization is represented by a graph, including but not limited to networks, trees, scatterplots, and bars. Therefore, similar molecules are grouped closely in the graph and biological properties are mapped to graph attributes. Informative visualization of structurally heterogeneous compounds with activity data has steadily gained more interest as the size of biological datasets and the amount of data has dramatically increased[41]. The visualization of heterogeneous chemical space is not limited to SBDD, but some can be applied to understand the relation between chemical structures and their general biological actions, like off-target effects [42], synergy effects [43], or drug repositioning [44], [45]. The second chapter of the thesis describes the development of a tool to present heterogeneous chemical space with multivariate properties in a tree structure. Two fundamental questions need to be answered for the graph representation: 1) how to quantify the similarity between molecules; 2) how to represent the similarity in a graph. This section discusses methods that are commonly used to address these questions. It details other available tools and their advantages and disadvantages that have inspired the development of ChemTreeMap described in the second chapter of this thesis.

1.2.1 Molecular similarity

Any visualization starts with the calculation of molecular similarity, which is a subjective concept which varies with the context and details of the molecules in the chemical database [46]. Chemoinformaticians, chemists, and medicinal chemists each may have a different perspective on what molecules are similar [47]. If the similarity method is chosen properly according to the cheminformatics problems, a more accurate model(s) of the chemical activity of the compounds would be built[47]. The visualization tool must be able to handle variety of similarity metrics. It is especially difficult to quantify and assess the similarity. In order to provide an objective measurement, computational methods are necessary to generate a consistent value for molecular similarity. The similarity calculation usually involves three steps: 1) representing a molecule structure by chemical and/or structural features; 2) weighting each feature by their contribution; and 3) quantifying the similarity between two representations. A similarity value between 0 and 1 is obtained. A value of 1 where 1 represents complete identity of two representations of the structure and may not require two structures to be identical, while a value of 0 indicates no overlap in the representation[47].

There are various ways to represent a molecule. Two common methods of representing the molecule are with a chemical representation or a structural representation. Chemical representation uses physicochemical properties, like solubility, boiling point, log P, molecular weight, electron density, or even reaction information to describe the molecule[47]. On the other hand, structural representation utilizes structural features, like shared substructures, ring systems, topologies, chemical environment of atoms, etc. Sometimes, new descriptors are created from combining multiple chemical or structural features using mathematical functions. Structural representation is more popular, because the experimental data for physicochemical properties is far from complete and many of them are only predictions based on the structural features [48]. The representation of the molecule is expressed as a vector of numbers, which indicate the presence of most relevant chemical or structural features.

Molecular representations have two categories: two dimensional (2D) and three dimensional (3D). 2D structure can be captured by molecular graphs, whose vertices correspond to the atoms and edges correspond to chemical bonds. Compounds are inherently three-dimensional and their molecular conformations can provide more information than 2D. 3D structure can be determined either by experiment (X-ray crystallography) or by calculation from molecular graphs and optionally followed by optimization. The 3D features include surface area, 3D shapes[49], [50], 3D distance between atoms[51], energy, dipole moment, and so on. 2D features that are directly calculated from molecular graph can be physicochemical properties (like atom counts, ring counts, formal charges, etc.) or graph properties (like fragment and topological atom environment). Many studies try to directly compare two molecular graphs, but the computational complexity of these algorithms make them infeasible to large scale molecular similarity analysis [52], [53]. 2D features are more widely applied in comparison with 3D, including extracted from molecular graphs. Two-dimensional features are preferred, because 1). 2D Molecular graphs can contain annotation for conformational and stereo chemical information; 2) chemists are more comfortable with molecular graphs than 3D conformation. 3) The determination of biologically active conformations in vast ensembles of test compounds is uncertain [47]. In contrast, 2D features are much more robust and often have good prediction power for bioactivity [54], [55].

The thesis adopts a 2D representation, extended connectivity fingerprint (ECFP), which is currently one of the most popular fingerprints for similarity searching[56]. ECFP can capture many different atom environment features by recording local bond topology, which provides more details than other fingerprints with predefined fragment libraries [47]. The ECFP generation process is as follows. ECFP begins with assigning atom identifiers to each heavy atom of the molecule by a hash function, which captures atom local information (e.g. atomic number, connection count, type of bond, etc.) (Figure 1-2). After that, the algorithm combines the current atom identifier with the identifiers of neighboring atoms within a certain diameter to generate new identifiers for each fragment. The iteration will continue until a specified diameter is reached. The larger the diameter can capture greater structural details. The produced identifier list will be mapped to a fixed-length binary representation by another hash function[47].

The similarity of two representations (i.e. ECFP) can be quantified by Tanimoto Coefficient (Tc) [57], [58], Dice [59], or Tversky [60] similarity coefficient. Tc is generally defined as the number of features shared by A and B divided by the total number of features of A or B:

$$Tc(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Dice can be expressed as:

$$Dice(A,B) = \frac{2|A \cap B|}{|A| + |B|}$$

Tversky coefficient is an asymmetric similarity measure defined as:

$$S(A,B) = \frac{|A \cap B|}{|A \cap B| + \alpha|A - B| + \beta|B - A|}$$

Here, $|A - B|$ denotes the number of features existing in A but not in B. $\alpha, \beta \geq 0$ are parameters. The Tversky coefficient can be viewed as a generalized form of Tanimoto Coefficient and Dice coefficient. Setting $\alpha = \beta = 1$produces the Tanimoto Coefficient. Dice coefficient can be produced by setting $\alpha = \beta = 0.5$. The second Chapter uses Tc, because Tc is by far the most widely used method in chemical informatics and computational medicinal chemistry, because of the simplicity of implementation and speed [47].

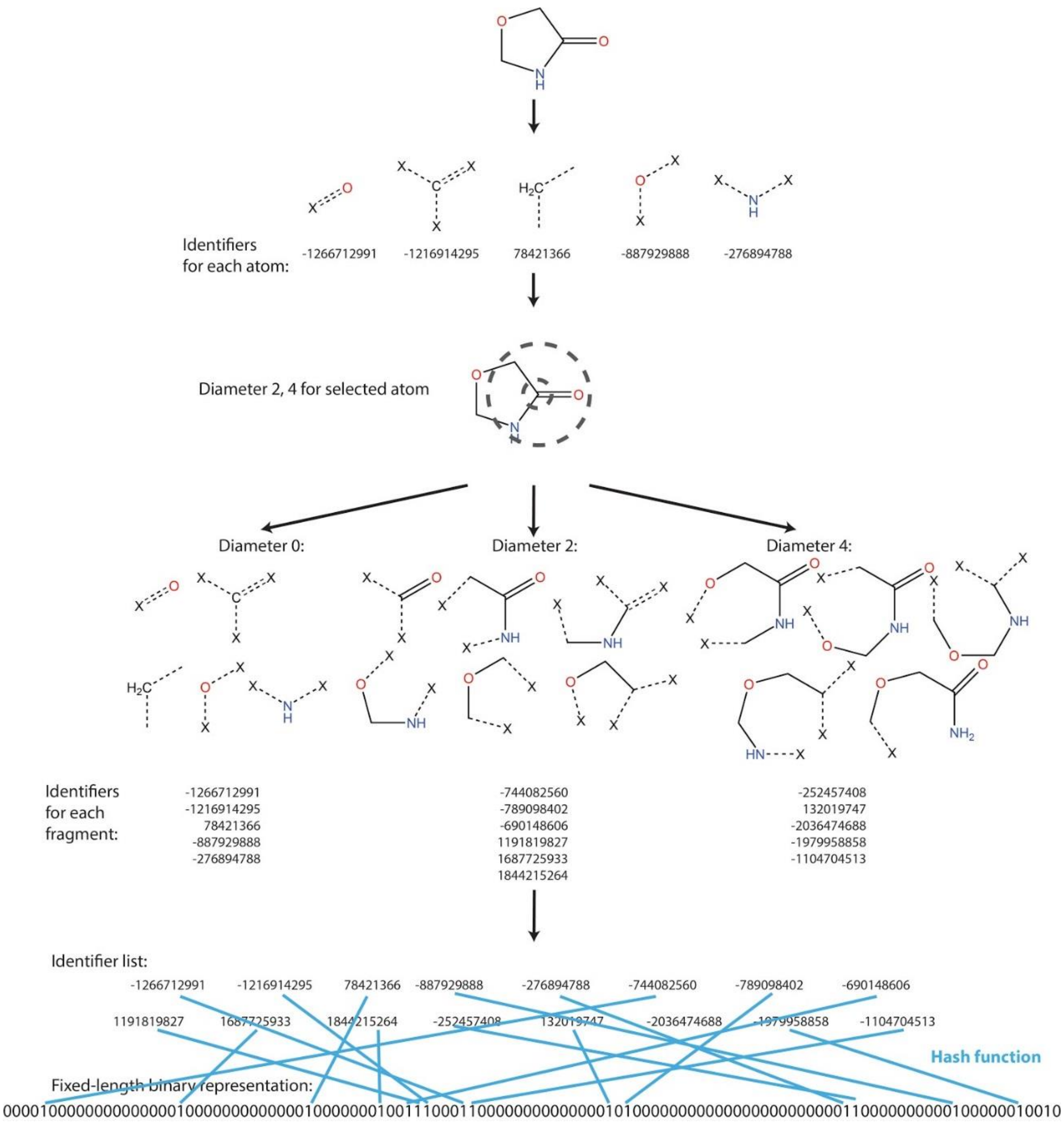


Figure 1-2. Generation of ECFP with diameter 4.

1.2.2 Graph representations of molecular data sets

Graphical representations of structure-based drug design data are increasingly required in order to comprehend and manage the large amounts of data and capture complex relationships involved in drug discovery processes[61]. The representation of large compound libraries is made possible through the use of numerical similarity methods. To generate a graphical summary [62], [63] on the compound sets, many methods are developed to visually capture patterns among molecular structures with other properties (e.g. biological activity). These methods be roughly grouped into three categories: Network-like, Tree-like and Scatterplot-like, each having been optimized for specific purposes. For example, various graph layouts have been designed for Structure−Activity Relationships [64]–[66], representing large chemical databases [67], analyzing complex multidimensional data [7], exploring sub- and superstructure relations [61], [68]–[74], and finding common structural features [75].

Table 1-2. Example chemical space visualization methods developed in recent years.

	Network-like	Tree-like	Scatterplot-like
Descriptor-based	CSNs[76], Similarity Graphs[64]	Data Warrior[66], Similarity-Potency Trees[51]	Synergy-map[43], CheS-Mapper[77], [78], MQN-Mapplet[53], neural networks based[65], Synergy Maps[30]
MCS*-based	Hasse Diagrams-based[68], HierS[55]	Scaffold Hunter[70], inSARa**[61], Core Trees[57], Fragment Hierarchies[58], SCONP[59], The Scaffold Tree[60]	The Molecule Cloud[75]

*MCS: maximum common structure

** inSARa starts with a network layout and convert to a tree structure by calculating the minimum spanning tree(MST) using the Kruskal algorithm[79].

Network-like graphs can represent relationships between many different elements in coordinate-free spaces, which can capture complex interaction relations by explicitly accounting for all pairwise relationships [41]. Each node can either represent a single molecule in the data set (i.e. CSNs [76]), or a substructure shared by a set of molecules, (e.g. HierS[69] and Hasse-based graph[68]). Many network principles (like homophily) and analysis gained from the study of social networks can be applied to compound network and used to recognize community structures [80]. One challenge for network-like graphs is that each node in the network does not have any hierarchical order, which hides the relationship between molecule groups. Both HierS [69] and Hasse-based graphs [68] try to represent the hierarchical order of each node in subnetworks by assigning a number to each node. But, if one node shares a common scaffold with other nodes in terms of different substructures, those methods will assign the node multiple values for the hierarchical order. The hierarchical order assignment becomes harder with larger networks [61]. Another challenge is that optimization of the network topology, which is essential for detecting community structures, is computationally intensive [80]. Thus, a number of heuristic optimization methods have been introduced with varying levels of effectiveness [81]–[84].

Tree-like graphs satisfy the demand of hierarchical organization. The hierarchical order of nodes can easily be defined. The root is the top node representing level 0. The child node (level N) is directly connected to another existing node (level N - 1) when moving away from the root node. Each child has only one parent, which may have many children (siblings). Each non-leaf node can represent the most common structure of all its end leafs.

Existing tree-like methods all generate a collection of trees, with each tree representing molecules in a nearby neighborhood. Some methods focus on interpretation of structure-activity relationships [65], [66], while others concentrate on the relations among ring structures [70], [73], [74] and consensus fragments [71], [72]. Comparing with all those methods, inSARa is feasible for a large set of molecules [61]. inSARa constructs similarity networks first, and then simplifies networks to trees by calculating the minimum spanning tree (MST) [79]. The final output are trees with hierarchical organization, which is easier to interpret than the complex network. However, MST won't guarantee the edges between every two nodes can still represent their similarity. As much edge information is not encoded in trees, molecules with multiple parents may not be well positioned. Since the tree layout could become hard to read with more than thousands of nodes, many layout algorithms are developed to optimize the layout, like force-directed layout[85], spectral layout[86], hierarchical graph drawing[87] and so on. ChemTreeMap, presented in the second Chapter, organizes molecules in a single tree structure, for viewing large scale heterogeneous compounds in a single connected graph and positioning molecules based on their hierarchical relations.

Scatterplot-like graphs are coordinate based space representation in which each data point designates a compound on the basis of physicochemical properties or shared structure. It can reveal the coverage of the molecule set on selected descriptors. The most common way for generating coordinates is a dimension reduction method, which could be principal component analysis (PCA) by MQN-Mapplet and CheS-Mapper, multidimensional scaling (MDS) by CheS-Mapper, t-Distributed Stochastic Neighbor Embedding (t-SNE) by Synergy Map, or neural network based methods [88]. Another way to use the plotting space is tiling common structural features by structural images, like the Molecular Cloud, which indicates frequency of substructures by image size and biological activity by image color.

1.3 Ligand chemical space for allosteric compounds

Structure-based drug design strategies frequently focus on the targeting of a protein’s active site, but this can be difficult if the physicochemical properties of this site are such that it cannot be targeted with typical drug-like molecules[89]–[93]. One strategy to avoid this issue is to design allosteric modulators targeting other sites, which may have improved physicochemical properties. The third chapter discusses most distinguishable physicochemical properties to differentiate allosteric compounds from orthosteric competitive compounds. The usage of physicochemical properties analysis is discussed in 3.1 and allosteric mechanism and compound development are discussed in 3.2.

1.3.1 Rules derived from physicochemical properties analysis

Screening large compound libraries demands a large amount of computational resources and time. Researchers become more interested in focused libraries that are generated using target relevant information and its known active compounds [94]–[98]. The most widely used methods for building an enriched library are physicochemical rules and undesirable functional group searches [99]. The underlying assumption of using physicochemical-based descriptors to design focused libraries is that compounds with similar structures are more likely to have similar interactions with the same targets [100]–[103]. Similar chemical structures also lead to similar physicochemical property ranges [100]–[103].

The most popular physicochemical-based filters are drug likeness rules, such as Lipinski's rule of 5[19], Oprea lead like [104]–[106], and Veber rules [107]. These rules are based on simple physicochemical properties, like the number of hydrogen bond donors and acceptors, the number of rotatable bonds, molecular weight, logP, polar surface area [19]. They have been extensively adopted to reduce the size of the screening library and bias towards drugs or lead-like compounds. More specific rules has been developed for a few target classes like GPCRs [108] and kinases [109]. The desired rules should cover the biologically relevant chemical space for a particular class of targets and minimize the size of screening libraries [109].

There are other approaches to design a target-focused library: 1) scaffold-based design [109]–[112], and 2) two-dimensional and three-dimensional pharmacophore similarity-based design [113], [114]. Those approaches try to find molecules with structures similar to existing active compounds. Physicochemical property filters outperform these approaches in designing target-focused library on selecting compounds with novel structural scaffolds, because highly diverse scaffolds can share similar physicochemical properties.

1.3.2 Distinguishing allosteric modulators by physical chemical properties

Ligands exerting their effects via protein binding sites may be grouped into two classes, those that target the active site binding natural ligands, termed orthosteric modulators, and those that target distal sites, termed allosteric modulators. This thesis work aims to find general trends for physicochemical descriptors to differentiate allosteric modulators from orthosteric competitive modulators. The ability to differentiate these classes would provide deep insight into chemical properties underlying allosteric mechanisms and facilitate the building of a molecule libraries biased for the discovery of allosteric modulators.

Despite much efforts, scientists frequently encounter difficulty finding orthosteric competitive compounds with high selectivity, drug-like physicochemical properties, and that are non-toxic [27]. For example, it is difficult to achieve selectivity among compounds targeting class C GPCR’s orthosteric site is difficult as these compounds tend to be small in size, which limits the addition of specificity-improving R-groups [115]. The chemical property of natural ligands of class B GPCRs is outside of classic oral drug-like space [116]. As allosteric ligands bind at a site with less evolutionary pressure for conservation across a protein family, it is likely that inhibitor of these sites can be engineered to have high levels of selectivity. The drawback to reduced evolutionary-conservation is that these sites may be more likely to develop resistance mutations. Allosteric sites, being a secondary binding pocket, are not necessarily as conserved as orthosteric sites across paralogs and orthologs [117]. Especially for viral inhibitors where the genetic mutation and selection is rapid, the use of allosteric modulators might induce higher mutation rate. For example, the non-nucleoside reverse transcriptase inhibitors (NNRTI) is well known for a quick onset of drug resistance [118]. For systems with rapid genetic mutation and selection, non-allosteric inhibitors may be more effective due to their similarity to the natural substrate [119]. Nevertheless, allosteric modulators are still effective for targets with less evolutionary pressure.

The interests of allosteric drug development steadily increase after the clinical success of benzodiazepines. In 1955, benzodiazepines are discovered as an allosteric drug, which potentiate the effect of neurotransmitter γ-aminobutyric acid (GABA) by targeting the ionotropic GABAA receptor [120]. After benzodiazepines, more protein targets are investigated to develop allosteric ligands, including GPCRs, ion channels, kinases, caspases, phospholipase, and so on [27].

1.4 The impact of mutations on protein-ligand binding sites

As mutations conferring resistance are frequently encountered after drugs enter the clinic, it is wise to consider mutations at the early stages of structure-based drug design. This can lead to the discovery and development of candidate drugs which are less likely to select for mutations, or which may bypass known mutations [121]. Moreover, the structure and interaction information gleaned from these analyses may provide insights into the molecular basis of disease-associated or neutral mutations.

Variation is the spice of life[122]. For two individuals, the genomes could be roughly 99.9% identical, but these 0.1% differences account for phenotypic variation among individuals, including susceptibility to disease[122]. While the sequencing of human genome was a great milestone, how to interpret the downstream impact of variation is still ongoing. The variation of genotype can impact phenotype by many molecular mechanisms. This thesis focuses on the impact of SNP on protein-ligand binding sites.

1.4.1 The impact of SNP and possible association with diseases

Small scale mutations occur in one of three ways: single nucleotide polymorphisms (SNPs), insertions, or deletions[123]. SNPs are the most frequently encountered variant, which may occur in both coding and noncoding region of the genome. While some SNPs are known to be associated with a number of Mendelian diseases and complex genetic disorders[124], it is difficult to determine the biological impact of previously undefined SNPs. For example, SNPs in noncoding regions may affect protein transcription rates and expression levels. SNPs in coding regions can be either synonymous or nonsynonymous substitutions, depending on whether the mutation triggers a change in the amino acid sequence. Even for synonymous SNPs in coding regions, there may be phenotypic effects through altering the rate of translation of a protein[125]. Non-synonymous SNPs (nsSNPs) clearly result in changes to a protein through mutation or the introduction of a premature stop codon, which should affect the stability and function of protein, but this information is impossible to determine from the sequence-level changes alone.

1.4.2 nsSNP and the possible impact on protein structures

A protein exists as a delicate balance of structure and function, and dramatic changes may occur when this balance is offset. For example, local conformational changes induced by nsSNP mutations could introduce hydrophilic residues into protein core [126], disrupt of hydrogen bonding patterns, or alter protein secondary structural elements like beta-sheets, resulting in a change to the biological function of the protein. These changes often have phenotypic implication and increase the possibility of disease [127]–[129].

Except for nsSNPs that introduce dramatic change in size or physicochemical properties, the impact of most nsSNP is hard to interpret. A number of prediction methods to predict SNP effects have been developed [130]–[132], which analyze sequence conservation and provide a score postulating the risk of an individual SNP. Several possible improvements to prediction methods has been proposed, including incorporating information like protein stability and Gene Ontology [133]–[135]. On the other hand, measuring the distribution of nsSNPs on a global scale across many protein structures is also important to understand the molecular mechanism of nsSNPs. The distribution can provide insight for general understanding of disease mechanism and potential impact on drug binding. Yet, a number of questions still remain. For example, are nsSNPs concentrated on a specific location on proteins, such as binding sites? Are disease-associated nsSNPs statistically significantly in their difference from other benign nsSNPs?

1.4.3 Existing databases of protein-ligand interactions

Accurate annotation of protein ligand interaction sites is critical to understand the impact of SNPs on the binding site. Existing databases with protein-ligand complexes can be used to characterize the interaction between molecular complexes with the goal of interpreting the mechanism of various enzymatic processes. The worldwide Protein Data Bank (PDB) contains the largest collection for experimentally determined biomolecular structures [124]. As of March 2016, more than 117,000 structures have been deposited into PDB.

A number of secondary, derivative databases have been developed from the PDB to summarize some specialized information For example, PDBsum was developed as a pictorial summary of the key information on each macromolecular structure [136], Relibase+ to provide an easy accessible web-browser interface for searching protein–

ligand interactions [137], Het-PDB Navi for navigating heterogens (compounds or ligands) in PDB and provide statistics for interactive residues when enough data are available [138], scPDB for pictorial analysis of binding sites of PDB [139]. Collectively, these databases allow visual exploration of protein-ligand complexes on a large scale.

In SBDD, the high quality protein-ligand complexes with energetic properties (like binding affinity) is more desirable especially for developing scoring function for docking of small molecules. In recent years, many sets of protein-ligand complexes have been collected and have online web server for searching and downloading. Binding MOAD [140], [141], PDBbind [142], and BioLiP [143] are three most outstanding databases curated for high-quality, protein-ligand complexes, augmented with the inclusion of binding affinity data. Other similar databases, including LPDB [144], PLD [145] and AffiDB [146] are much smaller in size and haven't updated since their original release.

Binding MOAD, PDBbind, and BioLiP were all developed independently at the University of Michigan, Ann Arbor. Binding MOAD, developed in the Carlson lab, has 25,771 protein-ligand structures and 12,440 unique ligands for version 2014 [147]. The aim of Binding MOAD is to build and maintain the largest-possible collection of high-quality, protein-ligand complexes available from the PDB. PDBbind, originally developed under Shaomeng Wang, currently has 29,008 protein-ligand complexes and 11,376 small-molecule ligands for version 2015 [148]. The original focus is on developing scoring function and searching ligand substructure [142]. The project is currently maintained by Renxiao Wang at the Shanghai Institute of Organic Chemistry [149]. In general, Binding

MOAD and PDBbind share similar semi-manually curated procedure. The key difference is that: 1) PDBbind has various types of molecular complexes, including protein-ligand, nucleic acid-ligand, protein-protein, and protein-nucleic acid complexes; Binding MOAD doesn’t have any nucleic acid structure; 2) PDBbind does not have strict control about the protein structure resolution; Binding MOAD only takes structures with resolution 2.5 A or better; 3) PDBbind limits the entries with only one ligand in the crystal structure and excludes complex with only cofactors bound [142], [150]. BioLiP is developed in Yang Zhang’s lab in 2013 as a resource for protein-ligand binding site prediction [143]. Most of BioLiP binding affinity data are extracted from Binding MOAD and PDBbind [143].

1.4.4 Determining residue locations on protein’s structure

Depending on the specific location of mutations (surface, binding site, or protein core), nsSNPs may have distinctive biological consequences. These locations can be discriminated based on the degree of solvent exposure or even accessible surface area (ASA) can be used to quantify the interacting surface area between individual residue and solvent molecules and locate the residue on protein structures.

Many methods have been developed to calculate the ASA, such as ACCESS [151], DSSP [152], ASC [153], NACCESS [154], and GETAREA [155]. ASA was first described by Lee & Richards in 1971[156]. Accessible surface of an atom is the sphere surface of a solvent molecule, on which the points are required not penetrating any other atoms of the same molecule. The radius of the sphere surface is the sum of the Van der Waals (VDW) radius of the atom and of the solvent molecule. In 1973, Shrake and Rupley proposed a more efficient “rolling ball” algorithm to calculate ASA [157]. The algorithm is a numeric method that draws a mesh of points with equal distance from each atom of the protein and counts the number of points that can be accessible by solvent. The points are checked against the VDW surface of neighboring atoms to determine whether they are on the surface of the protein. Those points on the protein surface may still not be solvent accessible. Another check against the vdW radius plus a probe distance (the radius of the solvent, 1.4 A for a water molecule) determines whether the point is solvent accessible. ASA is calculated by the number of accessible points multiplied by the portion of surface area each point represents, which is determined by the van der Waals surface of that atom divided by the number of points created on the mesh. The result of the algorithm depends on the choice of the 'probe radius' (could be the radius of a water molecule) and the definition of vdW radius of each atom. For example, hydrogen atoms may be implicitly included in the atomic radii because some structures may not have hydrogen atoms.

This thesis uses DSSP by Kabsch and Sander in 1983, which implement an algorithm similar to Shrake-Rupley algorithm with triangle area instead of equal distance grid [152]. DSSP is commonly used to generate training data for predicting solvent accessibility [158]–[161], secondary structure [162]–[164] and aromatic-backbone NH interactions[165] from protein sequences. DSSP has also been applied to generate structural features for mutation impact studies [134], [166]. Another widely used implementation is NACCESS by Hubbard and Thornton [154]. This thesis is interested in two categories of residues “surface” or “non-surface”. The result is quite consistent between NACCESS and DSSP in terms of determine surface residues (> 5 Å SASA). Many other algorithms have attempted to improve the speed of surface area calculation and provide other structural metrics, but they mostly follow the same definition of ASA proposed by Lee & Richards [167], [168].

Chapter 2. ChemTreeMap: An Interactive Map of Biochemical Similarity in Molecular Datasets

2.1 Abstract

Motivation: What if you could explain complex chemistry in a simple tree and share that data online with your collaborators? Computational biology often incorporates diverse chemical data to probe a biological question, but the existing tools for chemical data are ill-suited for the very large datasets inherent to bioinformatics. Furthermore, existing visualization methods often require an expert chemist to interpret the patterns. Biologists need an interactive tool for visualizing chemical information in an intuitive, accessible way that facilitates its integration into today's team-based biological research.

Results: ChemTreeMap is an interactive, bioinformatics tool designed to explore chemical space and mine the relationships between chemical structure, molecular properties, and biological activity. ChemTreeMap synergistically combines extended connectivity fingerprints and a neighbour-joining algorithm to produce a hierarchical tree with branch lengths proportional to molecular similarity. Compound properties are

shown by leaf color, size, and outline to yield a user-defined visualization of the tree. Two representative analyses are included to demonstrate ChemTreeMap's capabilities and utility: assessing dataset overlap and mining structure-activity relationships (SAR).

Availability: Examples of ChemTreeMap may be accessed at http://ajing.github.io/ChemTreeMap/chemtreemap.html and http://ajing.github.io/ChemTreeMap/example. ChemTreeMap may be accessed at http://ajing.github.io/ChemTreeMap/. Code for the server and client are also available in the Appendix B (pg #162) and Appendix C (pg #215) and https://github.com/ajing/ChemTreeMap.

2.2 Introduction

Researchers in the field of bioinformatics are frequently tasked with exploring the relationship between chemical structures and their potential biological actions. For example, one can predict the interaction between a small molecule and a corresponding protein target, given the chemical structure [169]. The degree of similarity between chemical structures can also indicate a potential for drug repositioning [44], [45] or off-target effects [170]. A molecules' ability to inhibit protein-protein interactions may also be predicted [171]. In order to discover such interactions, researchers must harness large databases, including PubChem [172] and ChEMBL [173], which contain vast amounts of heterogeneous biological and chemical data. The size and complexity of these datasets necessitate automated tools to explore available chemical space in order to determine chemical relationships and predict potential interactions.

Exploring chemical space frequently requires a comparison of the number of shared chemicals among multiple databases [45], an understanding of the similarity within a compound series [44], [45], [170]series, and an analysis of how compound structures give rise to a specific biological action [171]. Such analysis begins with the visualization of molecular datasets. There are general visualization strategies such as Venn diagrams [45], networks [45], heat maps [170], and clusters [44]. However, those strategies have limited utility, depending on the biological question and how detailed the analysis must be.

Graphical tools have been developed for cheminformatics that group structurally similar molecules together and display information about molecular structure and bioactivity [61], [65], [66], [70], [74]. The previous methods usually require users to choose selection rules [65] and tune parameters [61]. This can limit the utility of these tools for large, diverse sets of data that are the hallmark of bioinformatics. Those tools can also require domain experts to effectively use them, which limits their extension to a larger user base.

To overcome these limitations, we have developed ChemTreeMap, an open-source tool for visualizing compound similarity coupled with associated biochemical information. We have carefully selected techniques for calculating molecular similarity and representing chemical space to satisfy the needs of bioinformaticians. The tool uses standard procedures, requires no tuning parameters, and allows users to interactively explore a molecular dataset. ChemTreeMap organizes molecules into a hierarchical tree based on chemical similarity, much like phylogenetic trees that are commonly used in biology. This provides a familiar framework for scientists to view and access desired information.

Users can map any property of interest to the graph’s leaf attributes (i.e. color, size, and border color). This facilitates an on-the-fly, customized exploration of the relationships between molecular structure and other properties. ChemTreeMap’s organization reflects the similarities of molecules at various levels and in different chemical series. It does not rely on any assumption about the similarity cutoff. Users can explore the branches to understand the similarity across the nearby molecules. The branch lengths quantify the difference in features, which is particularly useful for structurally diverse datasets. Longer distances between chemical families highlight more diverse regions of chemical space.

To illustrate ChemTreeMap’s utility, we describe two practical applications: the visualization of chemical overlap between molecular datasets and the extraction of structure-activity relationships (SAR).

2.3 Methods

Organization and visualization of a molecular library requires three considerations: first, how to represent a molecule, second, how to quantify the similarities between different molecules, and lastly, how to represent these similarities graphically. Each ChemTreeMap is then completed by coloring the resulting tree to convey biochemical information.

2.3.1 Representation of the molecules

Our primary molecular descriptors are stereochemistry-aware, extended connectivity fingerprints (ECFP6#S). These are topological descriptions that capture large, recursive, circular neighborhoods around each atom. This method identifies the functional groups in each molecule, and it is quick to calculate, which makes it well suited for large molecular datasets [56]. ChemTreeMap also calculates atom-pairs fingerprints [174] as an alternative metric. Others can be added easily, such as MACCS keys [175], topological torsion fingerprints [176], and 2D-pharmacophore fingerprints [177].

A potential concern about fingerprints is that their pair-wise comparisons may not be optimal for a global description of the data [61]. Fortunately, ChemTreeMap's display uses a global, hierarchical organization to convey information at the local level and in the overall patterns across the data.

2.3.2 Construction of the chemical similarity tree

The similarity of two molecules is calculated by a Tanimoto coefficient (T_c) [178], which refers to the number of chemical features they share in common divided by the union of all features (a % similarity that ranges in values from 0 to 1). T_c was chosen because of it is fast, easy to implement, and widely used in chem-informatics and drug-discovery software. Branch lengths in ChemTreeMap are inversely proportional to the T_c between the molecules, where shorter branches show high similarity and longer indicate greater diversity.

To build the hierarchical tree, we chose the Neighbor-Joining algorithm (NJ, see Appendix A (pg #148)) [179]. The NJ algorithm is widely used in building phylogenetic trees for large and diverse sequences [180], [181]. It has been mathematically proven that given a correct input distance matrix, the output tree and branch lengths from NJ will also be correct [182]. Furthermore, NJ does not rely on any parameter tuning, making the tree construction more robust. RapidNJ has a best-case running time of $\boldsymbol{O}(\boldsymbol{N}^2)$ and at worst $\boldsymbol{O}(\boldsymbol{N}^3)$ [183]. It is an agglomerative joining method that follows:

1. $\boldsymbol{D}$ is an $\boldsymbol{N} \times \boldsymbol{N}$ distance matrix, where each element $\boldsymbol{D_{ij}}$ = T_c(i,j) (*i*, *j* represent two molecules from the set of $\boldsymbol{N}$ molecules)
2. The average distance from molecule *i* to all other molecules *k* is:

$$\boldsymbol{u_i} = \sum_{\boldsymbol{k}} \boldsymbol{D_{ik}} / \boldsymbol{N-2}$$

3. With $\boldsymbol{N}$ average distances, we create the $\boldsymbol{Q}$ matrix, with elements:

$$\boldsymbol{Q_{ij}} = \boldsymbol{D_{ij}} - \boldsymbol{u_i} - \boldsymbol{u_j}.$$

4. Find the *i, j* with the smallest value $\boldsymbol{Q_{ij}}$.

5. For that *i*, *j* pair, an imaginary ancestor node a is created to replace *i* and *j*. The distance between a and *i*, *j* is:

$$\boldsymbol{v_i} = \boldsymbol{0.5} \times \boldsymbol{D_{ij}} + \boldsymbol{0.5} \times (\boldsymbol{u_i} - \boldsymbol{u_j})$$

$$\boldsymbol{v_j} = \boldsymbol{0.5} \times \boldsymbol{D_{ij}} + \boldsymbol{0.5} \times (\boldsymbol{u_j} - \boldsymbol{u_i})$$

6. ***D*** is then updated by replacing *i* and *j* with an ancestral node a. The distances between a and all the other nodes *k* are:

$$\boldsymbol{D_{ak}} = \left(\boldsymbol{D_{ik}} + \boldsymbol{D_{jk}} - \boldsymbol{D_{ij}}\right)/2$$

7. Keep updating until the last two nodes are joined.

2.3.3 Clustering of Molecules for Very Large Datasets

For huge datasets, such as ChEMBL [173], BindingDB [184], and ChemBank [185], the molecules need to be clustered by similarity to reduce the size of the task. An interactive display of millions of leafs is not possible with current technology. For tractability, ChEMBL and ChemBank were each independently sorted into their own 4000 clusters. Because BindingDB is smaller, its molecules were clustered into 2000 sets. All clusters were represented by the molecule with the highest sum of T_c across the subset, the molecule most similar to all the others in the group. The size of the leaf for that center is proportional to the number of molecules in the group. It would be possible to produce a ChemTreeMap for each cluster, but wading through thousands of smaller trees is not practical.

MiniBatch-KMeans was chosen for clustering, which has low runtime complexity $\boldsymbol{O}(\boldsymbol{N})$, memory usage $\boldsymbol{O}(\boldsymbol{N})$, and relatively low error [186]. The RDKit (*http://www.rdkit.org*, accessed 28 October 2015) fingerprint was selected for this initial clustering, which is an implementation of a Daylight-like fingerprint. The task of finding nearest neighbors is relatively easy, and this alternate fingerprint is fast and saves memory. With MiniBatch-Kmeans and RDKit fingerprint, the clustering step for ChEMBL – the largest dataset – took less than 5 hours to run on a machine with 16G memory (i3-2100 CPU @ 3.10GHz**).** In comparison, the maximum dissimilarity method [187] implemented in PipelinePilot [188] takes more than a week to cluster ChEMBL, and an algorithm with average runtime complexity $\boldsymbol{O}(\boldsymbol{N} \times \boldsymbol{logN})$ such as DBScan still takes more than 3 days.

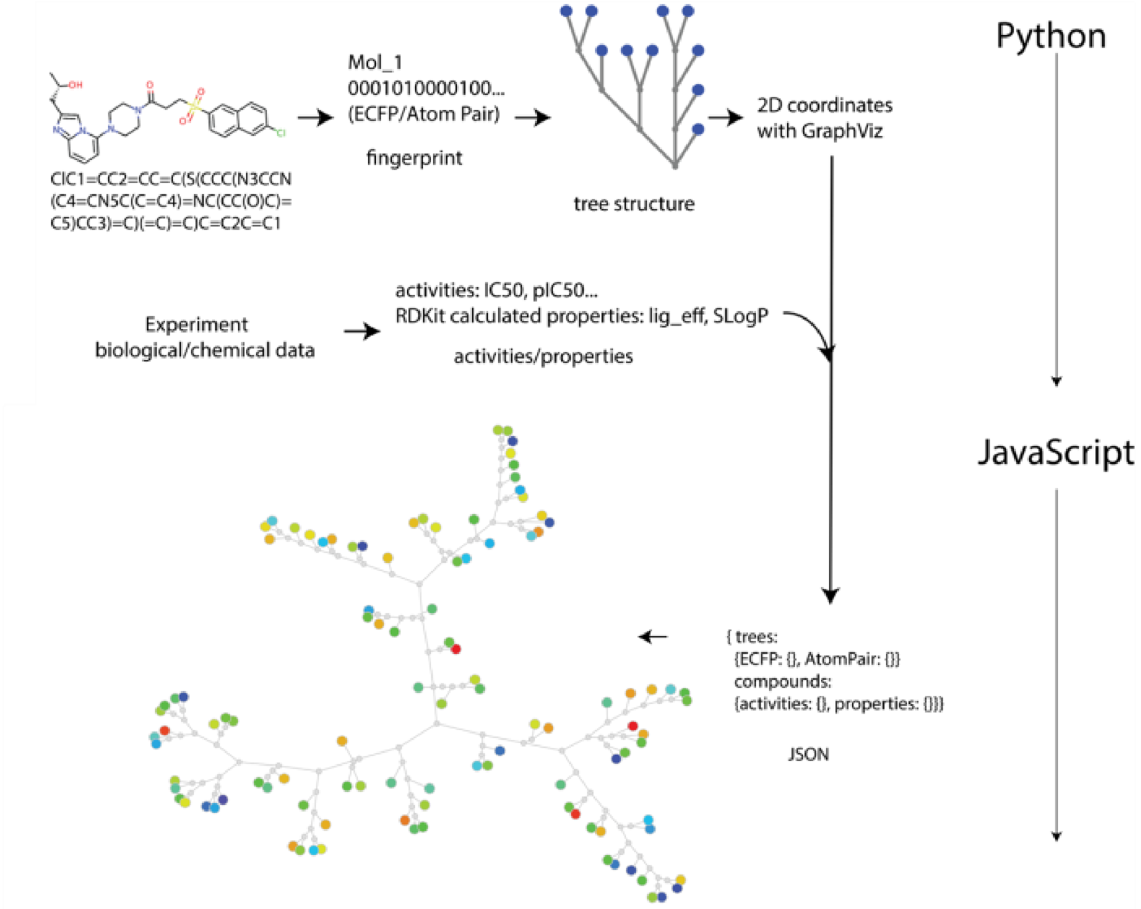


Figure 2-1. ChemTreeMap work flow. Each compound (encoded as a standard SMILE string) and its biological data are processed in steps to yield a JSON file, which is then used by a JavaScript App to create the graphic tree map (clearly three independent chemical series in this sample). Specifically, ECFP6#S and Atom-Pair fingerprints are calculated as two options for the T_c used in building a tree structure. By default, LE and SlogP are calculated for each molecule and its activity data. Tree structures and compound data with associated properties are required in the JSON file.

2.4 Implementation

The application has a client-server paradigm [189], in Figure 2-1, which allows online access to the results and facilitates sharing data with colleagues. The server performs tree construction, including fingerprint calculations, similarity calculations, neighbor joining, layout optimization, pIC_{50}, ligand efficiency (LE = 1.37×pIC_{50}/heavy-atoms, for rough conversion to ΔG_{bind}/HA), and SlogP calculations [190]. The code for the server is provided in Appendix C (pg #215). All output from the server processes are packaged into a JavaScript Object Notation (JSON) file[191][190][31] for client input. The client process displays the tree structures and supports functions, such as searching by ID and changing leaf attributes: circle size, colored border, and biochemical metric mapped to the circle. Layout optimization uses a multi-scale version of dynamic, spring-model layouts implemented in GraphViz [192]. This allows users to actively pull and reorient nodes/branches to improve the visual layout in a local area. The code for the client process is provided in Appendix B.

ChemTreeMap is a web-based tool, which is easy to set up on any computer with Python 2.7. GraphViz and RapidNJ are freely available online for major operating systems [183], [192]. The graphics can be viewed on any computer with an HTML5 capable browser (tested on Google Chrome Version 46, Internet Explorer 11, and Safari

9.0). We recommend Google Chrome for displaying ChemTreeMaps of thousands of molecules because of the computational intensity of its force-directed graph. Speed is also dependent on the client hardware.

2.4.1 ChemTreeMap can be easily extended with more features

A straightforward option to add functionality is to input additional data for each molecule, as columns in a tab-delineated input file (e.g. SMILE string, data 1, data 2, etc.). The program will present the additional data as options in drop-down menus for the user to map that information onto the ChemTreeMap leafs through display options (color, border, circle size).

A more powerful alternative is to extend the functionality in ChemTreeMap's TreeBuild.py file. Currently, the chemical properties of LE and SlogP are calculated using RDKit in TreeBuild.py, and developers can very easily add 55 additional descriptors from RDKit. For more descriptors, we recommend adding data from MOE [193] or PaDEL [194] with the simple tab-delimited option. Any in-house, custom analysis is best added directly in the python code if it will be used frequently.

2.4.2 Comparing ChemTreeMap to other SAR methods

We compared ChemTreeMap with four freely available programs: Similarity-Potency Tree (SPT, [65], Data Warrior [66], Scaffold Hunter [70], and CheS-Mapper [77], [78]. SPT is the method most similar to ChemTreeMap. It also uses T_c and ECFP6#S fingerprints to calculate chemical similarity, but it uses a different type of tree display and network analysis. SPT has a compound-centric view, focusing on potency and a limited set of nearest neighbors in chemical space. Their tree structures rely on a chosen reference compound. Similar molecules are not necessarily grouped together in some cases.

Data Warrior is an open-source program with highly interactive graphical views and interesting statistical analysis. It focuses on similar neighborhood relations only and ignores similarity relations below a certain threshold. Such methods may not capture enough structural breadth for a diversified dataset.

Scaffold Hunter identifies chemical cores for each molecule and associates them in a hierarchy based on medicinal chemistry rules. A tree-like output is used, but the local SAR of individual molecules sharing the same scaffold is not displayed.

CheS-Mapper organizes compounds in three dimensional (3D) space, where the compounds similarity is encoded in the spatial distance between them. This software employ multidimensional scaling methods to generate the 3D coordinates for each molecule. These methods usually do a decent job of grouping similar molecules closely together. However, the available space is not used efficiently and it only accept numerical descriptors. In the visualization, similar molecules tend to be overcrowded, which may make the difference between similar molecule structures difficult to recognize.

We did not have access to a maximum common substructure (MCS) method for comparison. MCS are another way to organize molecules into groups, based on pre-defined chemical functionalities and novel pattern matching [195]. Like Scaffold Hunter, MCS techniques identify common core patterns across a dataset. inSARa is a recently introduced MCS method [61] that uses reduced graphs to create tree-based output like ChemTreeMap, SPT, and Scaffold Hunter. Its final representative substructures are sensitive to many parameter choices [61], which is where ChemTreeMap and inSARa differ. Our generalized T_c-based distances have no tunable parameters, other than the choice of fingerprints. A tunable method has its own benefits, and we see inSARa as a complementary method to ChemTreeMap, SPT, and Scaffold Hunter.

2.5 Datasets

ChemTreeMap is applicable to a wide range of datasets with various levels of complexity and sizes. To demonstrate ChemTreeMap, we have chosen diverse biomolecular datasets ranging from thousands to millions of molecules. To show chem-data overlap, we use some of the largest datasets with bioactivity data: ChEMBL v. 20 [173], BindingDB [184], and ChemBank [185].

For SAR examples, we assembled chemical datasets for the four protein targets shown in Table 1. Clotting factor Xa (FXa), cyclin-dependent kinase 2 (CDK2), p38α MAP kinase (p38α), and cytochrome P450 3A4 (CYP3A4) were chosen because their SAR are well characterized [196], [197], and they have been used in previous studies for visualizing chemical data [61], [65]. The data for FXa, CDK2, and p38α are from BindingDB. CYP3A4 data (bioassay AID:884) was pulled from high-throughput screening (HTS) data in PubChem [172]. All four systems have a large range of inhibition data. The average T_c for each set is under 0.2, which indicates high chemical diversity.

Each of the SAR datasets was prepared using the following protocol:

1. For inclusion, a molecule must have an IC_{50} or pIC_{50} for bioactivity. For PubChem data, only molecules in the “Active” category was kept for analysis.

2. If there were multiple activity data for a molecule, the average of the IC_{50} was used (i.e. no repeat chemical structures).

3. The ionization state and tautomer for each chemical structure were determined using the “wash” utility in MOE 2014 [193].

Table 2-1 Datasets used for SAR analysis.

Protein Target	Target class	Number of Molecules	Maximum pIC_{50}	Minimum pIC_{50}	Ave T_c
FXa	protease	2161	10.7	3.0	0.172
CDK2	kinase	1923	9.5	2.9	0.141
p38α	kinase	5139	10.4	2.9	0.167
CYP3A4	oxidoreductase	6837	8.9	4.1	0.115

2.6 Results and Discussion

2.6.1 Comparing chemical diversity of large datasets

When more data is needed for a project, information that covers new chemical space is typically preferred. Depending on the project, very wide diversity may be desirable to enhance breadth, or new compounds in nearby chemical space may be needed to increase the depth of coverage. ChemTreeMap can identify both types of chemical similarity/diversity.

Existing methods count the number of duplicate molecules between the sets [45], or they map chemical features to a 2D scatter plot with overlapping regions, based on principal component analysis or multidimensional scaling [43], [67]. These methods can fail to convey enough information on the structural similarity within/between the molecular sets.

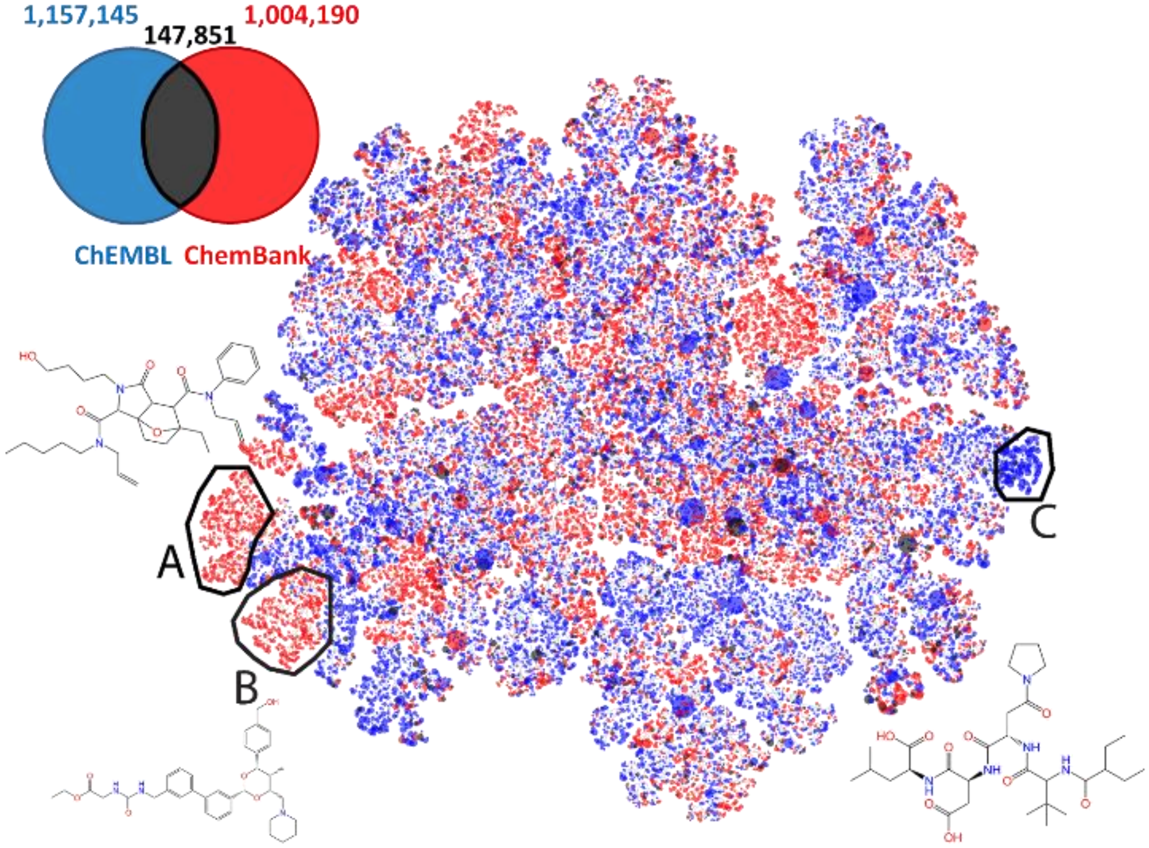


Figure 2-2. The addition of ChEMBL (blue) and ChemBank (red) adds breadth of chemical space. ChemTreeMap details the regions of chemical similarity and diversity for both sets. Shared molecules are in black clusters. Regions A and B contain molecules only from ChemBank. Region C contains molecules only from ChEMBL. Representative molecules are shown. Leaf size (circles) is proportional to the number of molecules in each cluster from the initial processing step for large databases. The view is scaled out to show the entire data space. Full details can be seen with ChemTreeMap by zooming in on any region.

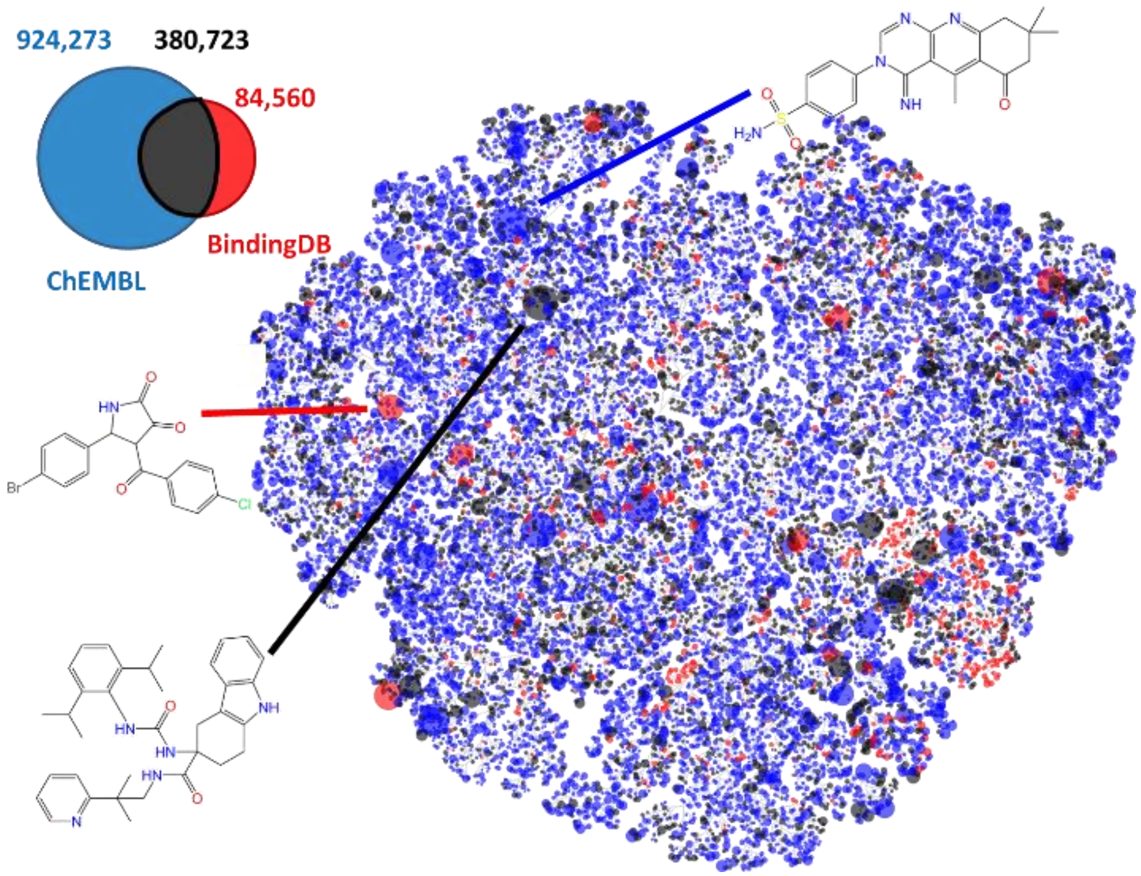


Figure 2-3 The addition of ChEMBL (blue) and BindingDB (red) adds depth to chemical space: Shared molecules are in black. Representative molecules are shown. The view is zoomed out to show all data.

Our examples for comparing large sets revolve around ChEMBL [173]. It is a dataset with ~1.3 million chemicals with significant biochemical annotation from the literature (~50 journals). We compared ChEMBL to ChemBank and BindingDB to show datasets that represent collections with more breadth vs more depth, respectively. ChemBank [185] is a set of ~1.15 million molecules constructed from chemical HTS studies. Many compounds in ChEMBL are not from HTS studies and will not appear in ChemBank. Though many HTS studies eventually appear in the literature, the full data for every compound rarely appears in a final publication. Therefore, we knew that many molecules would appear in one but not the other. We did not know how similar/different the chemical sets were from one another, but we expected significant populations of ChEMBL-only and ChemBank-only compounds with many in similar chemical space.

The traditional Venn diagram in Fig. 2 shows that the overlap is relatively small as expected; 11.3% of ChEMBL molecules are the same as 12.8% of ChemBank's compounds. To show the chemical similarity and diversity, a more detailed display is needed. ChemTreeMap highlights regions where the chemical space is unique to ChemBank (red branches A and B) and to ChEMBL (blue branch C). Though some branches contain red and blue leafs in similar chemical space, combining the two sets primarily increases the breath of chemistry structures overall.

ChEMBL [173] and BindingDB [184] both curate molecules from the biochemical literature. BindingDB focuses on protein targets with crystal structures in the Protein

Data Bank. It enhances the structures with experimental binding data for many ligands. ChEMBL is a major source of data for BindingDB's molecules, but BindingDB includes data from 12 biochemical journals not covered by ChEMBL. Together, ChEMBL and BindingDB's shared efforts provide ~60 journals-worth of data to the scientific community.

Based on their construction, a large overlap is expected for ChEMBL +BindingDB (Fig. 3). About 82% of the molecules in BindingDB are also in ChEMBL (black nodes). No large branches are dominated by BindingDB. Instead, many of ChEMBL's branches contain new molecules from BindingDB. Fig. 3 shows that this augmentation to ChEMBL from BindingDB adds depth of coverage, specific to drug-like space. (Note: The ChEMBL-only branches are for targets that do not have a protein-ligand crystal structure, a requirement for BindingDB).

2.6.2 Options for displaying more information

A distinct advantage for ChemTreeMap is the ability to display multiple layers of information simultaneously through leaf color, outline, and size. For example, molecules with good solubility and high LE are desirable for drug leads. Fig. 4 shows how ChemTreeMap adds more data through outlines. Molecules with good solubility (low number, red) or good LE (higher number, red) can be easily recognized in the graph. Users could extend ChemTreeMap with more SAR metrics, like SALI scores (Structure-

Activity Landscape Index, Guha and Van Drie, 2008) or PAINS alerts (Pan Assay Interference compounds, Baell and Holloway, 2010).

ChemTreeMap uses a broad color scheme purple-blue-cyan-green-yellow-red. Previous studies have scaled the colors green-yellow-red to fit the max/min range in each dataset [61], [65]. Having twice the number of colors allows us to show a larger span of properties with better clarity. It also allows users to easily compare different datasets because the static colors always indicate the same IC_{50} in any ChemTreeMap. The spectrum basically covers the full range of affinities obtained in biological assays, where inhibitors vary in IC_{50} from ~100 µM (purple: $pIC_{50} \geq 4$) to ~1 nM (red: $pIC_{50} \leq 9$). For LE, the range is 0 (purple) to 0.5 (red) kcal/mol-HeavyAtom, which covers 90% enzyme activity data from our previous study [198]. For SlogP, the range is set from -5 (red) to 5 (purple). Users can switch between attributes while inspecting the branches, and the features are interactive, e.g. navigating, dragging, or zooming to see details.

2.6.3 ChemTreeMap can be used to extract SAR information

For the rest of our examples, the molecules were not clustered, and all leafs represent a single compound. Traditionally, SAR information is deduced by analyzing molecules from several chemical series, often incorporating predictive statistical models [65].Interpretation can be strongly dependent on the experience of the chemists, and these paradigms are limited in the number of compounds one can analyze. ChemTreeMap provides an intuitive, robust, and effective way to extract SAR information from a chemical library. ChemTreeMap does not rely on any assumption about the similarity cutoff. Its hierarchical organization of molecules, based on structural similarity, facilitates the identification of SAR hotspots and activity cliffs. The distance between leafs highlights the (dis)similarity between any pair of molecules.

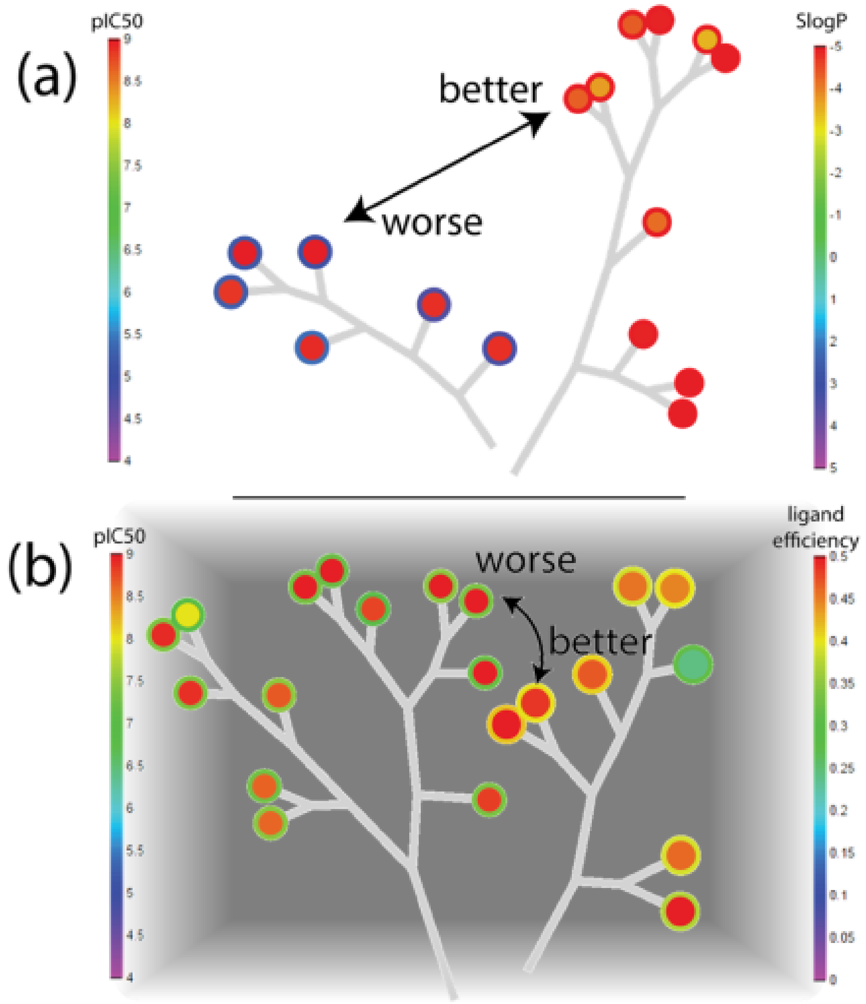
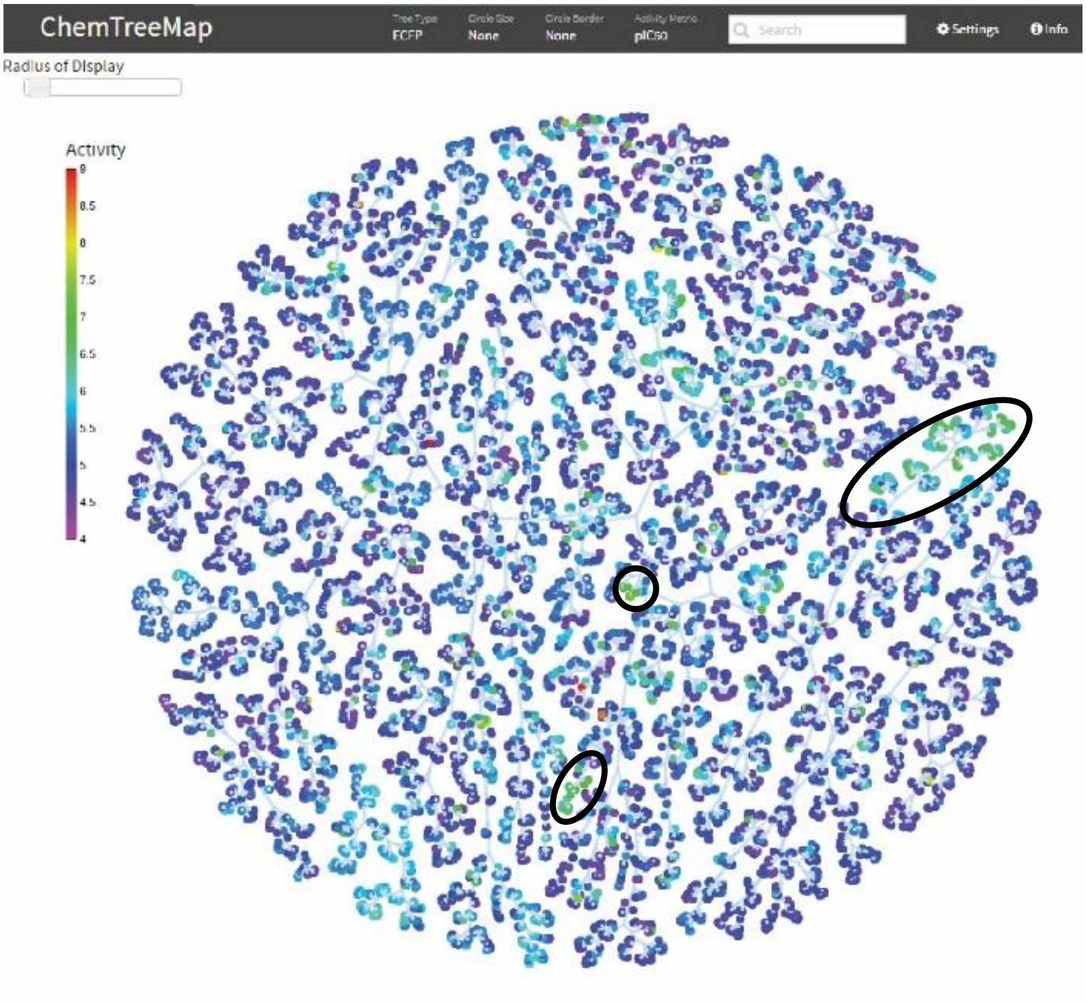


Figure 2-4 Multi-layer information: (a) pIC$_{50}$ color inside the circles and SlogP mapped to the outline color. The left branch shows a common pitfall of good affinity but poor solubility of the molecules. (b) Gray background added to make outlines more visible. pIC$_{50}$ color inside the circles and LE mapped to outline color. Most leafs have high potency, but those with green outlines are larger molecules with lower LE. These molecules are less "efficient" because they have the same binding affinity but need more contacts to the target.



Tree Type ECFP

Circle Size None

Circle Border None

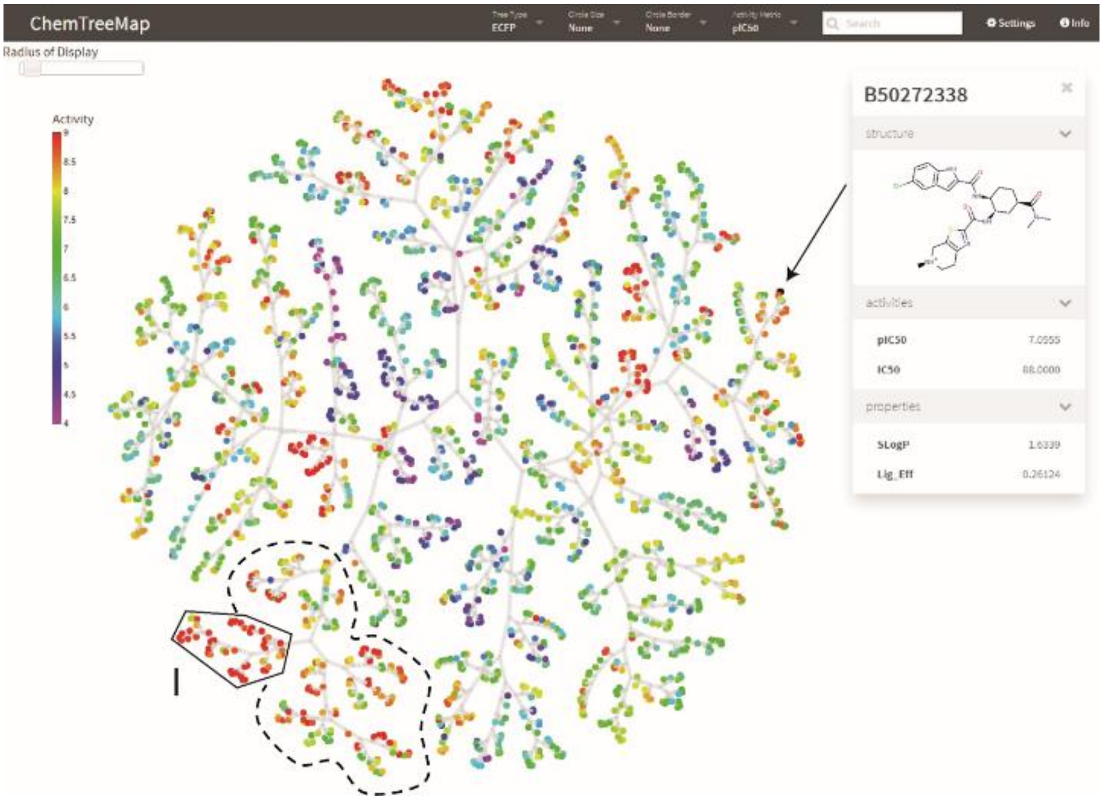
Activity Metric pIC50

Figure 2-5 The ChemTreeMap for CYP3A4 (6837 molecules) shows a few compact regions of chemical space with moderate activity (green leafs, highlighted by black ovals). Compounds in these groups have a higher potential for drug development. There are also a small number of one-off, strong inhibitors that are likely HTS error (rare orange and red leafs). The top bar contains drop down menus for ECFP or Atom-Pair fingerprints and the leafs (circle size, circle border, activity metric, settings, and info). The tree is dynamic using a force-directed graph with control of the "Radius of Display" (see Supplementary Information). The color bar shows the activity metric. Each color represents one level of potency (pIC$_{50}$): which is 4 (purple, IC$_{50}$ = 100 µM, indigo), 5 (blue), 6 (cyan, IC$_{50}$ = 1 µM,), 7 (green), 8 (yellow), 9 (IC$_{50}$ = 1 nM, red).

2.6.3.1 HTS

HTS is a screening technology that tests thousands of molecules in a biochemical assay. Hits are typically inhibitors of an enzymatic assay, but many different assays exist that test a variety of effects. ChemTreeMap displays can help with interpreting the data. ChemTreeMap has an advantage when exploring large, heterogeneous datasets from HTS because of its speed, hierarchical structure, and broad color range.

Most of the compounds used in HTS are inactive, which can be seen by the predominance of blue and purple in the CYP3A4 ChemTreeMap, Fig. 5. HTS data is notorious for many false positives and false negatives. One of the hallmarks of true positives is finding many similar molecules displaying moderate activity like those marked in Fig. 5. These regions indicate chemical space with potential for further development. Groupings of hits can be a small, 5-molecule sub-tree or a large branch of 30+ compounds. Of course, proper statistics are critical to assessing signal-to-noise in HTS data. One way that ChemTreeMap could be easily extended for a custom HTS application is to map statistical significance of each compound’s signal onto their leaf outlines (e.g. z-score).



Tree Type: ECFP
Circle Size: None
Circle Border: None
Activity Metric: pIC50

B50272338

pIC50	7.0555
IC50	88.0000
SLogP	1.6339
Lig_Eff	0.26124

Figure 2-6. ChemTreeMap for 2161 FXa inhibitors: Sub-tree I contains high potency molecules; the dashed region highlights many modifications in the nearby chemical space cause a drop in potency. Inspection of those compounds shows that the most detrimental changes involve removing a positive charge from an essential functional group. The color bar follows the same activity pattern as in Figure. 2-5. The upper-left region shows that clicking on a leaf provides the chemical information for that molecule.

2.6.3.2 SAR of FXa

Fig. 6 is the ChemTreeMap of our SAR set for FXa. A typical SAR set explores a more focused area of chemical space (inhibitors in this example). The molecules have a range of activities that change with the structural features. Branches dominated by strong inhibitors in red clearly denote specific chemistry linked to high potency.

Sub-tree I is the largest region of the ChemTreeMap with high activity. The neighboring, dashed region shows how potency starts to decrease as larger chemical modifications are made. Fig. 7(a) shows a magnified view of sub-tree I from Fig. 6, with LE data shown on the outlines of the leafs. By inspecting the neighboring molecules, we can identify shared chemical features that are correlated with high activity. Fig. 7(b) shows that all molecules A-L in sub-tree I share a 2-(4-(N,N-dimethyl carbamimidoyl)benzamido)-N-(pyridin-2-yl)benzamide core, marked in gray. *ChemTreeMap makes a rather complex chemical analysis and description straightforward for informaticians because the pattern is clear from the visual display.* An “activity cliff” is easy to identify for compounds J and K. Activity cliffs occur when a small chemical modification leads to a large change in activity [199]. ChemTreeMap makes it easy to find molecules in very close proximity with large color changes. Several other structural features are also seen in the full dataset, such as activity switches and SAR hotspots also identified using inSARa [61], but for brevity, we focus on region I.

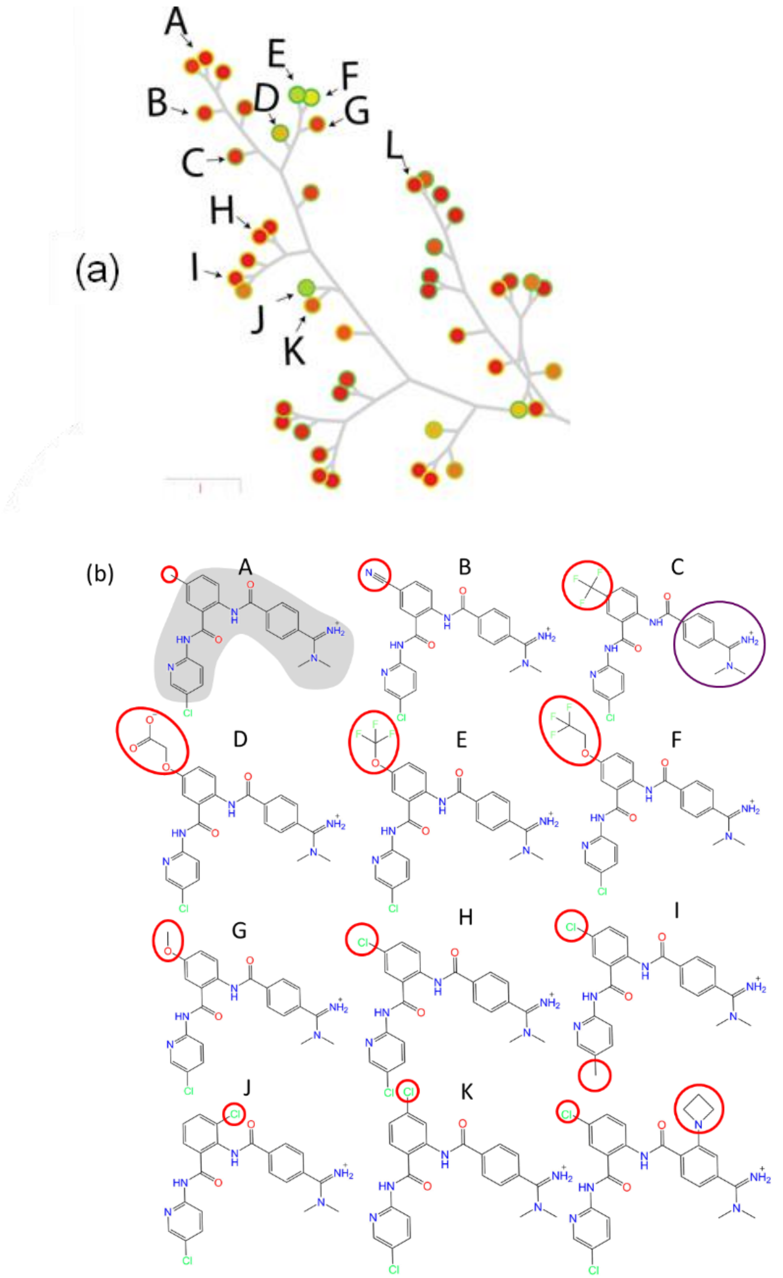


Figure 2-7 (a) Sub-tree I from Fig. 6 with activities as fill color and LE as outline color. Inhibitors with larger functional groups have less favorable LE values (compounds D-F). (b) The chemical structures of representative molecules A-L are shown. The common core is marked in gray, and the differences in the functional groups are noted with red circles. The functional group circled in purple is an essential feature for these FXa inhibitors, and L is a representative of a nearby sub-tree that modifies this part of the molecule.

Molecules A-H/J/K differ by small chemical changes on the central ring. Molecule I differs from H by a methyl group in place of a chlorine at the bottom of the molecule. I is a “descendent” of H because they are most similar, but I is one step away from the common structure of the other compounds. The addition of a four-membered ring in L is a large chemical change that places it in a nearby sub-tree separate from A-K.

Fig. 8 provides the analyses of the FXa dataset with SPT, Data Warrior, and Scaffold Hunter. Here, we focus the discussion on compounds from sub-tree I. SPT is a state-of-the-art method for cheminformatics. An accurate method should show agreement with SPT and improvement where possible. Fig. 8(a) shows that the same molecules from sub-tree I were a large structural feature in the highest-ranked tree found with SPT, which is in good agreement with our finding that sub-tree I is the largest contiguous region of high activity. The majority of compounds in Fig. 8(a) are from the two marked regions in Fig. 6. SPT constructs a different visual tree that uses the compounds as nodes. All molecules with T_c >0.4 of the root compound are organized into levels. Each level shows all molecules with T_c >0.55 to the root-molecule above it. SPT organizes each level by increasing activity from left to right, but removes the structural relationships between molecules in the same level. The ordering directs the user toward the most active compounds, but it may obscure structural features of the SAR. For instance, H/J/K only differ by the location of one chlorine atom, yet they appear unrelated in Fig. 8(a).

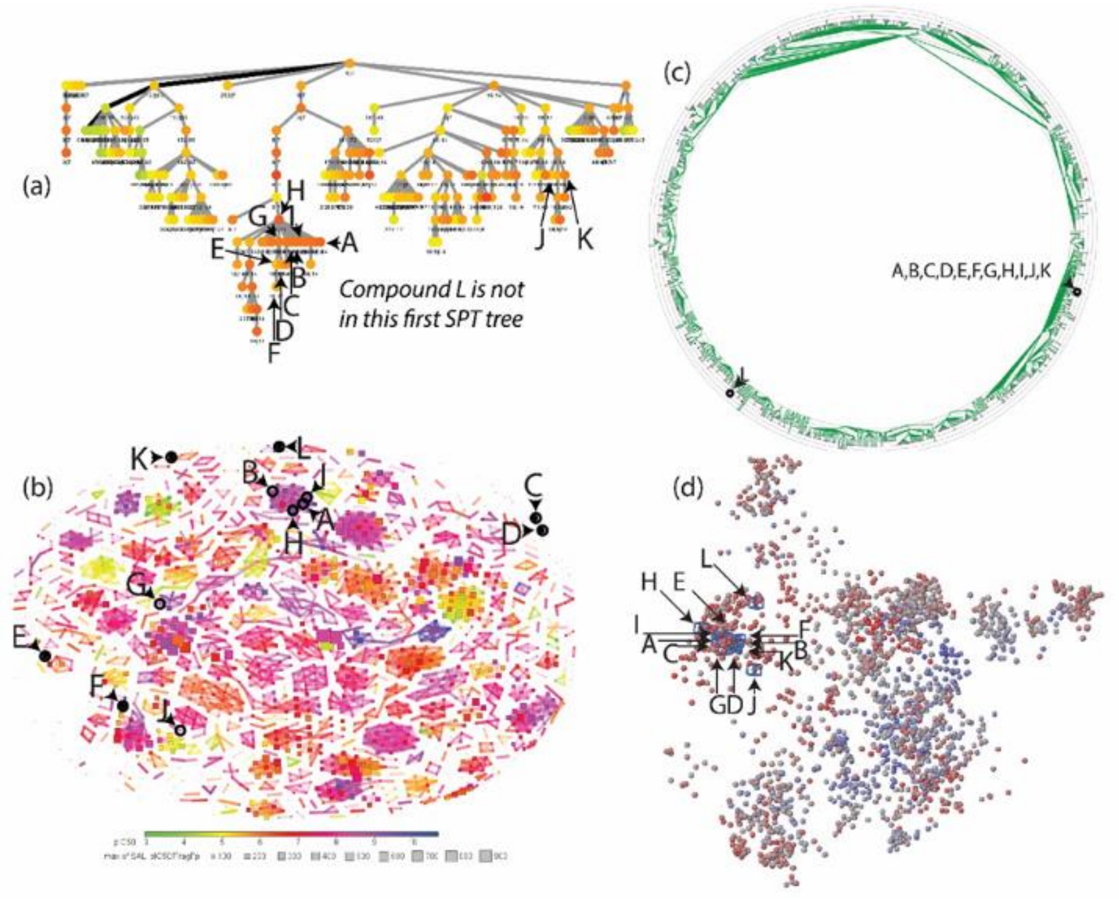


Figure 2-8. Sub-tree I of the FXa dataset visualized using three other tools: (a) SPT, (b) Data Warrior, (c) Scaffold Hunter, and (d) CheS-Mapper.

In Fig. 8(b), Data Warrior spreads A-L across its entire network, with few connections between them. Several of the molecules are shown as lone data points. In Fig. 8(c), Scaffold Hunter correctly identifies the common substructure of A-K, but molecule L is placed in another tree with cyclobutane as a root. The relationship between the A-K scaffold and the L scaffold is lost. The impact of small modifications on activity is not presented, but the information is necessary for SAR studies. In Fig. 8(d), CheS-Mapper groups A-L together, but as many molecules shared similar compound features with A-L, this region becomes too crowded to reveal the relationship between structure changes and activity.

Clearly, future enhancements for ChemTreeMap should include scaffold information. The ancestor nodes that serve as branch points are prime locations for showing the shared common substructure of the descendent molecules. It would also be an appropriate point in the graph to display SAR alerts based on the patterns of activity in the descendent branches.

2.6.3.3 SAR hotspot in CDK2 data

An SAR hotspot is a collection of similar molecules displaying a wide range of potency. Finding these series are important for SAR, but they can also facilitate “patent busting.” ChemTreeMap makes finding SAR hotspots easier by having 1) a wide color range for activity and 2) branch lengths between nodes that are proportional to chemical similarity. Short branches with many colors are candidates for SAR hotspots.

In Fig. 9(a), the ChemTreeMap for CDK2 contains sub-tree II, an SAR hotspot. In Fig. 9(b), two sets of molecules can be seen in different branches because of their chemical similarity: A/B (ether substitutions) and C-E (alkane substitutions). These are separate from F-H which have larger chemical differences between each other and the rest of sub-tree II. If these molecules were ordered by potency, these groupings would be more difficult to identify. However, the preference for a branched, 4-atom substituent would still be clear.

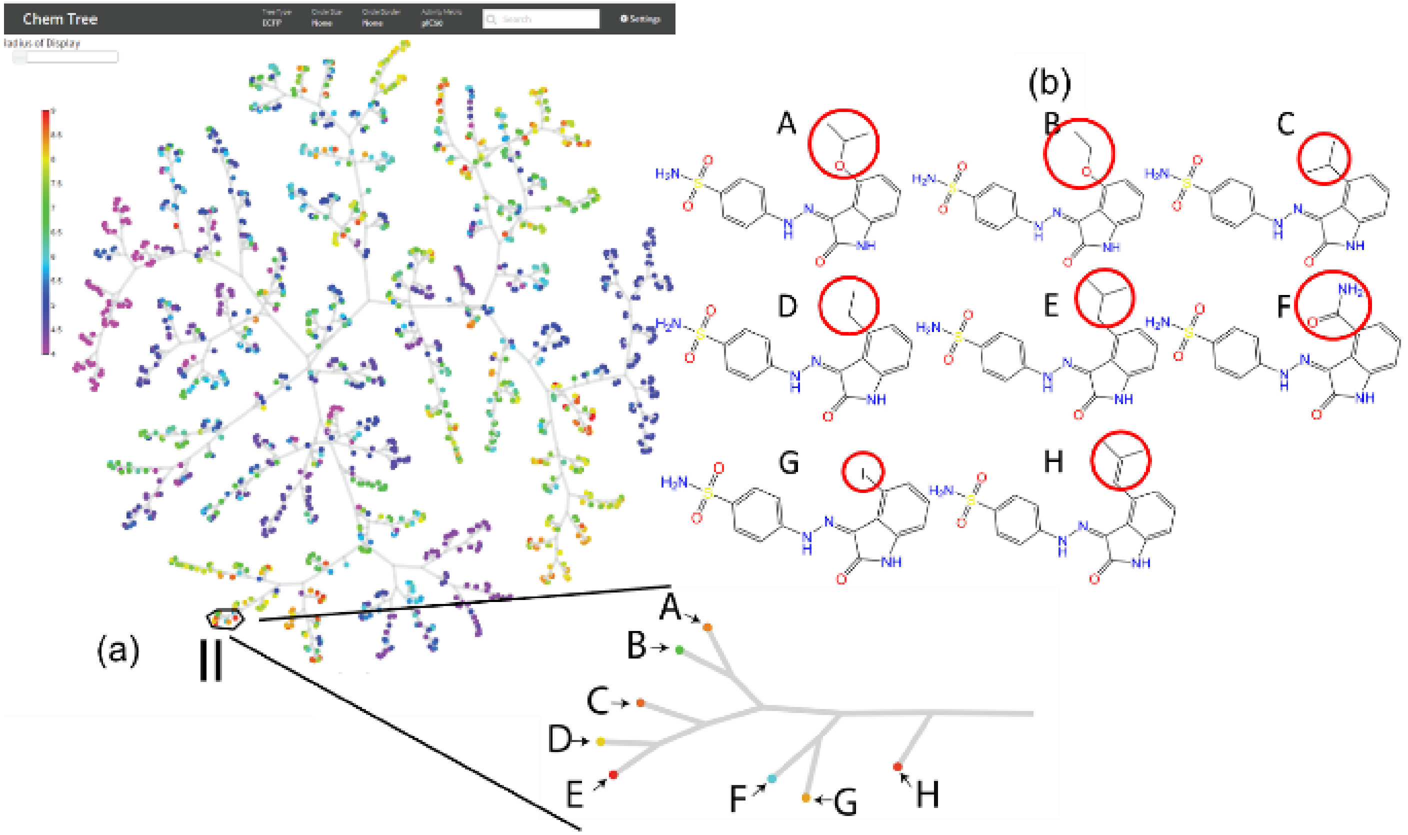


Figure 2-9 (a) ChemTreeMap for the CDK2 dataset. Sub-tree II contains molecules with large variance in bioactivity. (b) Expanded view of sub-tree II. Red circles mark the differences in the chemistry across the similar molecular cores.

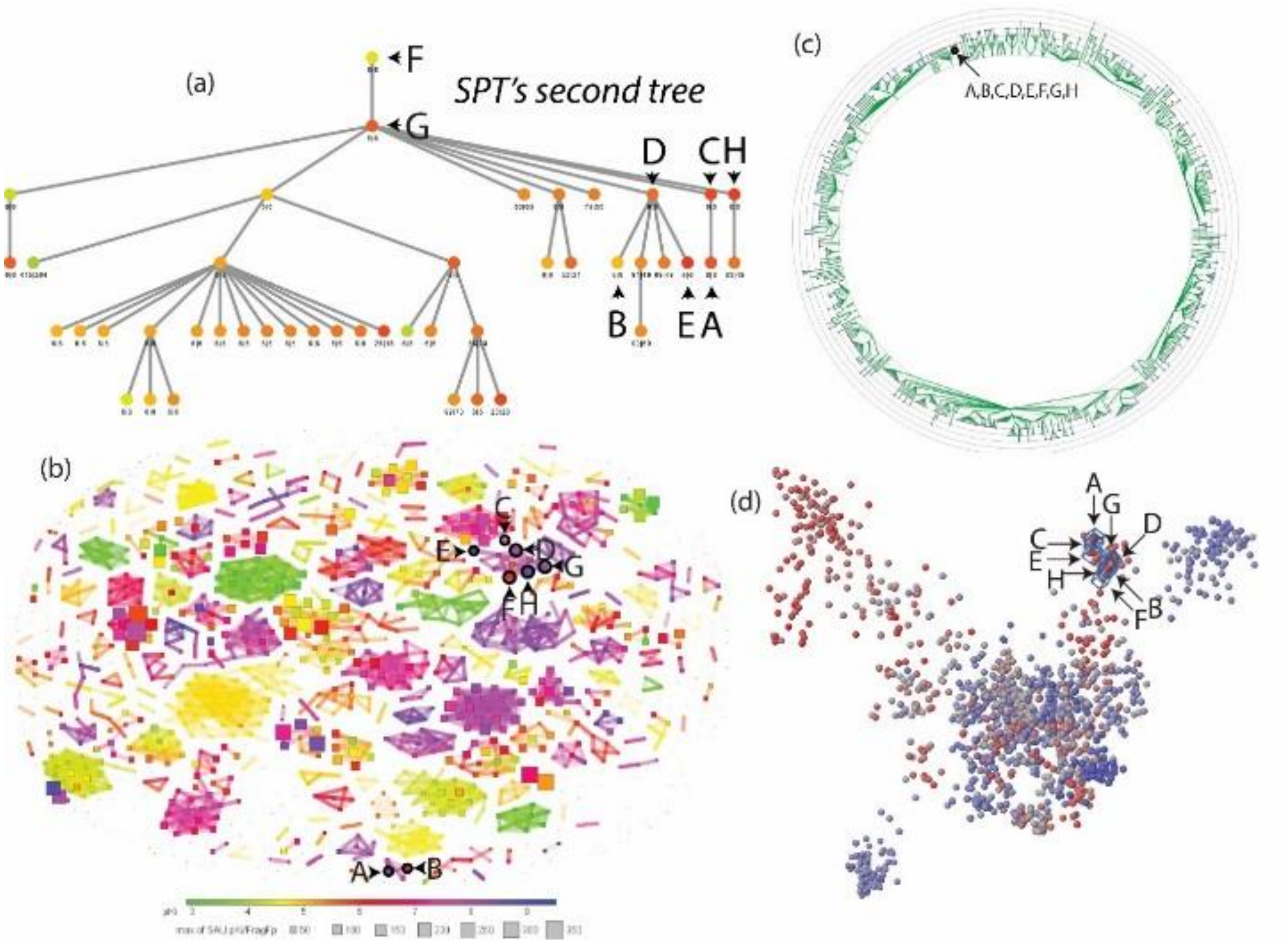


Figure 2-10 The visualization of CDK2 compounds using three other tools: (a) SPT, (b) Data Warrior, and (c) Scaffold Hunter.

In Fig 10(a), SPT identifies sub-tree II compounds in its second-ranked tree. The ranking of the tree is fitting because some of the compounds have low activity. The chemical similarity is clear from the SPT tree, but SPT's color scheme is a limitation. The colors are scaled to span the minimum to maximum values across the whole dataset. ChemTreeMap's color scale makes it easier to identify the drop in activity for compounds B and F. Data warrior in Fig 10(b) captures the similarity of the ethers in A and B, but they are far separated from the rest of the compounds. Also, C-E are not necessarily distinct from F-H. Scaffold Hunter in Fig 10(c) shows that all molecules of sub-tree II share the same scaffold, but the chemical differences between them are not revealed. CheS-Mapper in Fig 10(d) identifies the similarity among A-H, but overlapping nodes make distinguishing the activity difference difficult.

Results for p38α can be found in the Appendix A (pg #148). It should be noted that our analysis is based on global exploration of each dataset, but the discussions have focused on local sub-trees. The Appendix A (pg #148) contains resulting diagrams for each method, using solely the sub-tree compounds. Occasionally, there are slight reorganizations, which are expected. Overall, the results are the same.

2.7 Conclusion

ChemTreeMap is innovative for quantifying chemical similarity in the branch lengths as done for phylogenetic trees, organizing molecules in an alternative hierarchy, and mapping multiple properties to graphical attributes. It uses robust, widely accepted methods.

ChemTreeMap is designed as a general purpose chemical visualization and data mining tool with many interactive features to ease the navigation in large datasets, like dragging, zooming, and searching. Single clicks on a leaf yields detailed molecular information. Its dynamic layout allows users to modify the tree, using a click-and-drag feature on the nodes that can reposition branches to improve the view.

ChemTreeMap is applicable to a wide range of problems. Any data can be mapped onto the similarity tree. The approach does not make any assumptions about the relationship between activity and structure, thus enabling a data-driven interpretation of biochemical information. It is also implemented in a client-server format that allows efficient data sharing between collaborators.

Chapter 3. Physicochemical differences between allosteric and competitive ligands

3.1 Abstract

In many drug design projects, the target is the protein's active site, but in some cases, allosteric sites are more amenable to drug development. Here, we utilize the Allosteric Database (ASD) and ChEMBL to systematically obtain large datasets of both allosteric and competitive ligands. Our original set was created in 2012 and was composed of 8827 unique allosteric ligands and 3194 unique competitive compounds. This was updated in 2015 to contain 70,488 and 11,874 unique ligands for the allosteric and competitive sets, respectively. Physically relevant compound descriptors were computed to examine the differences in chemical properties of these extensive datasets. Particular attention was given to removing redundancy in the data and normalizing across ligand diversity and protein targets. The resulting individual and pair-wise distributions show that allosteric ligands are more hydrophobic, aromatic, and rigid. These results are robust across different normalization schemes, and they are found in both the 2012 and 2015 data. Furthermore, the ligands that are highly similar

between the two sets (compounds with the potential to have both allosteric and competitive mechanisms in different targets) fall squarely in allosteric chemical space. These insights may aid future molecule design to create compound libraries with specific biases for allosteric or competitive modes of action.

3.2 Introduction

The major goal in drug discovery is to design a small molecule that binds specifically to a particular protein target and achieves a desired physiological effect. Typically, the binding site for these small molecules is the protein's functional active site. However, a large number of active sites have physicochemical properties which are hard to target with a drug-like small molecule[200]–[203]. However, these proteins may also have secondary, allosteric sites that have the ability to modulate function by inducing conformational or dynamic changes. Typically, allosteric sites have no steric overlap with the active site. It is hypothesized that these binding sites have different physical and chemical properties which may be amenable to small molecule design when the active site has been found to be difficult to target and potentially "undruggable".[204]

Many examples of allosterically modulated proteins have been annotated and thoroughly studied in the literature since the formalization of the theory by Monod, Wyman, and Changeux in 1965[204], [205]. Until recently, most studies have focused on characterization of allosteric ligands to a single protein[206]–[209]. Studies of allosteric ligands have ranged from the control of metabolic mechanisms to signal-transduction pathways[210]. Large databases such as PubChem[211], DrugBank[212], and ChEMBL[213] have allowed researchers to mine interesting patterns to help predict protein-ligand interactions. In particular, ChEMBL is annotated with descriptions of the included assays, which often includes the type of interaction, including allostery[213]. An additional allosteric-specific database, Allosteric Database (ASD) has been created with >100,000 allosteric ligands for mining[214], [215]. This study utilizes both ChEMBL and ASD to mine patterns that discriminate allosteric from competitive ligands. Many studies have explored allosteric mechanisms, but they tend to focus on a single protein's mechanism[216]–[219] or investigate the issue from the perspective of the protein[220]–[223].

The goal of this study is to understand the ligand's role in allostery and what makes them unique. Two other studies have mined for generic properties of allosteric ligands. Wang *et al.* compared the properties of the ligands contained in ASD to several databases of known biological active compounds[224]. They showed that ligands in ASD contain more hydrophobic scaffolds and have a higher structural rigidity than the molecules in other databases, including Accelrys Available Chemicals Directory (ACD) [225], Accelrys Comprehensive Medicinal Chemistry (CMC) [226], Chinese Natural Product Database (CNPD) [227], DrugBank [2]–[4], MDDR [228], and NCI Open Database [229]. In this study, although ASD contains only allosteric compounds, there was no guarantee that the other databases do not contain allosteric ligands. In the second study, Van Westen *et al.* compared allosteric versus non-allosteric compounds in ChEMBL[230]. Their main focus was on a few classes of protein targets and the development of predictive models for: Class B GPCRs, HIV reverse transcriptase, Adenosine receptors, and Kinase modulators. In their analysis of allosteric versus non-allosteric compounds, two important observations were made: firstly, allosteric ligands are not distinct from and tend to be a subset of non-allosteric ligands; and secondly, allosteric ligands are more drug-like than non-allosteric ligands.[230]

In this study, we focus on specifically differentiating allosteric and competitive ligands, and their impact upon protein-substrate binding or protein activity by binding to allosteric or active sites, respectively. This answers a different question than that answered by Wang *et al.* or van Westen *et al* described above. In those studies, they compared allosteric ligands to all other biologically active ligands or non-allosteric ligands. These definitions can avoid any falsely identified non-allosteric complexes. This study uses both ASD[214], [215] and ChEMBL[213] to have better coverage of possible allosteric compounds than any previous study. An issue that has not been addressed before is normalizing the data to correct for biases that come from some systems having much more data than others. To address this, clustering was performed on two levels to reduce the redundancy of protein-ligand complexes and yet maintain the diversity across the sets. First, targets are grouped by sequence similarity, and then ligands are clustered within each protein family. Since each of these databases is also annotated with the protein target, we examined the distributions of protein targets from each dataset.

3.3 Methods:

3.3.1 Data Collection

The study utilizes three datasets: a 2012 dataset, a 2013-2015 dataset, and a full dataset (2015). Previous studies only used one dataset, a single snapshot of the compound databases before a certain date. However, with exponentially growing data, the results could be biased by newly studied compounds. Multiple datasets from multiple time points should reduce this potential problem, when they are normalized for redundancy. The approach in this study is relatively conservative. Significant descriptors are found with 2012 dataset and confirmed in the full dataset. Seeing consistent patterns across the different datasets and different normalization schemes show that our results are robust.

The information on molecular structures and target proteins in the 2012 dataset were collected for allosteric and competitive binding from ASD version 1.0 and ChEMBL version 11. ASD is a database with a detailed description of allostery, biological processes, related diseases, and binding affinities. ChEMBL is a large library of drug-like bioactive compounds, containing binding, functional, and ADMET (Absorption, Distribution, Metabolism, Excretion, and Toxicity) information. It covers >500,000 assays mapped to 8200 targets, including 2388 human proteins (Release 11). The full dataset was collected from ASD version 3.0 and ChEMBL version 20. The 2013-2015 dataset was the resulting data after taking 2012 dataset out of the full dataset obtained in 2015. Filtering was done on all datasets as described below.

The allosteric data was extracted from both ChEMBL and ASD. ASD data is imported without filtering since it focuses exclusively on allosteric mechanisms. Data from ChEMBL was filtered in order to select appropriate allosteric ligands to augment the ASD dataset. The entire set of ChEMBL assay descriptions was filtered for those which contain the term "alloster*". All ligands which were characterized by high throughput screening (HTS) assays were then removed because of the high error rate in HTS approaches. Allosteric-relevant assays from the remaining set were then selected by manually reading the descriptions. Of those selected, only active molecules were kept. The dataset for competitive compounds, which is based only on ChEMBL, was obtained by searching for the term "compet*". Then, the dataset is filtered by the same procedure as the allosteric set, including by-hand verification of assay descriptions.

The 2012 allosteric set produced by this process has 7873 unique ligands from ASD and 954 from ChEMBL, that together make 8827 unique allosteric ligands that target 314 unique proteins. The 2012 competitive set contains 3194 unique ligands from ChEMBL that target 338 unique proteins. For the full 2015 dataset, the allosteric set is composed of 68,612 unique ligands from ASD and 3,390 from ChEMBL, totaling 70,488 unique ligands targeting 1,048 unique proteins. The full competitive set has 11,874 unique ligands from ChEMBL, targeting 1,004 unique proteins. All these datasets (Table 1) are used to do the calculations described in the rest of the methods section.

Table 3-1 The number of protein families and their subsequent protein-ligand clusters at varying cutoffs for sequence (% Identity) and chemical (Tc) similarity.

# Unique Ligands	**#Protein Families (Providing #Protein-Ligand Clusters)**			
	100%/1	90%/0.9	75%/0.75	60%/0.6
Original Set (2012)				
8827 Allosteric Ligands	314 (9917)	294 (9353)	283 (6357)	269 (3114)
3194 Competitive Ligands	338 (5649)	287 (5104)	258 (3473)	220 (1809)
Full Set (2015)				
70,488 Allosteric Ligands	1048 (145,056)	924 (96,318)	859 (55,007)	760 (24,939)
11,874 Competitive Ligands	1004 (17,551)	896 (16,491)	784 (11,387)	679 (6474)

3.3.2 Calculation of Compound Descriptors

Molecular Operating Environment (MOE) 2014[193] was used to calculate the compound descriptors. The SMILES were converted to molecular structures in MOE and ligands were properly protonated using the default options of the "Wash" procedure in MOE. While we expect some small error rate, there is no reason for the rate to differ across the various subsets of molecules.

Descriptors were calculated to characterize molecules in various aspects, including atom counts, bond counts, physical properties (SlogP, FCharge), and drug/lead-like characterizations. Some descriptors are highly correlated with ligand size (i.e. the number of carbons is highly correlated to the number of heavy atoms), so the normalizations of these descriptors by size were also calculated (e.g. chiral/a_heavy, a_nC/a_heavy). All descriptors available in MOE were computed, but emphasis is placed only on the descriptors that are experimentally measurable and can be predictively modified. The list of the descriptors is in Table 2.

Table 3-2 The list of physicochemical properties that were compared.

Category	Code	Description	Additional Code*
Atom	a_heavy	Number of heavy atoms.	
	a_acc	Number of hydrogen bond acceptor atoms.	a_acc/HA
	a_acid	Number of acidic atoms.	a_acid/HA
	a_aro	Number of aromatic atoms.	a_aro/HA
	a_base	Number of basic atoms.	a_base/HA
	a_don	Number of hydrogen bond donor atoms.	a_don/HA
	a_nC	Number of carbon atoms.	a_nC/HA
	chiral	The number of chiral centers.	chiral/HA
	rings	The number of rings.	
Physical Properties	FCharge	Total charge of the molecule.	FCharge/HA
	SlogP	Log of the octanol/water partition coefficient.	
	logS	Log of the aqueous solubility (mol/L).	
Drug/Lead-like	lip_druglike	One if and only if lip_violation < 2 otherwise zero.	
	lip_violation	The number of violations of Lipinski's Rule of Five.	
	opr_leadlike	One if and only if opr_violation < 2 otherwise zero.	
	opr_violation	The number of violations of Oprea's lead-like test.	
Bond	b_1rotN	Number of rotatable single bonds.	b_1rotN/HA
	b_ar	Number of aromatic bonds.	b_ar/HA
	b_count	Number of bonds.	b_count/HA
	b_rotN	Number of rotatable bonds.	b_rotN/HA
	b_single	Number of single bonds.	b_single/HA
	b_single/b_count	(Number of single bonds) / (Number of bonds).	
	b_1rotN/b_count	(Number of rotatable single bonds) / (Number of bonds).	
	b_ar/b_count	(Number of aromatic bonds) / (Number of bonds).	

* Descriptors corrected by heavy atom, which corrects for the correlation between molecule size and number of atom and bond types.

3.3.3 Removing Redundancy

Redundancy has not been appropriately addressed in previous studies and may potentially lead to bias from overrepresented data. Previous studies have only clustered molecules[231] or limited their analysis to one protein family[232]. Properties of allosteric compounds are highly dependent on the function of the protein and the effects that compounds have. Different protein functions are involved with distinctive biological process, which may require allosteric molecules to have specific physicochemical properties to perform the interaction. Therefore, clustering for both proteins and ligands is necessary to have a dataset that evenly represents the various interactions.

This study adopted a two-level clustering method. The target proteins were clustered by sequence identity, which was calculated by BLAST[233] (formatdb and blastp) (x-axis of Figure 3-1), by running formatdb and blastp on the target protein sequence fasta file with default parameters. Four different thresholds for BLAST identity were used (60%, 75%, 90%, and 100%) to cluster sequences into protein families. Then, for each protein family, ligands are clustered by Pipeline Pilot 9.2[234] (y-axis in Figure 3-1), with the Cluster Molecules component at the maximum dissimilarity[187] setting. The ligand similarity is quantified by the Tanimoto coefficient (T_c) of the ECFP6 fingerprint[56]. Four different thresholds for similarity were used to cluster the ligands (T_c = 0.6 for protein families at 60% sequence identity, 0.75 for 75% sequence identity, 0.9 for 90% sequence identity, and 1 for 100% sequence identity). This process allowed us to examine the data in fine detail and across broad levels. At this point, the most common procedure is to choose the “center” of each protein-ligand cluster to represent those molecules in the data analysis. The center is defined as the molecule with smallest sum of T_c distance to other molecules in the cluster. This is the top scheme shown in Figure 3-1, and that analysis is provided in Appendix D as Table D-2 and Table D-3.

3.3.4 The Weighting Procedure, a Better Method for Removing Redundancy while Maintaining Diversity

The traditional clustering chooses only one ligand (the center) for each cluster. However, this loses the information from the other ligands in the cluster. We used a weighted procedure where every molecule is included, but its contribution to the analysis is down-weighted by the number of molecules in its cluster (also shown in Figure 3-1). In this approach, summing the weighted data in the cluster results in the full distribution of the properties of the cluster weighted as “one molecule.” Clustering in this fashion also facilitates bootstrap sampling to determine the errors and variability in the calculated physical properties. Here, 100,000 samples (on the order of the number of ligand clusters in each set) were generated by selecting from each bin with probabilities based on the weighted distributions. The bin widths for the weighted histograms are 1 for descriptors with discrete values and 0.02 for continuous variables. The distributions of the mean and median from each sample are computed, and the 95% confidence interval (95%ci) of the mean and median are then defined as the range of 2.5% to 97.5% of those distributions. Statistically significant differences required no overlap in the 95%ci of the medians.

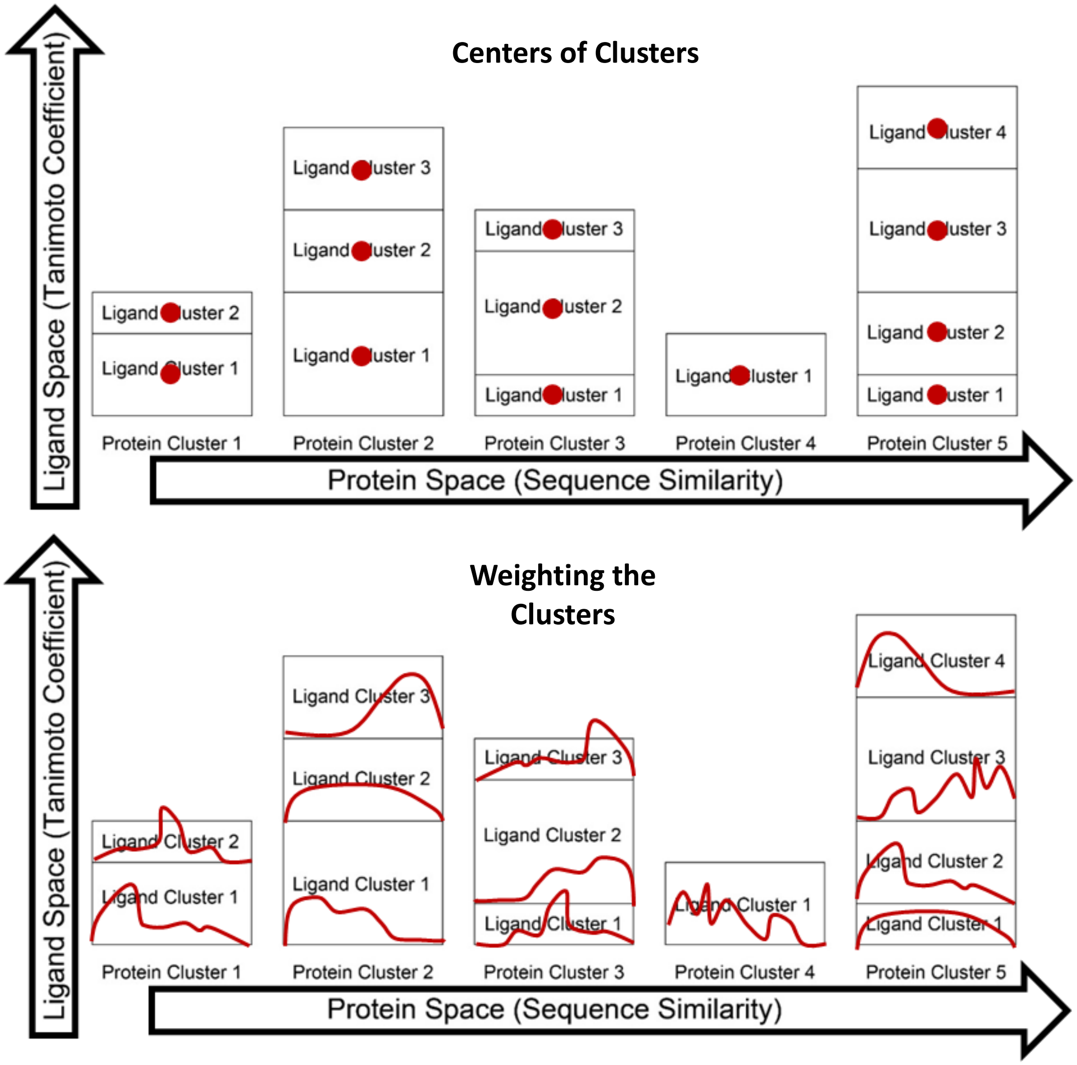


Figure 3-1 Clustering the data in two levels that includes both protein and ligand diversity. Horizontal axis represents clustering in protein space, which is done first. Vertical axis represents clustering the ligands found in all members of the protein cluster. The top scheme shows one approach for normalization where each protein-ligand cluster is represented by the ligand in the "center" as described in the text. The lower scheme shows our preferred approach where we maintain the full details of all molecules in the protein-ligand cluster, but we scale them by the number of ligands so that each protein-ligand cluster has the same contribution (i.e. counts as one molecule just like the top example).

3.3.5 Overlap between allosteric and competitive dataset

ChemTreeMap[235] is used to view the chemical space of the structures and determine differences in the two sets. ChemTreeMap can organize molecules in a hierarchical tree structure to convey molecular similarity information by combining extended connectivity fingerprint and a neighbor-joining algorithm. With hierarchical organization and color coding, ChemTreeMap is able to highlight the regions where chemical space is unique to one group of compounds. ECFP6 is used to characterize the molecules, and the hierarchical structure is determined from the T_c between molecules.

3.3.6 Chemical composition of datasets

Ring structures and chain assemblies are essential to the molecule scaffold and can significantly contribute to the physicochemical properties of the molecule. To investigate the difference between the allosteric and competitive sets, the count of the most frequent ring and chain assemblies are analyzed using Pipeline Pilot. All substructures and their frequency were obtained using the Most Frequent Fragments component in Pipeline Pilot, by setting the NumberToKeep variable to an extremely high value (the number of returned fragments is less than that value). The default maximum fragment size was kept at 25.

3.3.7 Data Analysis

3.3.7.1 Removing seven overrepresented molecules.

There are seven compounds in the competitive set that are overrepresented because of a study, where each compound was tested against a panel of >200 kinases. ChEMBL has all included data and marked them as active[236]. Figure 3-2 shows the influence that these seven “standards” molecules have on the set of thousands of competitive ligands; clearly, they had to be removed because they reflect a man-made artifact.

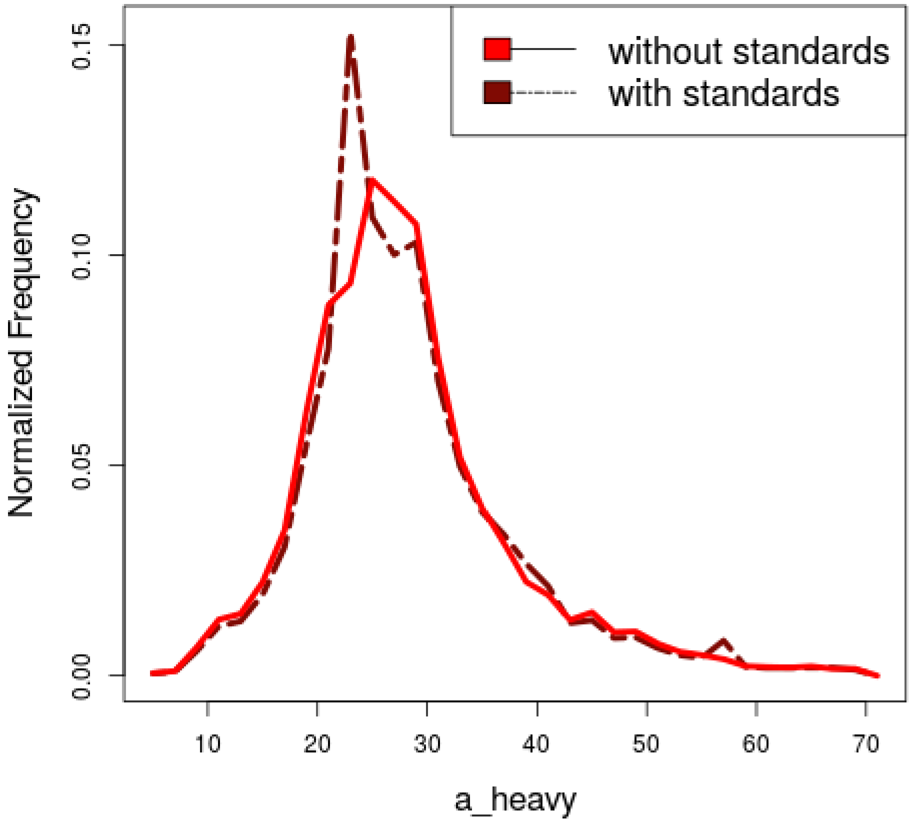


Figure 3-2 When the seven kinase standards are included in the full 2015 dataset of 11,874 competitive ligands, there are obvious artifacts in the distributions of physicochemical properties. As an example, the distribution of heavy atoms is shown with (dark, dashed line) and without (red, solid line) the seven compounds included. We chose to exclude these compounds.

3.3.7.2 Identifying physicochemical properties that discriminate between allosteric and competitive ligands.

In addition to requiring no overlap in the 95%ci of the physicochemical properties, the differences between the allosteric and competitive distributions were assessed by two methods: the Wilcoxon test[237] for data from the centers of the clusters and the weighted Wilcoxon test for our weighted distributions. Many descriptors have non-Gaussian distributions; therefore, the non-parametric Wilcoxon test was the appropriate choice. The weighted Wilcoxon test is calculated with all compounds in the cluster based on the weights determined in the weighting procedure described above. We used a strict threshold of p-value < 0.0001 to define a statistically significant difference between allosteric and competitive distributions; this was to reduce the likelihood that the differences were a coincidental artifact of having two extremely large datasets and 37 comparisons of physical properties. These tests were implemented through R-Statistics (version 3.2.2), the Wilcoxon Test in stats package[238] is used for Wilcoxon test. Functions svydesign and svyranktest in survey package are used for the weighted Wilcoxon test.

3.3.7.3 Analysis of pair-wise distributions.

Pairs of descriptors were mined to find simple and intuitive rules to differentiate them. For example, the SlogP may be higher because there is A) a decrease in the number of hydrogen-bond donors/acceptors or B) the addition of hydrophobic groups, and the two cases could be differentiated using two-dimensional histograms of A) SlogP and a_acc+a_don vs B) SlogP and a_nC. The significance of a descriptor pair is then quantified by 2D Kolmogorov–Smirnov (K-S) test. The 2D K-S test is a nonparametric test of significance with no assumption about the descriptor distribution. The results generated by this method are sensitive to differences in both location and shape of the empirical cumulative distribution of the two samples; therefore, the distance can capture multiple aspects of the distributional differences.

3.4 Result and Discussion:

We clustered the data at several levels to ensure that we identified robust trends that were valid for the data in fine detail and over broad categories. The clustering was done at four levels of BLAST and chemical similarity: sequence identity/T_c = 100%/≥1.0, 90%/≥0.9, 75%/≥0.75, or 60%/≥0.6. Trends in the physical properties had to be statistically significant at all four levels of clustering, and those that were observed in both the full dataset and the original 2012 dataset were considered more universal. Below, we show that allosteric ligands are more hydrophobic, aromatic, and rigid. This is in agreement with previous studies.[224],[230]

Furthermore, we examined the compounds with the highest similarity between the allosteric and competitive sets. These ligands should have the most potential to exhibit both allosteric and competitive mechanisms of action in different protein assays. We show that this “overlap chemistry” falls squarely in allosteric chemical space.

3.4.1 Distributions of Protein targets

Due to the different modes of action, one might expect to see a difference in the types of proteins targeted by allosteric vs competitive ligands as this depends upon protein function (Figure 3-3).

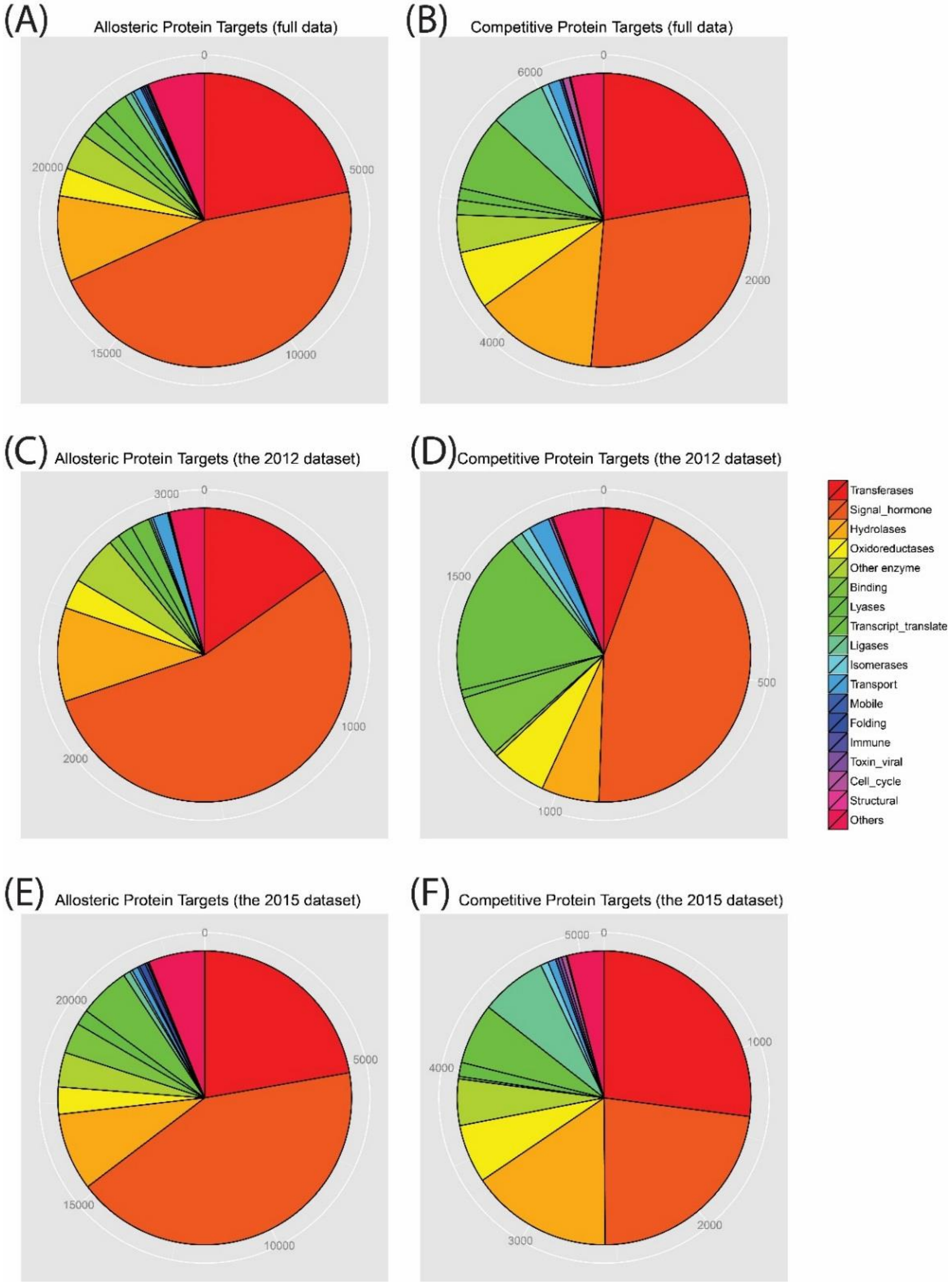


Figure 3-3 The distribution of protein targets in the protein-ligand clusters of allosteric and competitive compounds. The first row (A, B) is for the full 2015 dataset, second row (C, D) is for the original 2012 dataset, and the third row (E, F) is for 2013-2015 dataset. The protein categories are taken from Enzyme Classification numbers and keywords in the assay descriptions.

When clustering the 2015 dataset at 60%, the number of unique proteins in the full allosteric dataset is 760, while there are 679 different proteins in the competitive dataset (2012 dataset has 269 allosteric and 220 competitive protein targets). ASD 3.0 has increased by >400% since the initial ASD 1.0, due to the significant expansion of allosteric drug discovery[26], [239], [240]. Three categories of allosteric proteins are dramatically augmented from ASD 1.0 to ASD 3.0: kinases (from 46 to 207), GPCRs (from 48 to 118), and ion channels (from 21 to 134)[26], which are highly associated with the therapeutic targets in recent drug discovery studies[3], [241]. As one would expect, there are differences in the protein targets of allosteric and competitive ligands. A large portion (46.3%) of allosteric ligands target GPCR proteins, while 29.1% of competitive compounds bind to these proteins. A larger percentage of allosteric compounds (14.4%) target neuronal proteins (for example, the GABA receptor, glutamate receptor, and acetylcholine receptor) compared to competitive compounds (8.1%). Li *et al.* found that the majority of allosteric proteins obtained from ASD were transferases (44.8%) instead of GPCR; however, they did not have a designation for non-enzymatic proteins, so this percentage is artificially high.[242] In our full dataset, 21.9% of allosteric ligands and 13.2% of competitive ligands target transferases. For enzymes in the datasets, 53.3% of the targets in the allosteric set and 26.7% of the targets of the competitive set are transferase. A complete set of the protein targets in each set is given in Appendix D (pg #240)

A large number of GPCR-based allosteric compounds were developed[26] over 2013-2015. GPCRs play a critical role in multiple diseases, which provides an attraction for large efforts in developing new GPCR-based drugs[243]. Moreover, many subtype GPCRs have high sequence similarity in the orthosteric site. Targeting those sites has been difficult to obtain high selectivity. In recent years, targeting allosteric sites is a major thrust for developing GPCR drugs[244], [245].

Van Westen *et al.* built a dataset from ChEMBL (417 targets for allosteric ligands and 1,869 for non-allosteric ligands) and examined the distribution of targets and found a bias to transmembrane proteins (~50%). That study did not remove redundancy on the protein level or the ligand level. Our study reduced this bias caused by overrepresented proteins and ligands utilizing two-level clustering for both protein and ligands to remove redundancy.

3.4.2 Protein-Ligand Clusters

It is worth noting that the full dataset in 2015 is >500% larger than the 2012 dataset. The number of protein-ligand clusters is different for each protein and is dictated by the chemical diversity of its ligands. In our full dataset, there are 70,488 allosteric ligands and 11,874 competitive ligands. These are spread across 1048 and 1004 protein targets, respectively. Some compounds can interact with more than one target, and the number of protein-ligand clusters ranges from 679 clusters for the competitive compounds at

60/0.6 clustering up to 145,056 for allosteric ligands clustered at 100/1.0 (Table 3-1). The clusters are used to reduce the bias from heavily studied proteins vs newer targets.

The sizes of the ligand datasets derived from ChEMBL are smaller than those used by van Westen *et al.* because the datasets were designed with compounds that are expressly either allosteric or competitive. Therefore, concrete evidence must exist based on restrictive search terms on the assay descriptions as opposed to parsing the language of the ChEMBL documents. The allosteric set from ChEMBL is smaller (3,553 vs. 17,829) due to the fact only "alloster*" was used in the keyword search of the *assay description* and a manual curation was performed to remove HTS data. However, van Westen used several additional terms, which may imply allostery, when searching the whole *document* from ChEMBL. Discarding the HTS data also limits the size of the dataset used in this study. Using the assay description and the more restrictive search term helps to ensure that each ligand obtained from ChEMBL is indeed an allosteric modulator. The "competitive" dataset obtained from ChEMBL is also smaller than the "non-allosteric" set, since the focus is on competitive inhibitors annotated in the assay descriptions, and thus natural ligands for the protein active site are not included in this set. Any bias the natural ligand has on the properties of the competitive dataset is removed by searching for only "competitive" ligands in ChEMBL. The growth of allosteric ligands from the original 2012 set to the full set in 2015 brings the dataset to a size comparable to van Westen's study.

3.4.3 Chemical Space of Allosteric and Competitive Ligands

The allosteric and competitive ligands both span similar chemical space (see Figure 3-4). Van Westen *et al.* suggested that allosteric modulators form a subset of non-allosteric modulators, based on the observation that their allosteric ligands had a narrower range of molecular weight and covered a smaller area in a scatter plot of logP vs molecular weight. However, when we compare our allosteric and competitive sets based on ligand similarity (T_c), the compounds of both categories appear to cover similar chemical space.

We were particularly interested in the ligands that were the most similar (Tc≥0.9) between the allosteric and competitive sets. These molecules should exhibit both allosteric and competitive mechanisms of action depending on different protein targets. It turns out that the "overlap chemical space" is significant. For the full dataset, 2589 allosteric ligands (3.7%) are within Tc ≥ 0.9 of a competitive ligand, and 2494 competitive ligands (21.0%) are within Tc ≥ 0.9 of an allosteric compound. Combined together, there are 2631 "overlap" molecules. In the last section, we compare these compounds to the differences between allosteric and competitive compounds.

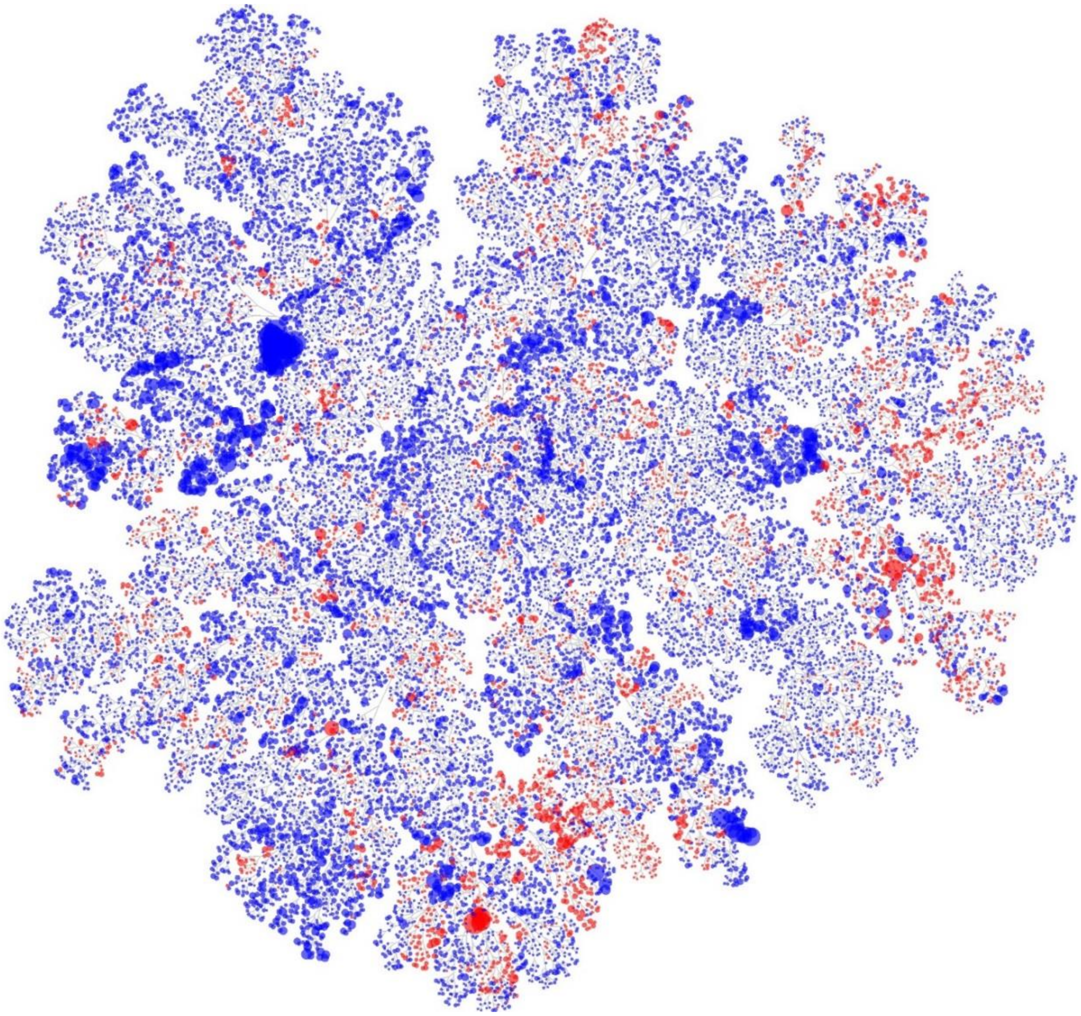


Figure 3-4 This ChemTreeMap[235] is a hierarchical tree based on grouping ligands by Tc. There is good overlap in chemical space for allosteric (blue) and competitive (red) ligands in the 2015 set. The size of each circle represents the number of ligands in a cluster of Tc ⩾0.6. Though a few small regions of chemical space are dominated by one set or the other, the large number of branches with interdigitated red and blue circles shows that there is a great deal of chemical similarity between the allosteric and competitive ligands. To quantify the overlap, we should note that 5226 out of 70,488 allosteric ligands (7.4%) are within Tc ⩾0.6 of a competitive ligand, and 3113 of the 11,874 competitive ligands (26.2%) are within Tc ⩾0.6 of an allosteric compound.

3.4.4 Physicochemical Differences between Allosteric and Competitive Ligands

The medians of each physicochemical property (and their 95%ci) are given in Table 3-3 and Table 3-4. The values that are listed with bold font have statistically significant differences between the allosteric and competitive sets. This is determined by both the weighted Wilcoxon test (p-value <0.0001) and no overlap in the 95%ci. The physical property label is in bold font when the same statistically significant trend is seen for all levels of protein-ligand clustering. Properties listed in red font are only significant for the 2012 or 2015 dataset, but not both. For instance, the original dataset showed that competitive ligands had more chiral centers, but that was no longer observed when the data was updated. Therefore, we do not consider the trend to be robust.

Table 3-3 Medians (95%ci) of the 37 physicochemical properties for the original 2012 dataset. Numbers in bold are statistically significant differences between allosteric and competitive compounds. The list is ordered by largest differences. Red properties do not repeat in 2015 data.

Clustering level	60%/0.6		75%/0.75		90%/0.9		100%/1.0	
Descriptors	**Allosteric**	**Competitive**	**Allosteric**	**Competitive**	**Allosteric**	**Competitive**	**Allosteric**	**Competitive**
chiral	**0(±0)**	**1(±0)**	**0(±0)**	**1(±0)**	**0(±0)**	**1(±0)**	**0(±0)**	**1(±0)**
chiral/HA	**0(±0)**	**0.049(±0.001)**	**0(±0)**	**0.04(±0.00)**	**0(±0)**	**0.04(±0)**	**0(±0)**	**0.043(±0.007)**
SlogP	**3.3(±0.2)**	**1.9(±0.2)**	**3.58(±0.06)**	**2.1(±0.2)**	**3.70(±0.06)**	**2.2(±0.1)**	**3.69(±0.06)**	**2.1(±0.2)**
b_single/HA	**1.18(±0.01)**	**1.52(±0.02)**	**1.17(±0.01)**	**1.50(±0.02)**	**1.17(±0.01)**	**1.53(±0.02)**	**1.18(±0.01)**	**1.53(±0.01)**
b_ar/b_count	**0.29(±0.01)**	**0.21(±0.01)**	**0.309(±0.001)**	**0.222(±0.002)**	**0.309(±0.001)**	**0.214(±0.006)**	**0.309(±0.001)**	**0.214(±0.006)**
b_single	**29(±1)**	**36(±2)**	**30(±1)**	**38(±1)**	**32(±0)**	**41(±1)**	**32(±1)**	**41(±1)**
b_ar/HA	**0.52(±0.02)**	**0.43(±0.02)**	**0.545(±0.005)**	**0.44(±0.01)**	**0.545(±0.005)**	**0.429(±0.001)**	**0.548(±0.002)**	**0.429(±0.001)**
a_aro/HA	**0.50(±0.02)**	**0.41(±0.02)**	**0.54(±0.01)**	**0.43(±0.01)**	**0.55(±0.02)**	**0.43(±0.01)**	**0.55(±0.01)**	**0.42(±0.01)**
b_1rotN/HA	**0.17(±0.01)**	**0.20(±0.01)**	**0.172(±0.002)**	**0.194(±0.006)**	**0.172(±0.002)**	**0.20(±0.01)**	**0.172(±0.002)**	**0.211(±0.001)**
b_single/b_count	**0.667(±0.003)**	**0.763(±0.007)**	**0.66(±0.01)**	**0.76(±0.01)**	**0.655(±0.005)**	**0.763(±0.003)**	**0.66(±0.01)**	**0.76(±0.01)**
logS	**-4.7(±0.1)**	**-4.2(±0.1)**	**-5.01(±0.07)**	**-4.42(±0.05)**	**-5.26(±0.08)**	**-4.61(±0.06)**	**-5.32(±0.06)**	**-4.7(±0.1)**
a_acc/HA	**0.107(±0.003)**	**0.097(±0.003)**	**0.107(±0.003)**	**0.091(±0.009)**	**0.107(±0.003)**	**0.095(±0.005)**	**0.107(±0.003)**	**0.097(±0.003)**
b_count/HA	**1.80(±0.02)**	**1.97(±0.03)**	**1.80(±0.01)**	**1.97(±0.03)**	**1.81(±0.01)**	**2.00(±0.03)**	**1.81(±0.01)**	**1.98(±0.02)**
b_count	**44(±1)**	**48(±1)**	**46(±1)**	**50(±1)**	**49(±1)**	**54(±1)**	**49(±0)**	**55(±1)**
a_nC/HA	**0.74(±0.01)**	**0.76(±0.01)**	**0.74(±0.01)**	**0.77(±0.01)**	**0.75(±0.00)**	**0.767(±0.007)**	0.75(±0.00)	0.756(±0.006)
a_base	0(±0)	1(±1)	**0(±0)**	**1(±0)**	**0(±0)**	**1(±0)**	**0(±0)**	**1(±0)**
a_base/HA	0(±0)	0.02(±0.02)	**0(±0)**	**0.024(±0.006)**	**0(±0)**	**0.02(±0.01)**	**0(±0)**	**0.025(±0.005)**
a_acc	3(±1)	2(±0)	**3(±0)**	**2(±0)**	3(±0)	3(±0)	3(±0)	3(±0)
a_don/HA	0.037(±0.003)	0.041(±0.001)	0.03(±0.01)	0.03(±0.01)	0.034(±0.004)	0.034(±0.006)	0.03(±0.01)	0.03(±0.01)
a_aro	12(±0)	11(±1)	12(±0)	12(±0)	**14(±1)**	**12(±0)**	**15(±1)**	**12(±0)**
b_ar	12(±0)	11(±1)	12(±0)	12(±0)	**15(±1)**	**12(±0)**	**16(±1)**	**12(±0)**
a_nC	18(±0)	19(±1)	19(±0)	20(±1)	**20(±0)**	**21(±0)**	**20(±0)**	**21(±0)**
b_rotN/HA	0.21(±0.01)	0.21(±0.01)	0.2(±0.0)	0.209(±0.009)	**0.2(±0.0)**	**0.217(±0.007)**	**0.2(±0.0)**	**0.226(±0.006)**
b_1rotN/b_count	0.1(±0.0)	0.103(±0.007)	0.098(±0.002)	0.1(±0.0)	0.098(±0.002)	0.103(±0.003)	**0.097(±0.003)**	**0.106(±0.004)**
b_1rotN	4(±0)	5(±0)	4(±1)	5(±0)	5(±0)	5(±0)	5(±0)	5(±0)
a_acid	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
a_acid/HA	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
a_don	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
a_heavy	24(±0)	24(±1)	26(±0)	26(±0)	27(±0)	27(±1)	27(±0)	28(±1)
b_rotN	5(±0)	5(±0)	5(±0)	5(±0)	6(±1)	5(±1)	6(±1)	6(±1)
FCharge	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
FCharge/HA	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
lip_druglike	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
lip_violation	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
opr_leadlike	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
opr_violation	0(±0)	0(±1)	0(±0)	0(±0)	**0(±0)**	**1(±0)**	**0(±0)**	**1(±0)**
rings	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)

Table 3-4 Medians (95%ci) of the 37 physicochemical properties for the full 2015 dataset. Numbers in bold are statistically significant differences between allosteric and competitive compounds. The list is ordered by largest differences. Red properties are not found in 2012 data.

Clustering level	**60%/0.6**		**75%/0.75**		**90%/0.9**		**100%/1.0**	
Descriptors	**Allosteric**	**Competitive**	**Allosteric**	**Competitive**	**Allosteric**	**Competitive**	**Allosteric**	**Competitive**
a_don/HA	**0.036(±0.004)**	**0.048(±0.002)**	**0.034(±0.004)**	**0.043(±0.003)**	**0.032(±0.002)**	**0.042(±0.002)**	**0.031(±0.001)**	**0.042(±0.002)**
SlogP	**3.20(±0.06)**	**2.67(±0.07)**	**3.40(±0.04)**	**2.88(±0.06)**	**3.535(±0.004)**	**2.946(±0.007)**	**3.56(±0.02)**	**2.90(±0.06)**
b_single	**30(±1)**	**34(±1)**	**31(±1)**	**36(±1)**	**34(±0)**	**39(±1)**	**34(±1)**	**39(±0)**
b_single/HA	**1.22(±0.01)**	**1.38(±0.02)**	**1.20(±0.01)**	**1.38(±0.01)**	**1.20(±0.01)**	**1.40(±0.01)**	**1.212(±0.002)**	**1.42(±0.01)**
b_ar/b_count	**0.28(±0.01)**	**0.25(±0.01)**	**0.293(±0.003)**	**0.254(±0.006)**	**0.29(±0.01)**	**0.24(±0.01)**	**0.293(±0.003)**	**0.239(±0.001)**
b_count	**44(±0)**	**48(±1)**	**47(±0)**	**50(±1)**	**50(±1)**	**53(±1)**	**51(±0)**	**54(±0)**
a_acc/HA	**0.12(±0.00)**	**0.111(±0.001)**	**0.121(±0.001)**	**0.107(±0.003)**	**0.12(±0.00)**	**0.107(±0.003)**	**0.12(±0)**	**0.107(±0.003)**
b_1rotN/HA	**0.167(±0.003)**	**0.179(±0.001)**	**0.167(±0.003)**	**0.179(±0.001)**	**0.172(±0.002)**	**0.188(±0.003)**	**0.174(±0.004)**	**0.189(±0.001)**
b_count/HA	**1.8095(±0.0005)**	**1.92(±0.01)**	**1.8(±0.0)**	**1.926(±0.006)**	**1.806(±0.004)**	**1.933(±0.007)**	**1.81(±0.01)**	**1.94(±0.01)**
b_single/b_count	**0.682(±0.002)**	**0.721(±0.001)**	**0.672(±0.002)**	**0.72(±0.00)**	**0.672(±0.002)**	**0.727(±0.003)**	**0.673(±0.003)**	**0.732(±0.002)**
a_aro/HA	**0.5(±0.0)**	**0.47(±0.01)**	**0.522(±0.002)**	**0.48(±0.01)**	**0.522(±0.002)**	**0.462(±0.008)**	**0.522(±0.002)**	**0.458(±0.008)**
b_rotN/HA	**0.192(±0.002)**	**0.200(±0.005)**	**0.1905(±0.0005)**	**0.2(±0.0)**	**0.2(±0.0)**	**0.2105(±0.0005)**	**0.2(±0)**	**0.214(±0.004)**
b_ar/HA	**0.50(±0.01)**	**0.48(±0.01)**	**0.52(±0.01)**	**0.48(±0.02)**	**0.53(±0.01)**	**0.47(±0.01)**	**0.522(±0.002)**	**0.462(±0.002)**
a_nC/HA	**0.7308(±0.0008)**	**0.75(±0.00)**	**0.731(±0.001)**	**0.75(±0.01)**	**0.7308(±0.0008)**	**0.75(±0.00)**	**0.7308(±0.0008)**	**0.75(±0.00)**
a_nC	**18(±0)**	**19(±0)**	**19(±0)**	**20(±0)**	**20(±0)**	**21(±0)**	20(±1)	21(±0)
b_1rotN/b_count	**0.093(±0.003)**	**0.097(±0.003)**	**0.094(±0.004)**	**0.096(±0.004)**	0.098(±0.002)	0.1(±0.0)	0.098(±0.002)	0.1(±0)
logS	**-4.50(±0.01)**	**-4.63(±0.08)**	-4.806(±0.002)	-4.86(±0.05)	-5.16(±0.05)	-5.12(±0.01)	-5.17(±0.06)	-5.14(±0.07)
b_1rotN	4(±0)	5(±1)	**4(±0)**	**5(±0)**	5(±0)	5(±0)	5(±0)	5(±0)
a_acc	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)
a_acid	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
a_acid/HA	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
a_aro	12(±0)	12(±0)	12(±0)	12(±0)	**15(±0)**	**12(±0)**	**15(±0)**	**12(±0)**
a_base	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
a_base/HA	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
a_don	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
a_heavy	25(±1)	25(±0)	26(±0)	26(±0)	28(±0)	28(±0)	28(±0)	28(±0)
b_ar	12(±0)	12(±0)	12(±0)	12(±0)	**16(±0)**	**12(±0)**	**16(±0)**	**12(±0)**
b_rotN	5(±0)	5(±0)	5(±0)	5(±0)	**5(±0)**	**6(±0)**	6(±0)	6(±0)
chiral	0(±0)	0(±0)	0(±0)	0(±0)	**0(±0)**	**1(±0)**	**0(±0)**	**1(±0)**
chiral/HA	0(±0)	0(±0)	0(±0)	0(±0)	**0(±0)**	**0.026(±0.006)**	**0(±0)**	**0.028(±0.002)**
FCharge	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
FCharge/HA	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
lip_druglike	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
lip_violation	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
opr_leadlike	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
opr_violation	0(±0)	0(±0)	0(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
rings	3(±0)	3(±0)	3(±0)	3(±0)	4(±0)	4(±1)	4(±0)	4(±0)

3.4.4.1 Allosteric ligands are more hydrophobic.

Allosteric ligands have more positive SlogP than do competitive ligands (see Table 3-3, Table 3-4, and Figure 3-5). In keeping with the increased hydrophobicity, the types of atoms found in allosteric molecules are different compared to competitive compounds. Allosteric ligands have a lower number of hydrogen-bond acceptors per heavy atom. Allosteric ligands also have fewer hydroxyl groups (Table 3-5). Furthermore, competitive ligands may be more hydrophilic because they have more pyrrolidine, piperazine and piperidine rings (see Table 3-6).

This is consistent with the study by Li *et al.*[246] where they found allosteric binding sites contain more hydrophobic surface area, hence would bind more hydrophobic ligands. Hydrophobicity is also used as an important characteristic for the prediction of allosteric sites by Huang *et al.*[247] and by Demberdash *et al.* [220]. It is also important in the prediction of ligand-protein interactions using models developed by Li *et al.*[248] The MWC model also states that protein-protein or subunit interfaces are frequent allosteric binding sites[204], and protein-protein interfaces have generally been shown to be more hydrophobic in nature[249]–[251]. A large portion of allosteric compounds were found to contain hydrophobic scaffolds by Wang *et al.,* and that contributed to the overall hydrophobic nature of their allosteric set[224].

The structure properties of allosteric binding sites likely constrain the chemical characteristic of allosteric ligands. Studies for identifying allosteric binding sites show that the solvent-accessible surface area (SASA)[220], the number of hydrogen bonds[220], interaction between residues, local hydrophobic density[246], [252], pocket size[220], [247], and correlated features are important for describing an allosteric binding site. Van Westen *et al* observed that allosteric compounds for GPCRs tend to be more lipophilic, more rigid (higher sp2 C and lower sp3 C), and relatively smaller than non-allosteric ligands from ChEMBL data[230]. Wang *et al.* compared allosteric ligands from ASD with compounds from databases like DrugBank, MDDR, ACD, etc[224]. However, the compounds in these databases can have many mechanisms to bind target proteins, which are not necessarily binding at the active site. Even so, they also showed that the allosteric ligands contain more hydrophobic scaffolds and are more rigid.

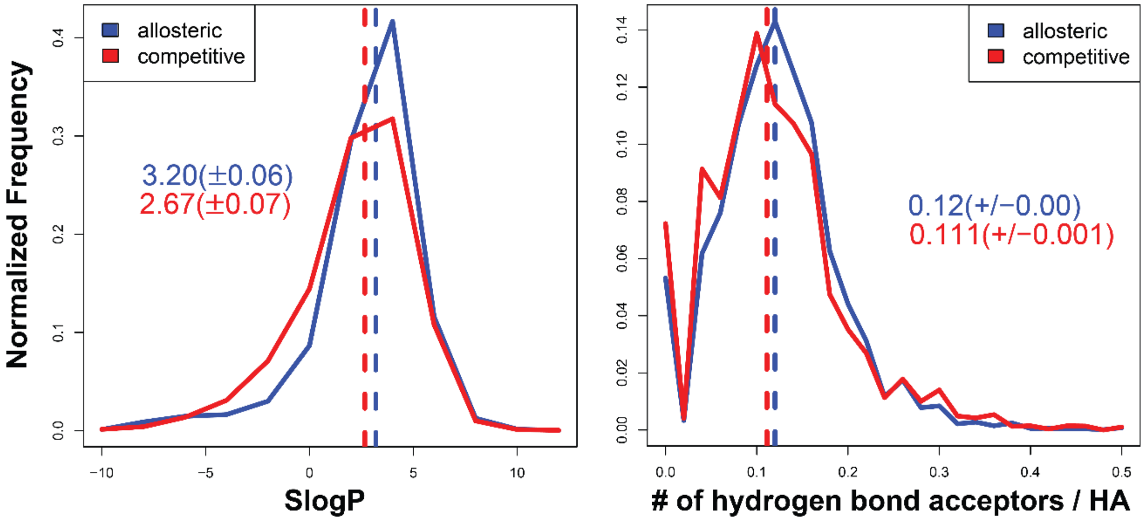


Figure 3-5 The histograms of SlogP and the number of hydrogen-bond acceptors per heavy atom for the full dataset. The median (dashed line) is labeled on the graph with the 95%ci from bootstrap sampling. For brevity, only data clustered at the 60/0.6 level is show

Table 3-5 Chemical analysis with Pipeline Pilot showed that these 10 functional groups have the greatest population differences between allosteric and competitive ligands of the full dataset. The list is organized by the highest absolute difference between normalized frequencies. Bold indicates which set of ligands have more such functional groups. For brevity, only data clustered at the 60/0.6 level is shown.

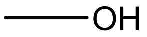
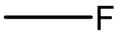
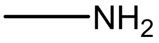
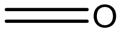
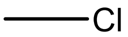
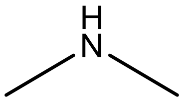
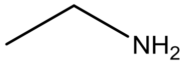
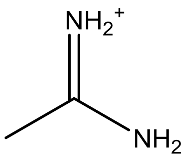
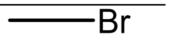
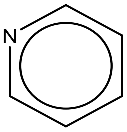
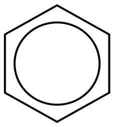
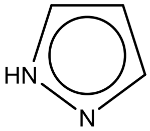
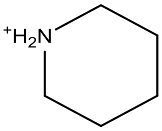
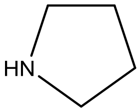
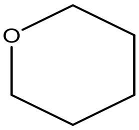
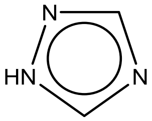
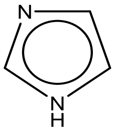
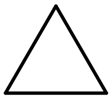
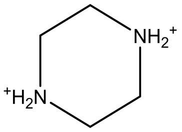
	% Allosteric	% Competitive	Absolute difference
	4.31	**10.95**	6.64
	**6.01**	2.68	3.33
	**9.67**	6.54	3.13
	8.02	**10.34**	2.32
	**6.10**	4.69	1.41
	**1.97**	0.80	1.17
	**1.98**	0.83	1.15
	0.01	**1.14**	1.13
	16.79	15.67	1.12
	**1.71**	0.79	0.92

Table 3-6 Analysis with Pipeline Pilot showed that these 10 rings have the greatest population differences between allosteric and competitive ligands in the full dataset. The list is organized by the highest absolute difference between normalized frequencies. Bold indicates which set of ligands have more such rings. SlogP < 0 are noted in italics. For brevity, only data clustered at the 60/0.6 level is shown.

Ring	% Allosteric	% Competitive	Absolute difference	SlogP
	**7.0**	2.7	4.4	1.08
	37.6	**40.2**	2.6	1.69
	**2.5**	0.8	1.7	0.41
	1.0	**2.3**	1.4	*-0.27*
	0.5	**1.6**	1.1	*-0.66*
	0.7	**1.6**	0.9	1.19
	**0.8**	0.1	0.8	*-0.20*
	**1.4**	0.6	0.7	0.41
	**1.5**	0.7	0.7	1.17
	0.2	**0.9**	0.7	*-2.87*

3.4.4.2 Allosteric ligands are more aromatic and constrained.

In keeping with those same previous studies, we too find that there are multiple descriptors that indicate the allosteric ligands are more rigid. For both the original and the full datasets, the distributions indicate that allosteric ligands have more aromatic atoms and fewer bonds per heavy atom (meaning fewer saturated bonds). There are also fewer rotatable single bonds. The distributions are shown in Figure 3-6. The relation between SlogP and aromaticity can be shown in pair-wise tests and 2D plots (Figure 3-7). The population with ~3 rings in allosteric compounds have clearly higher SlogP, which is shown in both b_ar and a_aro. The increase in density of structures with three rings coincides with a shift in the SlogP, as the black circle in Figure 3-7 A and B shows that the density is shifted to the upper right as is appropriate for the higher populations in the allosteric ligands. The distribution of the number of aromatic atoms (not shown) shows that although allosteric and competitive ligands have the same median (12 atoms, or ~2 ring systems); the allosteric ligands have a much larger peak around 18 atoms or ~ 3 aromatic rings.

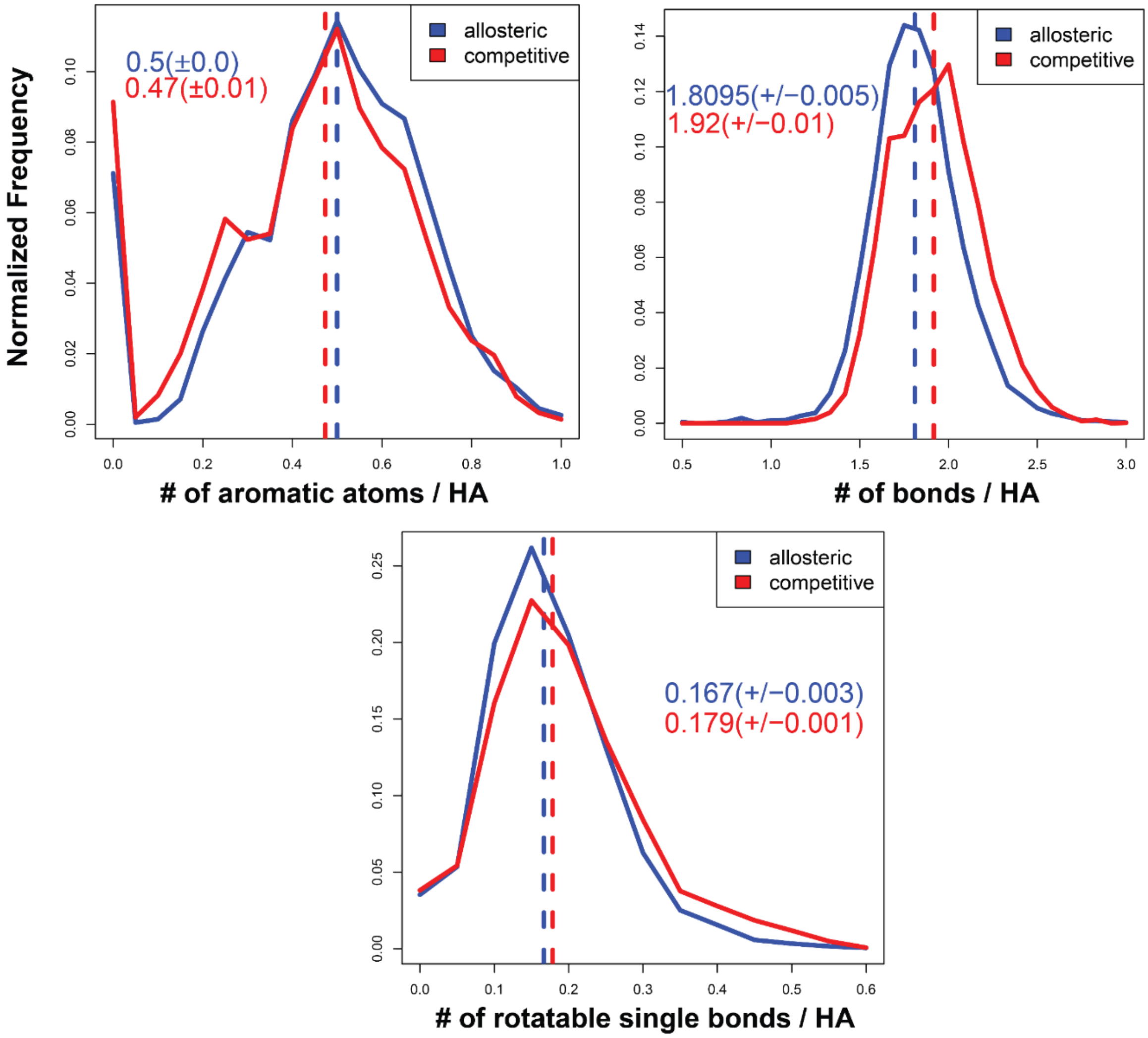


Figure 3-6 The histograms of the number of aromatic atoms corrected by size, the number of bonds per heavy atom, and the number of rotatable single bonds per heavy atom for the full dataset. The median (dashed line) is labeled on the graph with the 95%ci from bootstrap sampling. For brevity, only data clustered at the 60/0.6 level is shown.

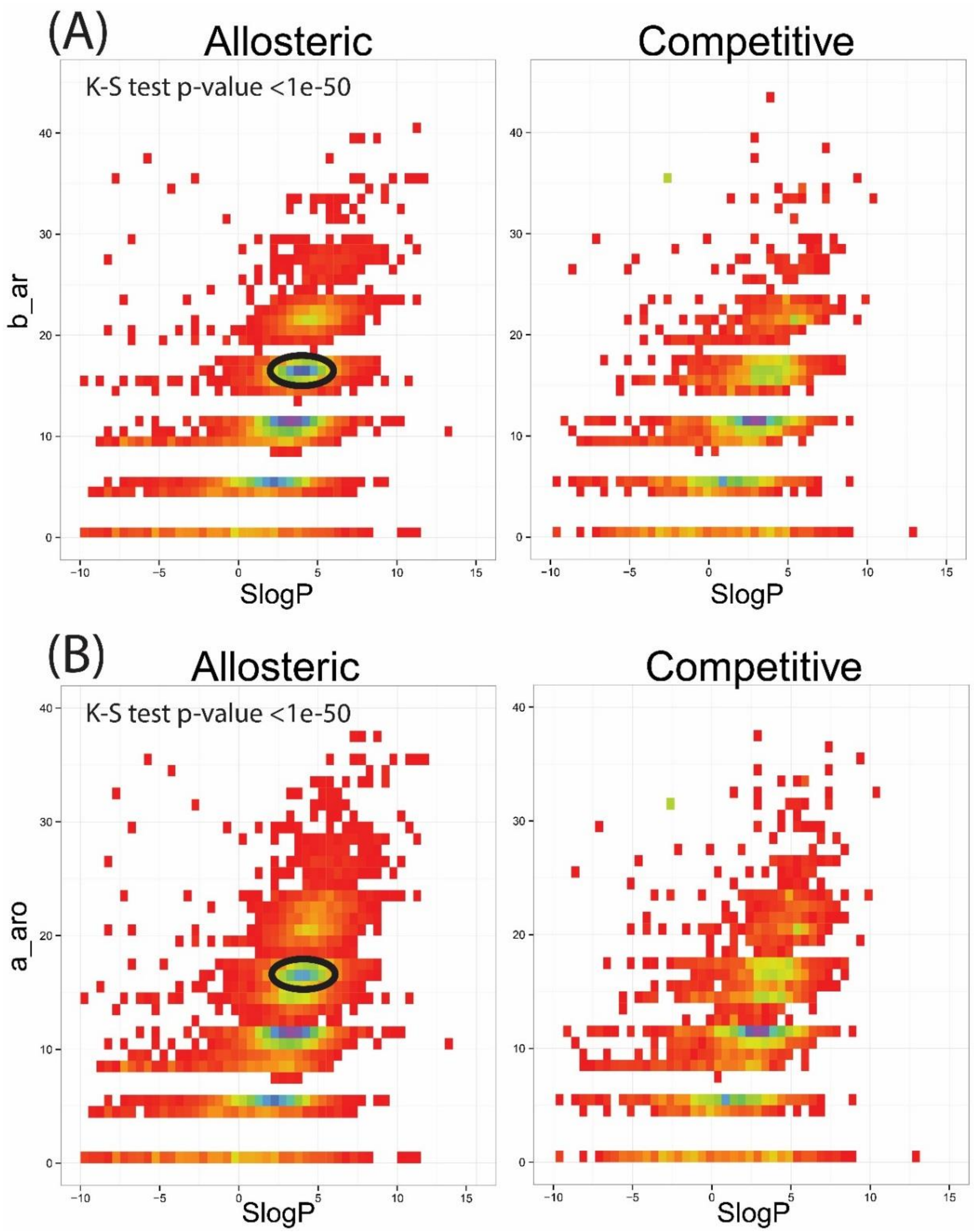


Figure 3-7 For the full dataset clustered at the 60%/0.6 level, 2D density plots showing the distributions of each molecule's SlogP and A) their number of aromatic bonds or B) their number of aromatic atoms. The black circle highlights the populations that contribute most significantly to the greater aromaticity and hydrophobicity of allosteric ligands over competitive ligands. White regions have no density, and the bins are colored from red (low population) to green (middle) to blue/purple (high populations).

The combination of the increase in aromatic atoms and the decrease in the number of rotatable single bonds in the allosteric ligands suggests that these molecules tend to be more rigid. This rigidity is somewhat surprising since allosteric binding sites must undergo a change in conformation upon ligand binding. However, a recent study by Li *et al.* suggested that the pocket flexibility (normalized B-factor) and pocket depth are not significantly different for allosteric binding sites.[246]

The change of protein flexibility upon ligand binding has been indicated by Demerdash *et al.*[220]. That study used a Support Vector Machine (SVM) and indicated that the deformation energy and change in SASA of the protein residues are important features in predicting an allosteric hotspot. The SVM models indicated that allosteric hotspots would form dense networks within the protein. They did not look specifically at the residues in contact with allosteric ligands, and they make no comment on their flexibility[220]. Panjkovich and Daura also noted that a large change in B-factors can be used to indicate the location of allosteric binding sites[253]. The decrease in flexibility of allosteric compounds is consistent with two other studies on large datasets of allosteric compounds[224], [230]. In Wang *et al.*'s work, allosteric ligands have significantly less rotatable bond fraction than drug molecules from DrugBank (their criteria was p-value <0.01, two sample t-test). Van Westen *et al.* describes that allosteric ligands have a higher sp2 hybridized carbon fraction, lower sp3 hybridized carbon fraction, and higher aromatic bonds fraction[230] compared to non-allosteric compounds targeting transmembrane proteins. Taken together, it appears that relatively rigid ligands bind to allosteric sites, inducing a change in flexibility of the protein as it adapts to the presence of the allosteric ligand.

3.4.4.3 "Overlap" molecules are more like allosteric ligands than competitive ligands.

As noted above, we are interested in the molecules with the highest similarity (Tc ≥ 0.9) between the allosteric and competitive sets. Basically, do the sets overlap where allosteric ligands look like competitive ligands or vice versa? The full dataset has 2631 unique molecules that comprise its overlap set, and the same analysis of physical properties is given in Table 3-7. Across all levels of clustering, the physicochemical properties of the overlap compounds are nearly identical to the allosteric set in Table 3-4. These overlap molecules should exhibit both allosteric and competitive mechanisms of action depending on different protein targets. Compound libraries that represent this overlap space may be particularly fruitful for finding leads for protein targets where competitive inhibition is possible, but has proved difficult. It is possible that allosteric leads could be found that take the research in a productive direction for drug discovery.

Table 3-7 For the combined set of highly similar allosteric and competitive ligands (the 2015 "overlap" set), the medians of the significant properties (see Table 3-4) show that the compounds are allosteric in nature.

Clustering level	60%/0.6	75%/0.75	90%/0.9	100%/1.0
	Allo+Comp (Tc ≥ 0.9)	Allo+Comp (Tc ≥ 0.9)	Allo+Comp (Tc ≥ 0.9)	Allo+Comp (Tc ≥ 0.9)
a_don/HA	0.04(±0.00)	0.037(±0.003)	0.034(±0.004)	0.03(±0.01)
SlogP	2.98(±0.08)	3.14(±0.06)	3.40(±0.1)	3.50(±0.1)
b_single	31(±0)	31(±0)	32(±0)	32(±0)
b_single/HA	1.23(±0.03)	1.20(±0.01)	1.19(±0.02)	1.20(±0.01)
b_ar/b_count	0.29(±0.01)	0.30(±0.01)	0.30(±0.00)	0.30(±0.00)
b_count	45(±0)	46(±1)	47(±0)	47(±0)
a_acc/HA	0.11(±0.01)	0.10(±0.01)	0.103(±0.003)	0.1000(±0)
b_1rotN/HA	0.16(±0.01)	0.15(±0.01)	0.15(±0.01)	0.15(±0.01)
b_count/HA	1.83(±0.02)	1.82(±0.01)	1.82(±0.01)	1.83(±0.01)
b_single/b_count	0.68(±0.01)	0.67(±0.01)	0.66(±0.01)	0.66(±0.01)
a_aro/HA	0.50(±0.02)	0.53(±0.02)	0.53(±0.02)	0.55(±0.02)
b_rotN/HA	0.17(±0.01)	0.167(±0.003)	0.167(±0.003)	0.167(±0.003)
b_ar/HA	0.53(±0.02)	0.55(±0.02)	0.545(±0.005)	0.548(±0.002)
a_nC/HA	0.76(±0.01)	0.760(±0.002)	0.760(±0.002)	0.76(±0.01)

3.5 Conclusions

This study aims to elucidate common features of allosteric ligands compared to competitive ligands in order to understand their unique properties. The datasets were carefully curated to ensure the correct designation of their known mechanisms. Verifying the assays assures that we are only comparing allosteric ligands to competitive ligands. Lastly, this study also provides a larger dataset than previous studies performed on allosteric ligands.

The chemical properties of allosteric and competitive ligands were compared. We took great care in normalizing the data so that frequently studied proteins did not overly bias the outcomes. The results indicate that allosteric compounds tend to be more hydrophobic, aromatic, and rigid. This is supported by an increase in SlogP and aromatic atoms per heavy atom. It is also supported by a decrease in chemical saturation and rotatable single bonds. The allosteric ligands also have an increased population of ligands with ~3 aromatic rings. The rigid nature of these ligands, combined with other studies that have shown protein allosteric hotspots are more flexible, suggest that the protein may adapt its conformation to the more rigid ligand and inducing an allosteric conformational change. Lastly, "overlap" compounds are most like allosteric compounds, which means compounds with dual activity are found when the competitive ligands look allosteric, rather than allosteric compounds that resemble competitive ligands.

Chapter 4. Exploring the Effect of Mendelian Disease and Neutral: nsSNPs in Protein Structure - a Large Scale Analysis

4.1 Abstract

This paper focuses on genetic variations which cause amino acid changes and their effect on protein structure and function. Single nucleotide polymorphisms (SNPs) are the simplest and most frequent DNA variation in humans. Non-synonymous missense SNPs (nsSNP) have direct impact on the coding region, causing substitution of protein residues different from wild type. Many disease-causing SNPs have structural or functional impact on the protein depending on the nature of the substitution and where it occurs. We have integrated the nsSNP dataset (UniProt) of nsSNPs, Binding MOAD, and other annotation databases (like UniProtKB) into a single MySQL database. The location of nsSNPs in protein cores, protein surfaces, and ligand-binding sites based on protein-ligand structures and solvent accessible surface area were annotated. The result shows that disease nsSNPs occur more frequently in a protein core or binding site, rather than the rest of the protein surface. Neutral nsSNPs did not show this trend. The disruption of the protein-ligand interaction can be explained by a range structural

effects including the destabilization due to increasing of side chain size, a decrease in flexibility, and the loss of an electrostatic salt bridge.

4.2 Introduction

Over 4000 human Mendelian disorders, which are heritable diseases caused by a single-gene defect[254]. The most common type of single-gene defect in humans takes the form of a single nucleotide polymorphism (SNPs), which is the replacement of a single nucleotide that may have observable impact on the phenotype. Among SNPs, most of the disease-associated gene defects are non-synonymous SNPs (nsSNPs), which are located in coding regions and results in a residue change in the translated protein. Deciphering the link between genetic mutations and a patient’s phenotype is still a major challenge in understanding when and how the variants cause disease. One possible link is that the mutation affects the protein structure which in turn influences biological functions. The location of the amino acid change caused by an SNP on the protein’s structure may be related to detrimental biological consequences. The amino acid residues which are changed may be in the core of the protein, on the surface, or in the ligand binding site. Given the data on both SNPs and protein structures, several questions can be asked and addressed. Are disease-associated missense mutations more likely to occur at certain structure locations? Secondly, are missense mutations at certain locations more likely to be disease-associated? Lastly, what are the residue changes and corresponding consequences of these mutations?

A vast amount of genome data generated by new sequencing methods and international research efforts such as the 1000 Genome Project[255] and HapMap projects[256] allows discovery of disease susceptible mutations genome wide. These methods facilitate massively parallel sequencing in a short time and at low cost[257]. The common SNPs are identified by comparative genome analysis of large-scale sequencing on thousands of individuals [258], [259]. Identification of disease associated mutations is based on statistical analyses of patients' and control group's sequences. Disease or neutral mutations are selected and reviewed in databases such as the Online Database of Mendelian Inheritance in Man (OMIM) [260], dbSNP[261], and the Human Gene Mutation Database (HGMD)[262] based on the published peer-reviewed biomedical literature. The Universal Protein Resource (UniProt) database provides high quality annotations of single amino acid polymorphisms (SAPs) by mapping nsSNP onto protein sequences[263]. These SAPs are selected by manual curation of peer-reviewed literature using strict inclusion criteria. The annotations of SAPs are used to recognize protein sequence locations of disease/neutral mutations.

Meanwhile, structural genomics projects promoted the identification of 3-dimensional structures of proteins in living organisms[264]. The structures are experimentally solved from X-ray crystallography. In order to identify residue locations on proteins, biologically relevant protein-ligand complexes are needed. Careful curation of the Protein Databank has been undertaken recently to identify high-quality protein-ligand structures which can be used to identify where in the protein structure the amino acid mutation caused by an SNP occurs. Binding MOAD is one of the largest databases, which contain high quality protein-ligand complexes from the PDB with ligand annotation (biologically valid/invalid). This database can be used to extract interacting binding residues protein-ligand complexes.

The impact of disease-linked nsSNPs on various functional sites on protein structures is discussed in previous studies[34], [127], [134], [265]. Wang and Moult performed the first study with 262 monogenetic disease mutations from 26 proteins and found that 5% of mutations involve ligand binding and 80% are more likely to destabilize proteins[127]. More recently Dingerdissen et al examined enzyme active sites and showed that nsSNPs occurring at those sites make up <1% of all currently known nsSNPs[265]. In their study both nsSNPs and catalytic sites are mapped to residues of protein sequences from UniProtKB/Swiss-Prot. The result shows that 196 nsSNPs are located at 128 protein active sites[265]. Sun et al performed a molecular docking study with 69 therapeutic drug targets and 232 drugs. Then, distances from mutations to drug binding sites are calculated. They found that the majority (92.4%) of the SNPs are far from the binding sites of the docked drugs (>12 Å)[34]. Gao et al found that disease-associated mutations are much more likely to be found in the functionally relevant ligand-binding pockets created by protein-protein association. Alessia et al's study indicated that protein-protein interaction sites are hot spots for nsSNPs[134] with homology modeling of 537 protein-protein interactions. However, the existing literatures are either limited by a small data set[34], [127], [134], [265] or haven't fully discussed the general impact of nsSNPs on protein-ligand interaction[266].

Given high quality data on both SAPs and biologically valid protein-ligand complexes, the impact of SNPs on protein ligand interactions can be revealed, which can deepen the understanding of molecular mechanisms of disease associations. The result has direct implications for the predictions of disease association and provides possible indication for drug resistance in the early stages of drug discovery.

This study focuses on nsSNPs locations in the protein core, on the surface, and in biological relevant protein-ligand binding sites. In previous studies, for each protein binding site, one protein-ligand structure is used to annotate binding residues. In this study, the union binding site is used to fully capture all essential binding sites by joining binding site residues from multiple structures of the same protein. Many previous large scale studies include cancer mutations[34], [134], [266]. However, the mechanism behind cancer involves multiple mutations impairing proteins in cell cycle control or DNA repair, which is different from Mendelian disease mechanisms. This work focuses on only Mendelian diseases.

4.3 Results & Discussion

4.3.1 Summary of the data set

A large-scale analysis is performed on 3669 disease associated mutations in 242 proteins with experimental structures, as well as 1726 neutral mutations in 580 proteins for comparison. The dataset is much larger than previous studies with only 26, 69, or 128 proteins[34], [265], [267].The mutation and corresponding amino acid change is gathered/compiled based on the August, 2015 release of the UniProt (Universal Protein Resource) database. The structural locations are the union of 5,167 biologically valid protein-ligand complexes downloaded from Binding MOAD 2014. Three locations are analyzed, including ligand binding site, core, and surface (that is not ligand binding site). The locations were identified by mapping the structural information to UniProt canonical sequence. Cancer associated mutations are removed to keep those with Mendelian disorder association. Every mutation collected has at least one high quality protein-ligand structure (has a resolution of 2.5 Å or better). The location preference of these mutations are analyzed in the following section.

4.3.2 Disease-Associated Missense Mutations Have High Preference for Ligand Binding Sites and the Protein Core

The result shows that disease-associated mutations are more likely to have impact on locations that affect protein stability and functionality. Neutral mutations are less likely to happen on locations associated with protein stability. Disease associated mutations are less likely to occur on the protein surface outside of the binding site than within the binding site or protein core. A little more than half (52%) of disease mutations are located on the surface (not binding site) of protein, however surface residues make up ~70% of the protein (Table 4-1). Therefore, the number of disease mutations on the surface is in fact lower than one would expect at random given the large number of surface residues. In contrast, neutral mutations on the protein surface residues occur slightly more than expected at random, since 74% is larger than 70% which one would expect given the number of surface residues. The result is also supported by the odds ratio test. The odds-ratio (OR) of mutations on the surface to other locations in the protein is less than one (OR: 0.461, p-value < 0.05, Table 4-2) further indicating a preference for disease mutations for the binding site. It is not surprising that disease mutations would occur less preferentially on the surface of the protein as surface residues generally do not serve a functional or structural role[268].

Table 4-1. The total number of nsSNPs, the total number of residues in each of the locations, the probability of observing nsSNPs at one location, and odds for observing a nsSNPs at one location.

Mutation Types	Locations	#nsSNPs (frequency)	#total residues in sequences (frequency)	probabilities	odds
Disease (total # 3708)	#in core	1193 (0.33)	47477 (0.22)	0.025	0.026
	#in binding site	564 (0.15)	16237 (0.08)	0.035	0.036
	#surface not binding site	1912 (0.52)	148049 (0.70)	0.013	0.013
	total # nsSNPs	3669	211763	0.017	0.018
Neutral (total # 1716)	#in core	240 (0.14)	47477 (0.22)	0.005	0.005
	#in binding site	205 (0.12)	16237 (0.08)	0.013	0.013
	#surface not binding site	1271 (0.74)	148049 (0.70)	0.009	0.009
	total # nsSNPs	1716	211763	0.008	0.008

Table 4-2. The odds ratio (OR), 95% confidence interval (CI), and propensity for comparing the number of nsSNPs at two locations.

Mutation Types	Locations	odds ratio	95% CI (OR)	propensity
Disease	core vs non-core	1.68	1.57 - 1.81	1.67
	non-surface vs surface	2.17	2.03 - 2.31	2.14
	binding site vs non-binding site	2.23	2.04 - 2.44	2.19
	binding site vs surface non-binding-site	2.75	2.50 - 3.03	2.69
	binding site vs core	1.40	1.26 - 1.55	1.38
Neutral	core vs non-core	0.56	0.49 - 0.64	0.56
	non-surface vs surface	0.81	0.73 - 0.91	0.81
	binding site vs non-binding site	1.64	1.42 - 1.9	1.63
	binding site vs surface non-binding-site	1.48	1.27 - 1.71	1.47
	binding site vs core	2.52	2.09 - 3.04	2.50

Table 4-3. The estimation of mixed effect model.

	Fixed Effects			Random Effects
Location	Slope estimate ± Std. Error	odds ratio	Intercept estimate ± Std. Error	Std. Dev.
Binding sites	1.6145 ± 0.2785	5.025	-7.2091 ± 0.4721	10.02
Surface	-1.5848 ± 0.1607	0.205	-5.8768 ± 0.5032	9.867
Protein core	1.1652 ± 0.1736	3.207	-7.3161 ± 0.4665	10.09

Table 4-4. The odds ratio (OR), 95% confidence interval (CI), and propensity for comparing the number of nsSNPs at two locations for 5 Å^2 SASA as surface residues cutoff.

Mutation Types	Locations	odds ratio	95% CI (OR)	propensity
Disease	core vs non-core	1.52	1.39 - 1.67	1.51
	non-surface vs surface	2.00	1.86 - 2.15	1.97
	binding site vs non-binding site	2.23	2.04 - 2.44	2.19
	binding site vs surface non-binding-site	2.40	2.19 - 2.63	2.35
	binding site vs core	1.41	1.25 - 1.59	1.40
Neutral	core vs non-core	0.55	0.45 - 0.68	0.55
	non-surface vs surface	1.00	0.89 - 1.14	1.00
	binding site vs non-binding site	1.64	1.42 - 1.9	1.63
	binding site vs surface non-binding-site	1.57	1.35 - 1.81	1.56
	binding site vs core	2.70	2.13 - 3.43	2.68

To further support the result in the previous paragraph, a mixed model approach is used to evaluate the effect of nsSNPs located at different proteins. The distribution of nsSNPs on proteins are highly heterogeneous since a large number of mutations are located in a low percentage of proteins; 40% mutations are located on 22 (3%) proteins. This issue will be discussed further in "Check for Robustness" section. Mixed effect model is adopted in order to take the heterogeneity among proteins into account. The results using the mixed-effect model further support that disease nsSNPs have a decreased preference to the protein surface (OR: 0.205, p-value < 0.05, Table 4-3). The standard deviation of random effects in the model reveals the degree of variation that exists in the population of proteins. The standard deviation (9.867 for the model with protein surface residues, Table 4-3) indicates that the distribution of nsSNPs on proteins is highly skewed, thus a check for robustness is necessary to analyze how the skewed distributions affect the results.

Disease mutations are more likely to happen in ligand binding site (OR: 1.40, p-value < 0.05, Table 4-2), when comparing protein core and ligand binding site. The binding site residues only make up 8% of all residues, however, 15% of disease mutations are located in the binding site. The mixed effect model also indicates the importance of binding site. The binding site (vs non-binding site, OR: 5.025, p-value < 0.05, Table 4-3) is favored by disease mutations than protein core (vs non-core, OR: 3.207, p-value < 0.05, Table 4-3) after taking account the heterogeneity among proteins. Both protein core and ligand binding sites are hotspots for disease mutations, but these mutations display an increased preference for ligand binding sites.

Neutral mutations seem to have a decreased preference to be located in the protein core, while have higher preference on non-core and surface non-binding site residues. Neutral mutations have lower probability to locate on protein core (OR: 2.52, p-value < 0.05, Table 4-2). One explanation is that core residues likely play a more important role in maintaining stability than binding site residues[269]. This result are in agreement with previous studies[127], [134], [266], [269].

The preference of neutral mutations in the ligand binding site is ambiguous. For the full data set, ligand binding sites are preferred by neutral mutations when compared to non-binding site residues (OR: 1.64, p-value < 0.05, Table 4-2). One of the reasons might be some proteins are more resistant to mutations in the ligand binding site, especially for proteins with a large number of binding site residues. The preference for binding sites becomes insignificant (binding site versus non-binding site OR: 0.810, p-value > 0.05), if proteins with more than 60 binding site residues are removed (22 out of 671). This result is also valid for cutoffs of 45, 50, 55, 60, 65, and 70 binding site residues

4.3.3 Robustness check for location preference of disease mutation

4.3.3.1 Varying the cutoff for surface residue.

The solvent accessible surface area (SASA) cutoff adopted to identify surface residues may affect the result, since different cutoffs will generate a different distribution of amino acids. The common standard is 5 Å^2 SASA [270], [271]. A cutoff of 0.5 Å^2 SASA allows for more residues to be identified as surface residues. For this cutoff, the number of surface residues increases from 148,049 to 21,447. The preference odds ratios of 0.5 Å^2 cutoff shown in Table 4-4 is still consistent with those under 5 Å^2 cutoff.

4.3.3.2 Varying the cutoff for binding site residue.

Binding site residues are identified by a distance cutoff, which indicates the energy between atoms. 4 Å is widely accepted as the cutoff for two contacted carbon atoms. Because the Van der Waal's radius of carbon is 1.7, the contact distance cutoff should be 1.7 * 2 plus a tolerance [272]. Difference tolerance value 0.1 Å, 0.6 Å, 1.1 Å are used to check the robustness. The preference result (in Table 4-5) is still valid given these cutoffs. The change of cutoff does not dramatically alter the number of binding site residues. The number of binding site residues decreases by 25.5% after reducing the cutoff to 3.5 Å and increases by 12.7% after extending the cutoff to 4.5 Å.

Binding site residues with shorter contact distances to ligands may have a higher interaction energy and an increased chance for protein-ligand interactions. The mutations of these residues are more likely to disrupt ligand interaction, thus influence the functionality. A steady increase in the preference of disease mutations in binding site residues is observed as the cutoff for interactions is decreased. The odds ratios of binding site residues to non-binding site residues would be 2.14 (cutoff: 4.5 Å), 2.23 (cutoff: 4.0 Å), 2.41 (cutoff: 3.5 Å), and 2.64 (cutoff: 3.0 Å).

Table 4-5. The odds ratio (OR), 95% confidence interval (CI), and propensity for comparing the number of nsSNPs at two locations for different distance cutoff (3.5 Å, 4.0 Å, and 4.5 Å) of ligand binding residues.

		3.5 Å			4.0 Å			4.5 Å		
Mutation Types	Locations	odds ratio	95% CI (OR)	propensity	odds ratio	95% CI (OR)	propensity	odds ratio	95% CI (OR)	propensity
Disease	core vs non-core	1.71	1.59 - 1.83	1.69	1.68	1.57 - 1.81	1.67	1.68	1.57 - 1.8	1.66
	non-surface vs surface	2.15	2.02 - 2.3	2.12	2.17	2.03 - 2.31	2.14	2.17	2.03 - 2.32	2.14
	binding site vs non-binding site	2.41	2.19 - 2.67	2.36	2.23	2.04 - 2.44	2.19	2.15	1.96 - 2.34	2.11
	binding site vs surface non-binding-site	2.96	2.67 - 3.28	2.89	2.75	2.50 - 3.03	2.69	2.65	2.42 - 2.91	2.60
	binding site vs core	1.52	1.36 - 1.69	1.50	1.40	1.26 - 1.55	1.38	1.34	1.21 - 1.48	1.33
Neutral	core vs non-core	0.57	0.5 - 0.65	0.57	0.56	0.49 - 0.64	0.56	0.56	0.49 - 0.65	0.56
	non-surface vs surface	0.79	0.71 - 0.89	0.79	0.81	0.73 - 0.91	0.81	0.81	0.73 - 0.91	0.81
	binding site vs non-binding site	1.75	1.49 - 2.06	1.74	1.64	1.42 - 1.9	1.63	1.55	1.35 - 1.79	1.55
	binding site vs surface non-binding-site	1.58	1.34 - 1.86	1.58	1.48	1.27 - 1.71	1.47	1.40	1.21 - 1.61	1.39
	binding site vs core	2.66	2.18 - 3.24	2.64	2.52	2.09 - 3.04	2.50	2.38	1.98 - 2.86	2.36

4.3.3.3 The impact of proteins with more than 50 mutations

The distribution of mutations on different proteins is highly skewed to the right; 68% of proteins have less than 5 mutations. More than 40% of mutations are concentrated at 22 (3%) proteins. Each has more than 50 mutations. Analysis of the subset with overrepresented proteins removed is necessary to understand whether the characteristics only belonged to these proteins would dominate the result.

The associated diseases of the 22 proteins with a large number of mutations are mostly related to metabolic processes, for example Phenylketonuria (160 mutations), Fabry disease (155 mutations), Gaucher disease (131 mutations), Factor VII deficiency (101 mutations), pyruvate kinase deficiency of red cells (99 mutations), Leukodystrophy metachromatic (96 mutations), Hemophilia B (85 mutations). A metabolic disease can be associated with a large number of genetic mutations because mutations can affect metabolic processes in various ways, including influencing enzyme structure and/or function, impeding transport and processing, or disrupting binding of cofactor.

The disease mutation preference for the protein core and ligand binding sites is robust after removing proteins with high number (>50) of mutations. No significant change on the odds ratio are observed when comparing protein core to non-core residues (OR: 0.56 to OR: 0.55) and binding site to non-binding site residues (OR: 2.23 to OR: 2.40). Also, neutral mutations still have high preference for binding sites compared to non-binding site residues (OR:1.64, p-value < 0.05) and non-core residues compared to core residues (OR: 0.56, p-value < 0.05).

4.3.3.4 Why removing somatic mutations is necessary?

Many previous studies[134], [266] and prediction methods[131], [273] include somatic mutations (mostly treated as neutral mutations), neutral mutations, and mutations from Mendelian diseases (germ line mutations). However, the somatic mutations may have different characteristics than other neutral mutations. There are two kinds of somatic mutations. The mutations that promote cancer development by providing a selective growth advantage are termed driver mutations, and those that do not are termed passenger mutations [274]. Most somatic mutation are passenger mutations [275]. The passenger mutations are treated as neutral mutations in previous studies[134], [265], because they don’t explicitly cause any disease.

Neutral mutations do not show significant preference to binding site after removing proteins with a large number (> 45) of binding site residues. The somatic mutations tend to favor binding site residues, even after removing overrepresented proteins. This result is valid under cutoffs with different numbers of binding site residues. The odds ratio of nsSNPs located at ligand binding site comparing to those located at other places is greater than 1.44 and is significant (p-value < 0.05) for all cutoffs. These odds ratios are 1.55 (45 residues), 1.50 (50 residues), 1.46 (55 residues), 1.44 (60 residues), 1.56 (65 residues), and 1.58 (70 residues).

The preference of somatic mutations to ligand binding sites are also supported by a much larger set. The larger data set is created by joining by proteins used in this study with COSMIC, covering 1,099 unique proteins, 1,905 PDB files, and 315,861 unique mutations. The ligand binding sites are the most preferred locations compared to non-binding site residues located on the surface (OR: 1.39, p-value < 0.05) and the protein's core (OR: 1.58, p-value < 0.05). This result is also validated by removing proteins with large number of mutations under cutoffs with different numbers of mutations on proteins (45, 50, 55, 60, 65, and 70).

The inclusion of somatic mutations in the neutral mutations diminishes the preference of disease mutation to ligand binding sites. This is supported by the odds ratio of disease nsSNPs on ligand binding sites to those on other locations in the mixed effect model. The odds ratio increases significantly from 1.85 to 5.025 after removing somatic mutations. Thus, somatic mutations are more likely to locate at ligand binding site than other neutral mutations, which is similar to disease mutations.

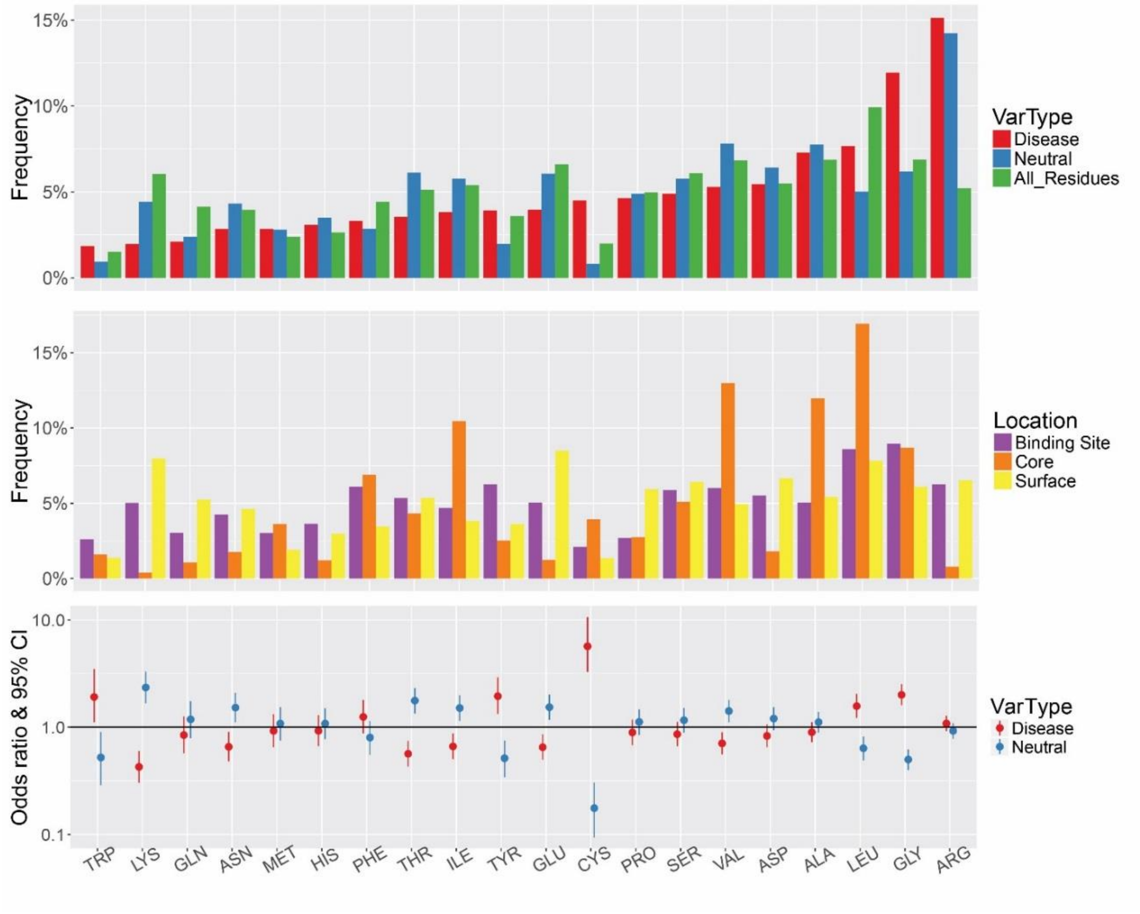


Figure 4-1. Side chain frequencies on entire protein with mutation (top). Side chain frequencies on protein structure (middle). Amino acid preference for all residues for disease/neutral nsSNPs (down). Frequency is divided by total number of residues and sorted by binding site residue. The data are sorted by frequencies of disease associated nsSNPs.

4.3.4 Amino acid preference

In general, the tendency for disease nsSNPs is to mutate amino acids which play critical roles in protein stability and functionality. Disease nsSNPs have high preference to mutate tryptophan, cysteine, glycine, tyrosine, arginine, and leucine (one mutated amino acid type versus other mutated amino acid types OR > 1, p-value < 0.0016). Tryptophan and cysteine only make up 3.5% of all residues and 4.7% of binding site residues (Figure 4-1), so, the confidence interval for the odds ratio for these two amino acids is large. The mutations of these tryptophan and cysteine are more likely to be disease-associated. These amino acids perform critical function for a protein, and they are different than looking at the binding site as these structural functions would not occur at binding sites. For example, tryptophan-tryptophan pairs in β-hairpin peptides were shown to contribute significantly to the stability[276]. Additionally, two cysteine residues can bond to form a disulfide bond, which plays an important role in protein folding and stability. The importance of cysteine is also supported by a very low preference for being a neutral mutation. Glycine, tyrosine, and leucine are also preferred by disease mutation, and they are among top four of the most frequent amino acid in binding sites (Figure 4-1). This is consistent with the above result that ligand binding site are hotspots for disease mutations.

Disease nsSNPs are less likely to mutate some hydrophilic amino acids, including lysine, glutamic acid, threonine, and asparagine. Hydrophilic amino acids are mostly located on the protein surface. Lysine is generally located on the protein surface with 92% of lysine residues on the surface. The same is true for glutamic acid (90% on protein surface) and asparagine (81% on protein surface). Other than hydrophilic amino acids, valine and isoleucine are also less likely to be mutated by disease SNP. Valine is hydrophobic and 42% of them are located in protein core.

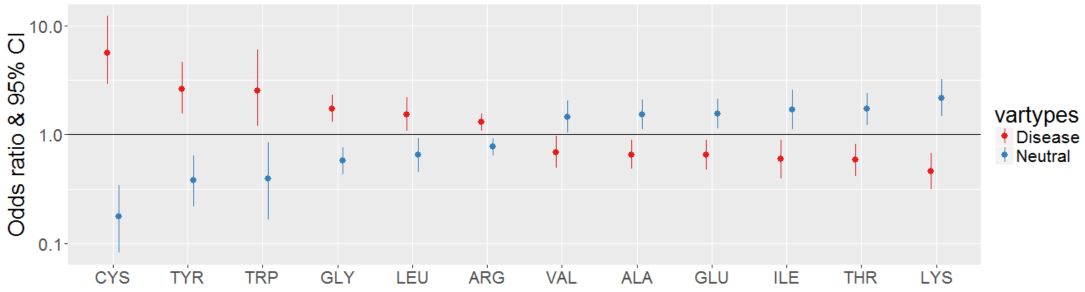


Figure 4-2. Amino acid preference on protein surface for disease/neutral nsSNPs. The data are sorted by estimated value of odds ratio of disease associated nsSNP.

In order to understand the impact of property changes of amid acids, the investigation of amino acid changes by disease and neutral mutation is conducted for each protein location (i.e. protein surface, binding site, and protein core). For the protein surface in Figure 4-2, disease mutations have significant preference to occur with amino acids which are more likely to participate in hydrogen bonds (cysteine, tyrosine, tryptophan, glycine, and arginine). Leucine and valine do not appear significant in the chi-square test because they are hydrophobic and thus under represented on the protein's surface.

In the protein core, disease nsSNPs are more likely to happen with amino acids that are used as a critical component in forming structure (Figure 4-3). Glutamic acid, asparagine, leucine, and glycine are significantly higher for disease nsSNPs. Glutamic acid and asparagine have low composition (<3% of all core residues) in protein core, but the mutation of glutamic acid can dramatically change the charges of local structure environment[277] and asparagine is used near the beginning of alpha-helices, which are essential to form the secondary structure[278]. Leucine and glycine have high composition (>20% of all core residues) in protein core. Leucine is one of the most prominent residues in β-strands, which play an essential role in the secondary structure[279]. Only glycine is allowed at certain turns of tight bends due to steric restrictions because of the small side chain size. The mutation of glycine can be deleterious, which are also supported by the low preference for neutral nsSNPs (OR: 0.57, CI: 0.33-0.92, p-value: 0.0176).

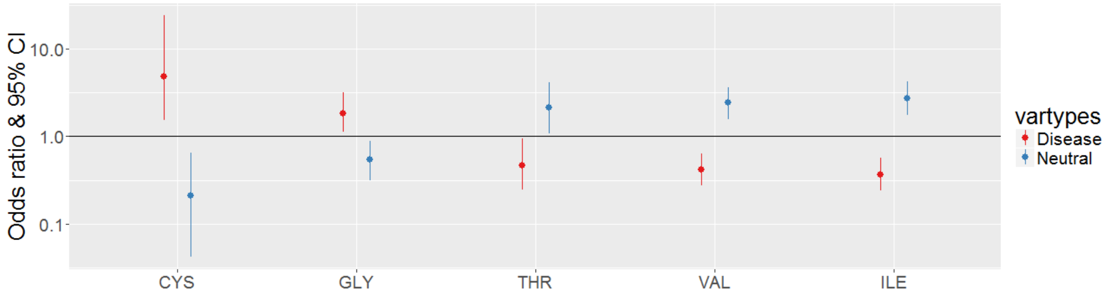


Figure 4-3. Amino acid preference on protein core for disease/neutral nsSNPs. The data are sorted by estimated value of odds ratio of disease associated nsSNP.

For protein-ligand binding sites, disease mutations occur with amino acids more likely to affect the hydrogen bonding and flexibility (Figure 4-4). Binding sites have high preference for mutations of proline, glycine and arginine. The mutation of arginine may abolish a salt bridge between binding site and ligand. The mutation of glycine may have impact on protein flexibility or block protein ligand binding by increasing the size of the side chain. In the meantime, neutral nsSNPs have less preference for glycine (OR: 0.48, p-value < 0.05).

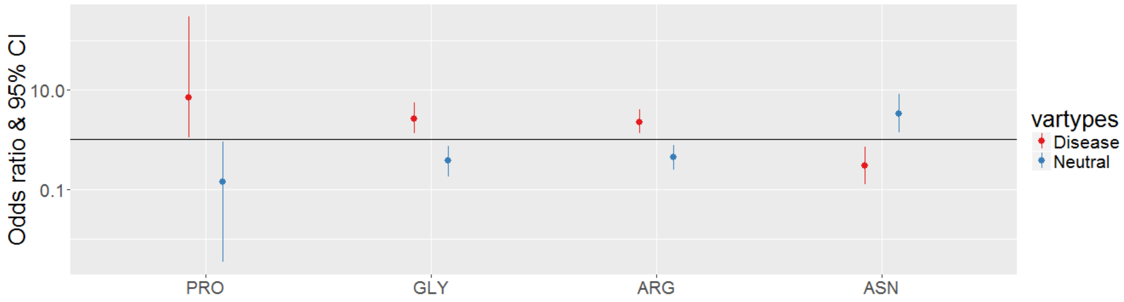


Figure 4-4 Amino acid preference on protein binding site for disease/neutral nsSNPs. The x-axis labels are sorted by estimated value of odds ratio of disease associated mutations. The data are sorted by estimated value of odds ratio of disease associated nsSNP.

4.4 Methods

4.4.1 Preference calculation

The preference of one type of nsSNP to a protein structural location i over location j is quantified by propensity and odds ratio. The protein structural location could be protein core, protein-ligand binding site, or protein surface. Propensity is widely used in many structural related studies, such as secondary structure propensity in model proteins[280], amino acid propensities for secondary structures[281], and amino acid propensities for protein-ligand binding sites[282]. Odds ratio is corresponding to the parameters of following mixed effect model. Propensity and odds ratio are calculated as follows.

The probability of observing an nsSNP in protein structural region i is

$$x_i = \frac{n_i}{N_i},$$

Where n_i is the total number of residues in region i, and N_i is the number of residue with nsSNP in region i. With probability, the odds for nsSNP in region i is

$$ODDS_i = \frac{x_i}{1-x_i}.$$

To quantify the preference between region i and region j, odds ratio is calculated by

$$OR_{ij} = \frac{\frac{x_i}{1-x_i}}{\frac{x_j}{1-x_j}}.$$

Propensity is calculated by

$$Propensity_{ij} = \frac{x_i}{x_j}.$$

If the odds ratio or propensity < 1, then nsSNP has higher preference to region i compared to location j. With the distribution of log odds ratio, a two-tailed p-value can be calculated by R package epitools[283]. A p-value less than 0.05 is considered significant.

With a similar idea, for amino acid preference, the probability of observing an nsSNP associated with variant type is

$$y_{ak} = \frac{n_{ak}}{N_k},$$

where a is amino acid type, i.e. *Ala, Arg, Cys...etc*, N_k is the total number of amino acid with nsSNP k, and n_{ak} is the number of amino acid *a* with nsSNP k in certain structural region. The type of mutation, k, can be either disease, polymorphism, and unclassified.

For the probability of amino acid in the same region, the probability of observing the amino acid is:

$$z_a = \frac{n_a}{N}$$

where N is the total number of amino acid in the region, and n_a is the number of amino acid a in that region. The odds ratio can be expressed as:

$$OR_{ij} = \frac{\frac{y_{ak}}{1 - y_{ak}}}{\frac{z_a}{1 - z_a}}$$

4.4.2 Mixed effects model

The odds ratio or propensity may be biased for proteins with many mutations. To understand whether the preference is a general trend for most proteins, building a mixed effect model is necessary given a skewed SNP distribution among proteins. The skewed SNP distribution in proteins has not been discussed in much detail in previous studies, partially because previous data sets do not have a large number of nsSNPs[127],

[134] or they deal with this problem by randomly selecting 50 nsSNPs for proteins with more than 50 nsSNPs[266].

A mixed effects model is a statistical model which is particularly effective in accommodating data that are collected in groups. In mixed effects model, the coefficient can vary with respect to grouping variables (e.g. protein identity in this study). The model contains two parts: fixed effects and random effects. Fixed-effects ($X\beta$) terms represent individual specific effects (e.g. location, type of amino acid) that may be correlated with the response (e.g. disease/neutral). Random-effects (Zb) estimates the variability of different groups (e.g. protein identity).

$$y = X\beta + Zb + \varepsilon$$

The significance of fixed effects in the model is evaluated by F tests via Kenward-Roger approximation[284]. The model takes heterogeneous nsSNPs distribution among proteins into account when evaluate the significance of association between nsSNP location and consequence (disease/neutral).

4.4.3 Datasets

The annotation of location for binding site residues and nsSNPs requires both high quality binding site data and SNP data. Binding MOAD [141], [285] is one of the largest collection of high resolution (2.5 Å or better) protein-ligand structures available from Protein Data Bank (PDB)[286]. Every ligand in Binding MOAD has annotation (biologically valid vs crystallographic additive invalids). The annotation is hand curated by reading the peer-reviewed crystallography paper, which makes it unique among databases of this kind. The curation step is accelerated by guidance from natural language processing and text mining. Binding MOAD is updated once a year.

UniProt is currently the largest collection of protein sequences and annotations for protein nomenclature, function, and important residues. UniProt provides comprehensive high-quality annotation based on labor-intensive literature-based expert curation. In terms of single amino acid polymorphism, a collection of information on human genetic diseases and variants are provided and manually reviewed by experts. All relevant biological knowledge for amino acid changes are linked, organized and made readily available to users. The relevant information includes gene-phenotype relationship characterized in Online Mendelian Inheritance in Man (OMIM) and identified genetic variation collected in Single Nucleotide Polymorphism Database (dbSNP). The manual annotation is based on statistical genetics considerations, computational methods predicting deleteriousness, and experimental evidence of variant effect on protein properties.

The nsSNP annotation in UniProt's human-variant data set has three categories: disease, polymorphisms, and unclassified[8]. "Disease" mutations are mutations associated to a known disease with statistical significance from peer-reviewed literatures. The experimental information for these nsSNPs are retrieved from the Single Nucleotide Polymorphism Database[9] (dbSNP) database. nsSNPs with a minor allele frequency (MAF) less than 0.1 will be further investigated to assess the disease-causing effect by more literature search. Most disease variant in the data set have a corresponding entry in the Online Mendelian Inheritance in Man Database[10] (OMIM). “Polymorphisms” are nsSNPs for which there is not any evidence of association with a disease. OMIM, however, does not have a complete list of all possible disease-causing variants, therefore some polymorphism nsSNPs may in fact be disease-associated. More than 90% polymorphism nsSNPs with corresponding dbSNP entries show that they are not rare, disease-causing mutations[4]. “Unclassified” nsSNPs are identified in a pathological sample, but lack statistical or experimental evidence to prove the disease association. Both unclassified and disease nsSNPs are rare variants with MAF less than 0.1.

UniProt human variant data includes a few thousands of somatic mutations. To obtain a better portrait of the location and mutated amino acids of somatically acquired mutations, the Catalogue Of Somatic Mutations In Cancer (COSMIC) database is utilized. The COSMIC database was specifically designed for collecting somatic mutations, curated from peer-reviewed papers and large scale experimental screen from The Cancer Genome Atlas[287] (TCGA; http://cancergenome.nih.gov) and International Cancer Genome Consortium[288] (ICGC; https://dcc.icgc.org) projects. The mutation information includes annotations of disease types and patient details. To retrieve the corresponding information for each experiment and mutation, COSMIC provides graphical Web system to help users navigate gene-centrically or tissue-centrically. The data set (COSMIC v70; Aug 2014) used in this study includes information from over 2.0 million coding mutations on 1.02 million samples.

4.4.4 Definition of surface residues, ligand binding residues, and core residues

Protein surface residues are recognized by solvent accessible surface area, quantified by DSSP[12]. DSSP rolls a water sphere (radius 1.4 Å) around the van der Waals surface of the protein to find all possible positions that are in contact with protein heavy atoms. For the protein-ligand complex, all known ligands are removed before calculating solvent accessible surface area (SASA). The default probe size is used, and waters and HETATOMS are ignored. Two definition of surface residue (≥ 5 Å^2 and ≥ 0.5 Å^2 SASA) are tested. For one protein with multiple biounit complexes, the maximum SASA is assigned to the residue so the flexibility of protein is also taken into account. The ligand binding residues are identified by a distance of 4.0 Å or less to the nearest biological valid ligand. Other cutoffs (3.5 Å, 4.5 Å) are also tested. An amino acid is considered a cored residue if it is neither surface nor binding site residue.

Chapter 5. Conclusions and Future Directions

5.1 Significant contributions of this thesis

Structure-Based Drug Discovery techniques reduce the time and cost for drug research and development (R&D) by providing efficient computational tools for the identification of potential hit molecules as starting points for drug discovery. Development of high quality ligand libraries and identifying binding sites of feasible target are critical for the success of SBDD. This thesis focuses on improving SBDD in two aspects, the construction of the ligand libraries and analysis of the target binding sites used. The visual exploration of compounds and differentiable physicochemical properties of allosteric ligands are discussed in Chapter 2 and 3, and the effect of genetic mutations on the target binding sites is studied in Chapter 4.

To investigate the depth and breadth of ligand datasets, ChemTreeMap was developed in Chapter 2. ChemTreeMap is a tool used to explore chemical space and analyze the relationships between chemical structures and their physicochemical properties and/or biological activities. ChemTreeMap organizes chemical compounds in a hierarchical tree structure with branch length proportional to the value of molecular similarity. Molecular similarity is quantified by ECFP, which is able to capture substructures and global similarity, but other similarity techniques can be incorporated. The hierarchical relationship and branch length is computed by the Neighbor-Joining algorithm, which has been widely used in building phylogenetic trees for large diverse sequences. Given the tree structure representation of molecular similarity, associated properties are shown by leaf color, size, and outline, which can be changed interactively by users.

In order to illustrate the advantage and capability of ChemTreeMap, two possible applications were presented in comparison with other tools. The first application assessed overlap between datasets. ChemTreeMap was used to delineate newly covered chemical space between ChEMBL and ChemBank, as well as that between ChEMBL and BindingDB. Each data set is very large in terms of the number of molecules (ChEMBL: ~1.3 million chemicals, ChemBank: ~1.15 million, BindingDB: ~0.5 million). ChemTreeMap highlights a large region of chemical space which is unique to ChemBank and to ChEMBL. Combining ChEMBL and ChemBank can increase the breath of chemical structures. For adding BindingDB to ChEMBL, ChemTreeMap shows that no large branches are dominated by BindingDB. This argumentation would increase the number of molecules with small modifications to those which already exist in ChEMBL, which adds depth of coverage.

The second application is for mining structure-activity relationships (SAR) in two protein-specific datasets. SAR of FXa and CDK2 are analyzed by ChemTreeMap and four other similar tools. With ChemTreeMap, the chemical core responsible for activity was successfully identified. “Activity cliffs” and SAR hot spots can be identified visually with ChemTreeMap by change of node color.

For the analysis of SAR, other tools (e.g. SPT, Data Warrior) do not present global similarity of all molecules well, are overcrowded in areas with similar molecules (CheS-Mapper), or they lose the relationship between molecules sharing a similar scaffold (Scaffold Hunter). The comparison demonstrates ChemTreeMap’s ability to deal with rather complex chemical analysis of diverse molecules since the pattern is clear from the visual representation.

To build a molecular library biased towards allosteric compounds, the physicochemical differences between allosteric and orthosteric competitive compounds were discussed in Chapter 3. We showed that allosteric ligands tend to be more hydrophobic, aromatic, and rigid as compared to orthosteric, competitive compounds. The data set used in this chapter has an increased coverage of known allosteric compounds by combining both ASD and ChEMBL than those of previous works that mined for generic properties of allosteric ligands. The work is innovative in dealing with overrepresented molecules, which have not been carefully handled in previous studies. Two-level clustering is conducted in this chapter to ensure redundancy is removed at both the protein and ligand level. Similar ligands are then grouped into clusters. The representative for each cluster is investigated choosing only the cluster center or weighting each molecule’s properties by the number of molecules in its cluster. This study focused on properties that can be experimentally measured and can be predictively modified. To demonstrate the robustness of the result, several statistical metrics were used to find the most persistent patterns. The statistical metrics include the Wilcoxon test if the cluster is represented by the center molecule, weighted Wilcoxon test if the representative is chosen by weighting the molecular properties, and overlap in the 95% confidence interval from a bootstrap based on weighted distribution. The overrepresented chain assemblies and ring structures and their impact on physicochemical properties were discussed. The data showed that allosteric ligands tend to be more hydrophobic, aromatic, and rigid. The differentiable physicochemical properties can be used as guidance for future design of allosteric molecules.

To select target binding sites that may be less likely to be influenced by disease-linked mutations, Chapter 4 focuses on the effect of Mendelian disease-associated genetic variations on protein structure and function. In previous work, the impact of disease-linked nsSNPs on various functional sites are discussed, including protein-protein interaction sites, the binding sites of docked drugs, and ligand binding pockets created by protein-protein association. This work is innovative on studying nsSNPs associated with Mendelian disease and using a “union ligand binding site” to identify interactive residues. The union binding site is a collection of residues from all experimentally determined protein-ligand complexes. The result shows that while mutations to protein cores are common among disease associated nsSNPs, mutations are actually statistically enriched at the observed ligand binding site. After taking the heterogeneous distribution of nsSNPs on different proteins into account, the result of the mixed effect model showed that the ligand binding site is most preferred by disease associated nsSNPs. The robustness of the result has been tested by changing cutoffs for surface residues and binding site residues. The result is also valid after removing overrepresented proteins with more than 50 nsSNPs. After investigating mutated amino acid at ligand binding sites, the data suggests that the interruption of ligand binding is caused by destabilization of protein structures, a decrease in binding site flexibility, and/or the loss of electrostatic salt bridges.

To improve ligand libraries and identifying feasible target binding sites, this thesis first provides a new visualization tool based on a phylogenetic tree to investigate structure property relationships, then discovers several guiding physicochemical properties for construction of ligand libraries biased to allosteric mechanism, and discusses possible impact of disease mutation on protein structure focusing on binding sites.

5.2 Future Directions

The potential extension for the thesis is listed as follows.

5.2.1 Incorporating networks of similarity across binding sites of protein targets

The chemical visualization tool, ChemTreeMap, is capable of displaying a molecular set with diverse structures. A proper way to display drug-target information has not yet been developed. Effective drugs may act on multiple targets rather than a single target.

A new approach, polypharmacology, is emerging with increasing understanding of the role of protein-protein and protein-ligand interaction networks in the robustness of biological systems. Polypharmacology is the concept that one compound specifically binds to two or more targets. Polypharmacology has shown promise for tackling the two major challenge in drug development - efficacy and toxicity [289]–[291]. However, the design of molecules based on polypharmacology faces considerable challenges, including exploring target combinations, and optimizing ligand efficiency while maintaining drug-like properties [290]. Visualization of both ligand and protein binding site similarity could provide intuitive understanding for this multi-objective optimization problem. In the future, a natural progression for ChemTreeMap would extend the current visualization tool to support exploration of possible binding targets for a molecule.

5.2.2 Shared common substructure and SAR alerts on the ancestor nodes

The parent nodes in ChemTreeMap can be used to represent the common features of all descendent molecules. Future enhancements for ChemTreeMap can display the common substructures and SAR alerts at parent nodes (i.e. where the tree branches). The common substructures can be generated by a general graph matching algorithms[292]–[294]. SAR alerts include measures, such as Structure Activity Landscape Index (SALI) [295], SAR Index (SARI) [296], and Pan-Assay Interference Compounds (PAINS)[297]. Incorporating and visualizing these scores in ChemTreeMap would help to identify shared chemical scaffolds and regions where large changes in compound potency with small or even moderate changes in chemical structure.

5.2.3 Analyzing allosteric binding sites

In Chapter 3, the differentiable physicochemical properties are found for allosteric compounds. The next logical step would be to analyze the allosteric binding sites to reveal characteristics of allosteric proteins and allow for coupling the structure of allosteric sites and the structure of their modulators. The number of allosteric binding sites with crystal structures has grown rapidly in recent years because of the significant expansion of allosteric drug discovery[26]. The allosteric binding sites analysis would provide insight into the foundation of allosteric interactions and matching features to allosteric compounds. The properties of both allosteric modulators and binding sites could shed light into how the allosteric regulation is triggered. This extension would improve the understanding of the essential features for allosteric site prediction and the design of proteins with allosteric mechanisms.

5.2.4 The effects and mechanism of the mutations around the allosteric site

A major finding from Chapter 4 of this thesis is that disease mutations are more likely to mutate interacting residues for ligand binding. However, the general impact of mutations on or around protein allosteric sites has not been fully explored. With the recent growth in both allosteric binding site annotation and disease associated mutations, the properties of mutations specifically in the allosteric sites will provide new insight into physiological abnormalities and the progression of genetic diseases. It is possible that disease-associated mutations are less likely to occur in allosteric binding sites than orthosteric competitive sites.

5.2.5 Comparing with other control sets to analyze disease associated mutations

For the analysis in Chapter 4, one could argue that mutations collected by UniProt may not represent a good control set to differentiate disease associated mutations. Two kinds of control sets could be used: 1. all possible single mutations for a genome and 2. the largest observed set of neutral mutations. For the first control set, the analysis of amino acid composition would be performed to get the most preferable amino acids by comparing disease associated mutations to mutations generated by random DNA single mutation. For the second control set, a large set of variants in about 1,000 healthy individuals could provide a neutral background. This would be collected from the 1000 Genomes Project[259]. The results on location and amino acid type preference can be validated on these two control data sets by checking whether the mixed effect model still show statistical significance on binding sites and certain amino acids. We expect these results are still valid, because the functional impact of disease mutations will be consistent across different control sets.

Appendices

Appendix A. ChemTreeMap Supplementary Information

S1. Neighbor-joining algorithm pseudo code

```
T is an empty tree structure

D is a N x N distance matrix for all pairwise distance


WHILE dimension of D > 2

FOR i = 1 to num_mols

  FOR j = 1 to num_mols

        u[i] = D[i][j] / (n - 2)

  END FOR

END FOR


FOR i = 1 to num_mols

  FOR j = 1 to num_mols

        Q[i][j] = D[i][j] - u[i] - u[j]
  END FOR
END FOR


min_i, min_j = find the index i, j with minimum Q
vi = 0.5 * D[min_i][min_j] + 0.5 * (u[min_i] - u[min_j])
vj = 0.5 * D[min_i][min_j] + 0.5 * (u[min_j] - u[min_i])
add node ij, min_i, min_j to tree T
add edge ij -- min_i, ij --min_j to tree T with length vi, vj


FOR k = 1 to num_mols
  D[num_mols + 1][k] = (D[min_i][k] + D[min_j][k] - D[min_i][min_j]) / 2
END FOR


remove column i, j from D
ENDWHILE
```

S2. Radius of display

ChemTreeMap's display is dynamic, meaning that the leafs are mobile. This allows users to reposition branches if they desire. The positions for the leafs is maintained by branches holding nodes together, nodes repelling one another (to avoid overlap), and a central force that keeps all nodes within a radius.

The "radius of display" slider actually increases/decreases the force from the center of the tree out to the leaves. A short, tight radius pulls the nodes closer to the center of the tree. A larger, loose radius allows the nodes to move farther away from one another. The branching can be easier to see with a large radius. The ability to zoom in and out allows users to still focus on a local region or pan out to see the whole dataset, as we have done below. The dataset for CYP3A4 is shown below.

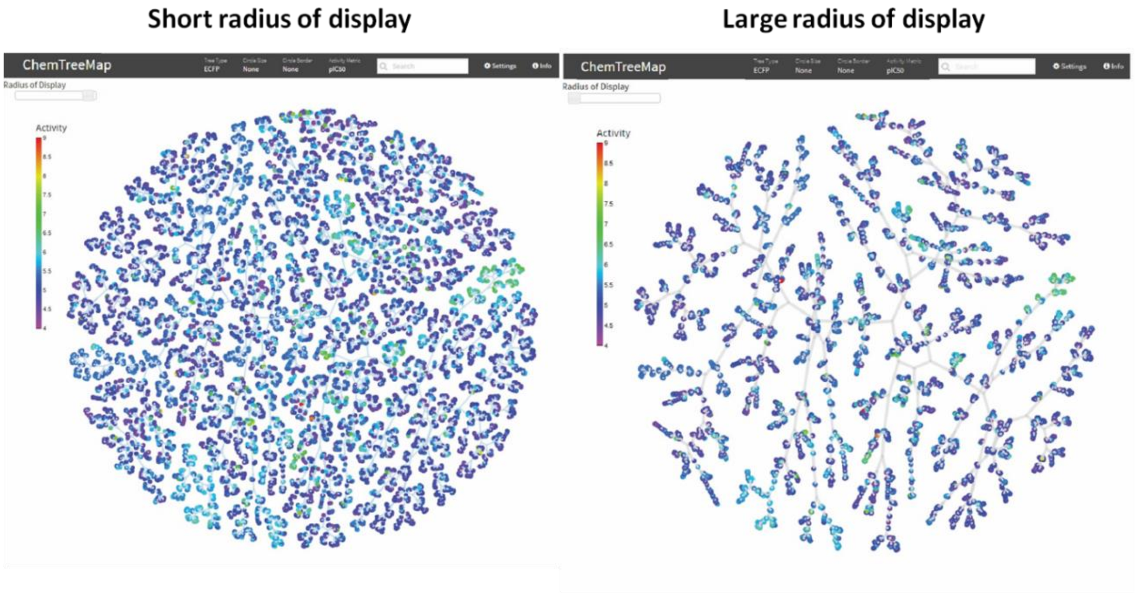


Figure A-1 The effect of radius of display.

S3. SAR in p38α data

Activity switches are a slight variation on activity cliffs; switches involve groups of active compounds with large differences in potency, yet the structures are similar analogs. They are important for the detection of structural features that effect activity. In ChemTreeMap, we can find examples from adjacent branches with large activity changes. This can be shown using p38α as an example.

The ChemTreeMap for 5139 inhibitors of p38α is given in Figure A-2(a). Sub-tree III contains three branches: high activity, low activity, and mixed activity. Six representative molecules are A-F that differ in the positions marked by the red and blue circles. Very little change in activity can be seen for molecules A-D, where a piperazine group in A is replaced by a wide variety of functional groups, see red circles in Figure A-2(b). By contrast, the difference between D-F shows the location of one chlorine atom causes significantly different activity (substitutions off of aromatic rings often cause large changes in bioactivity and chemical reactivity). ChemTreeMap's layout places more similar molecules closer together which better identifies the specific structural features that lead to significant drops in affinity.

Figure A-3(a) shows that SPT produces roughly the same organization of molecules compared to ChemTreeMap. In SPT, branches for A-F are flipped to read as F-A, which is trivial. Molecules D-F are separated from A-C, and the activity cliff appears to occur between D and F. In Figure A-3 (b), Data Warrior does not show a clean activity transition. In Figure A-3 (c), the added rings on A and B cause Scaffold Hunter to organize them into different scaffolds than C-F. Adding the piperazine in compound A does increase the potency significantly, which is shown by all the methods, Scaffold Hunter included.

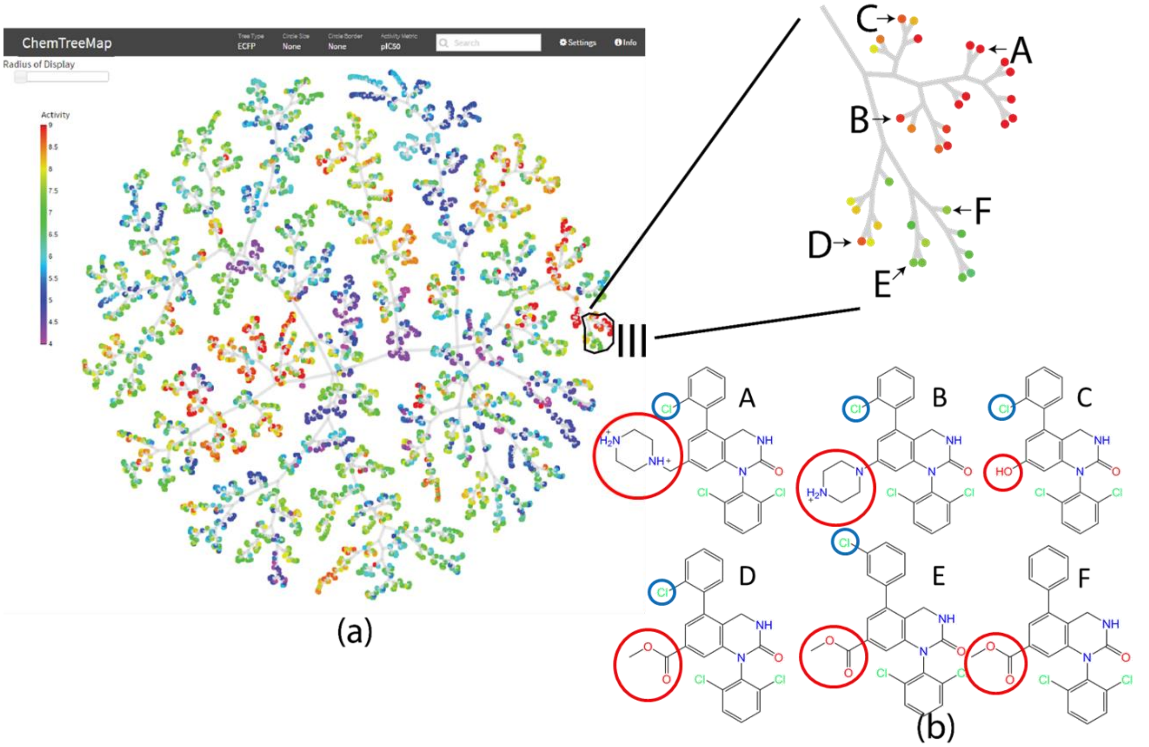


Figure A-2 (a) ChemTreeMap of p38α shows an activity cliff in sub-tree III. (b) Molecules of sub-tree III. Red circles show large chemical modifications in the center from 1-methyl-piperazine to hydroxyl. Blue circles show the positions of the critical chlorine atom.

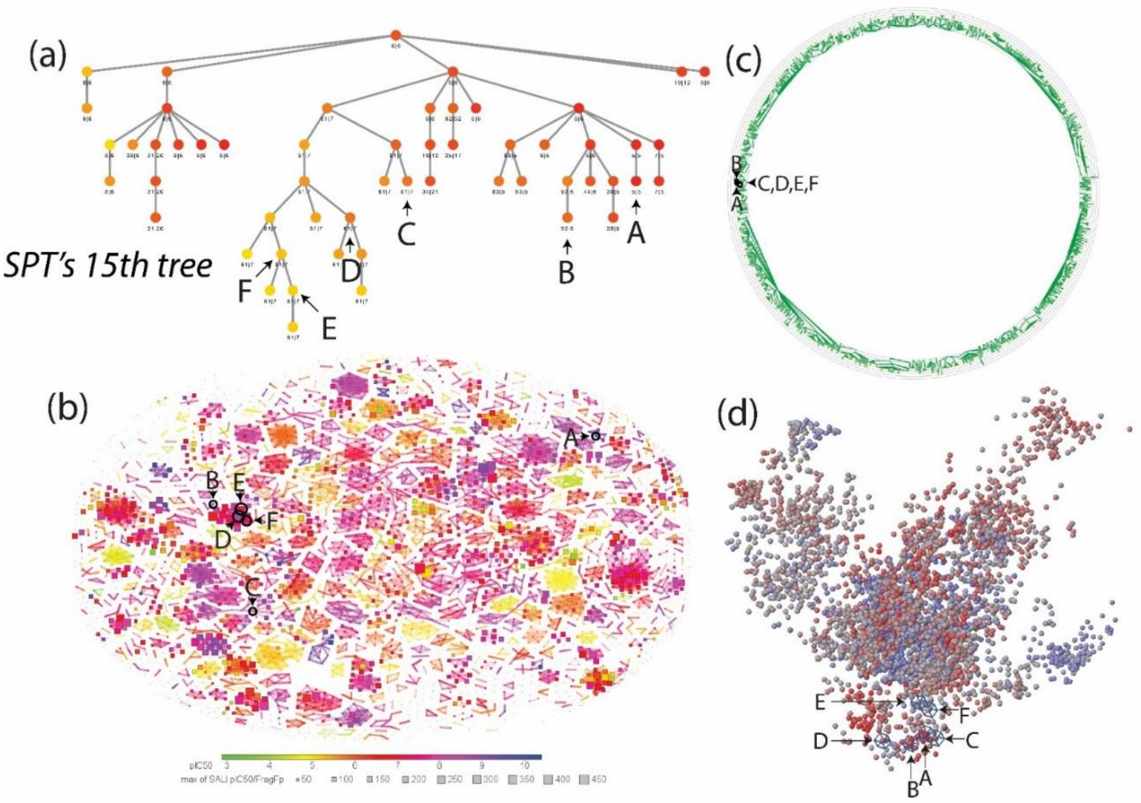


Figure A-3 The visualization of the p38α dataset using (a) SPT, (b) Data Warrior, and (c) Scaffold Hunter.

S4. SAR for local subsets

Our analysis in the paper is based on global exploration of each dataset, but the discussions have focused on local sub-trees. For complete comparisons, the diagrams below show how all four methods perform when given just the compounds of each sub-tree. Occasionally, there are very slight reorganizations, which are expected. Overall, the results are the same.

For ChemTreeMap's NJ algorithm, identifying the two nodes with the smallest $\boldsymbol{Q_{ij}}$ involves information from the entire dataset (step 3 in section 2.2 of the manuscript):

$$\boldsymbol{Q_{ij} = D_{ij} - u_i - u_j}.$$

$\boldsymbol{D_{ij}}$ is simply the T_c between molecule i and j, but $\boldsymbol{u_i}$ is the average distance between node i and all other nodes in the set (same for j). In essence, it finds pairs that are <u>most</u> like one another and <u>least</u> like the rest of the set. By removing some of compounds in the dataset, we change the values of $\boldsymbol{u_i}$.

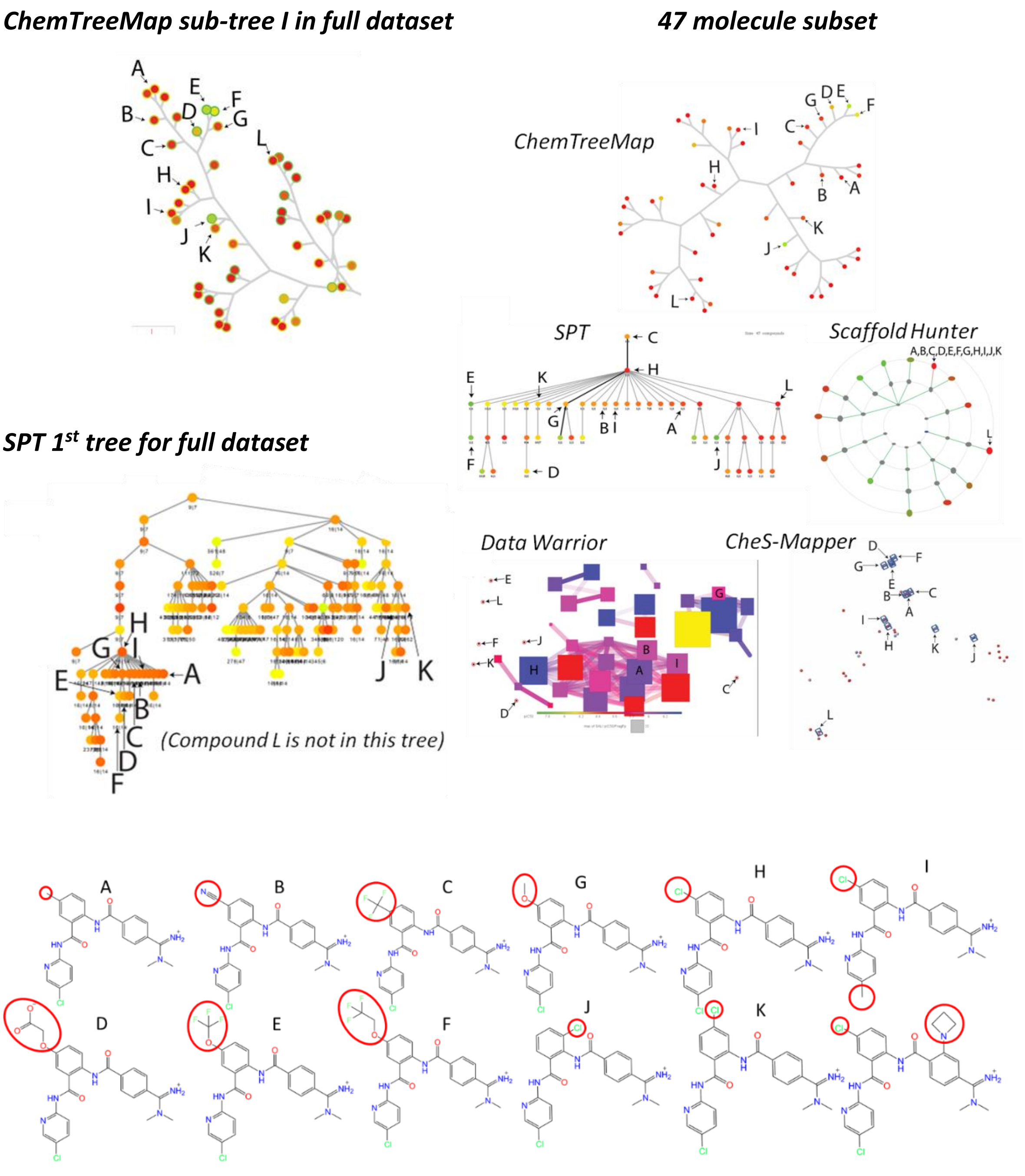
For SPT, **N** trees are still created for **N** molecules, where each is used as the root. The same T_c values exist, so the tiers should be the same, but the scores for the trees can change. Meaning the optimal tree for the local data may have slightly different positions for some molecules vs the global dataset.

For brevity, the discussions below compare ChemTreeMap and SPT. Scaffold Hunter is robust and produces the same classifications as it does with the full datasets. No improvement is seen for Data Warrior; it produces similar graphs with limited connectivity of logical groups of compounds.

FXa sub-tree I (47 molecules):

In ChemTreeMap, the connectivity is the same, except compounds D and G have flipped their order. In the full dataset, ChemTreeMap places E/F/G closest together, grouping all aliphatic esters together which is more appropriate. Compound H is in the middle of subtree I in the full dataset, and as expected, it is the compound closest to the "common ancestor" once the root from the full dataset is removed. Compound L is still in a branch that is well separated from the other compounds.

SPT levels remove some of the structural relationships between molecules but maintain relationships for others (e.g. compounds E and F). Compounds H, J, and K only differ by the location of one chlorine atom, yet they appear unrelated in Figure.2-8(a). Compound L has a unique substitution, but it is placed on the same level as A, B, E, G, I, and K. The colors of the nodes are different for the weaker compounds because the range of green to red is based on max/min of the dataset used (e.g. colors of E and F).



ChemTreeMap sub-tree I in full dataset

47 molecule subset

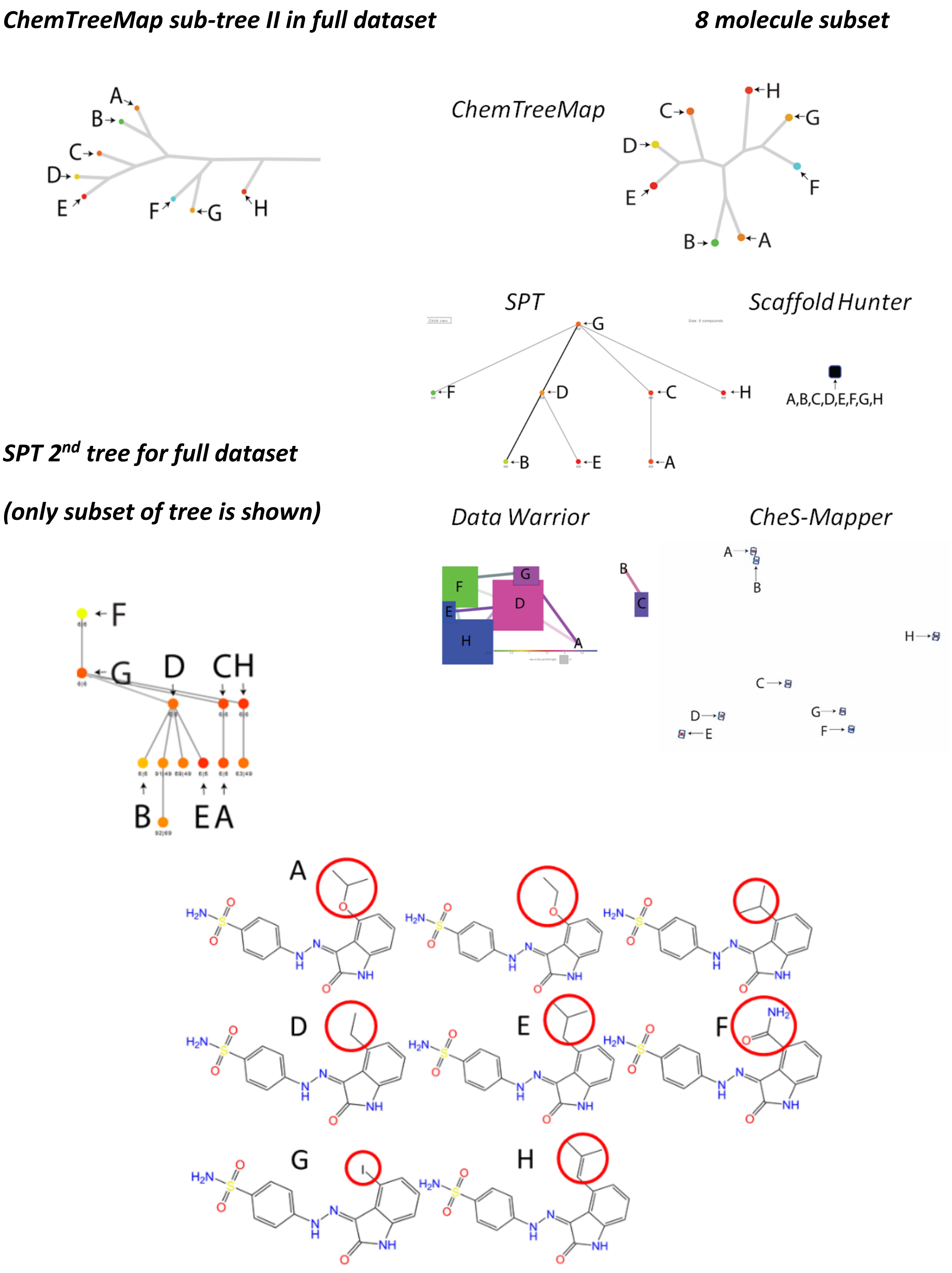
SPT 1st tree for full dataset

Figure A-4 Figure for FXa subtree.

CDK2 sub-tree II (8 molecules):

In ChemTreeMap, the connectivity is the same, except that compound H is closer to A/B instead of F/G being closer. In this instance, grouping H closer to A/B is more appropriate, given the large chemical differences of F and G from the rest of the set. For the full dataset, H is closest the root that connects the rest of the data, and once that restriction is removed, a small reposition of H is possible.

Though drawn slightly differently, the resulting diagram from SPT is the same as the full set. Again, the color of some molecular nodes have changed.



ChemTreeMap sub-tree II in full dataset

8 molecule subset

ChemTreeMap

SPT

Scaffold Hunter

SPT 2nd tree for full dataset

(only subset of tree is shown)

Data Warrior

CheS-Mapper

Figure A-5 Figure for CDK2 subtree.

p38α sub-tree III (34 molecules)

For ChemTreeMap, the connectivity across A-E is the same (A is closest to B, C is next, then D, etc.).

For SPT, using F – the molecule with no chlorine (blue circles) – as a root to connect compounds is a reasonable alternative for organizing the data. This allows branches to be based on the different locations of the added chlorine in compounds E and D. Color changes are seen for this dataset too.

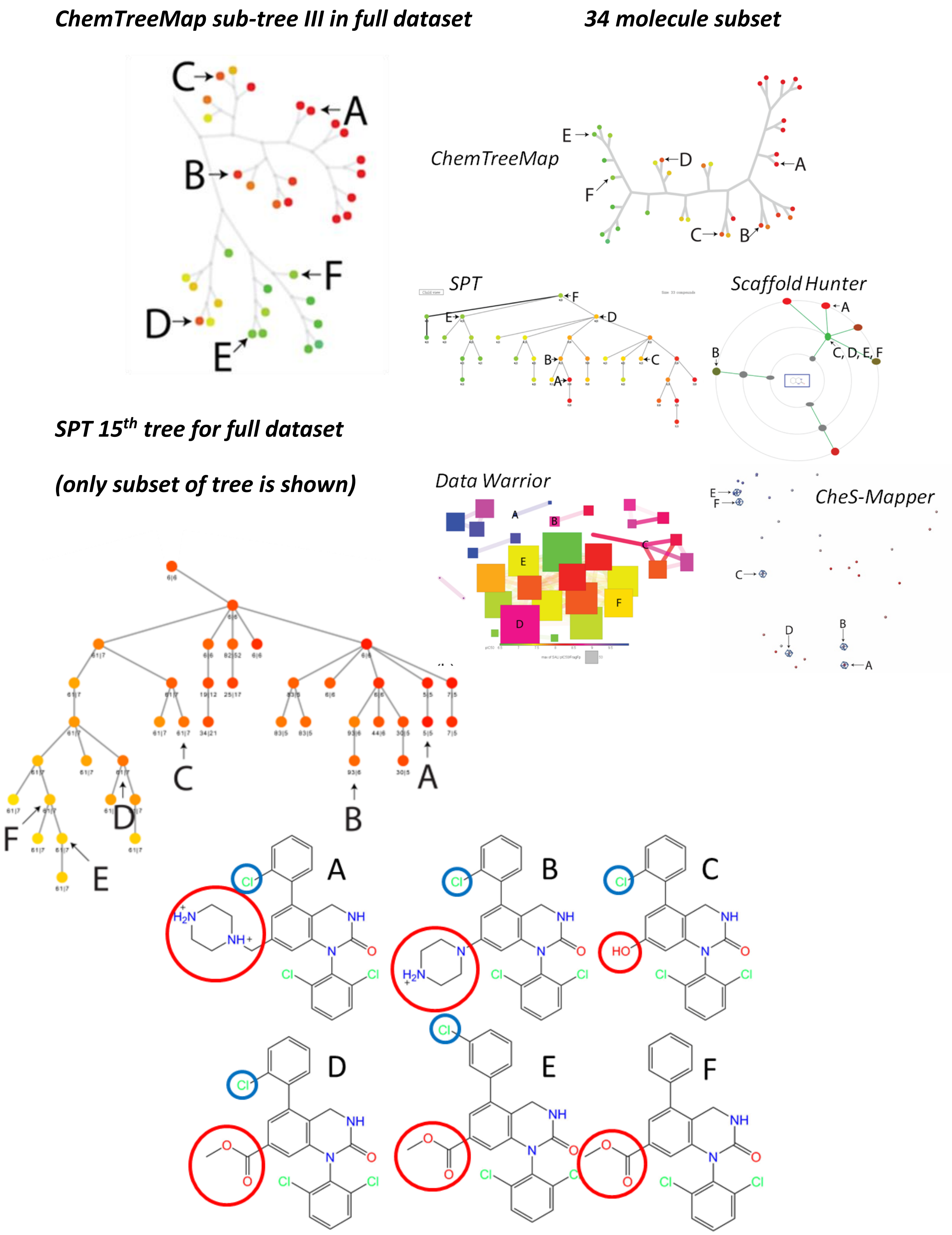


Figure A-6 Figure for p38α subtree.

Appendix B. ChemTreeMap Front End Code

index.html

```
<!doctype html>
<html class="no-js">
<head>
    <meta charset="utf-8">
    <title></title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width">

    <!-- build:css(.) styles/vendor.css -->
    <!-- bower:css -->
    <link rel="stylesheet" href="bower_components/sweetalert/dist/sweetalert.css"
/>
    <link rel="stylesheet"
href="bower_components/sweetalert2/dist/sweetalert2.css" />
    <!-- endbower -->
    <!-- not included in the main list of the components' bower file, but still
required -->
    <link rel="stylesheet" href="bower_components/foundation/css/foundation.css"
/>
    <link rel="stylesheet" href="bower_components/font-awesome/css/font-
awesome.css" />
    <link rel="stylesheet" href="bower_components/jquery-
ui/themes/smoothness/jquery-ui.css" />
    <link rel="stylesheet" href="bower_components/angular-foundation-
colorpicker/css/colorpicker.css" />
    <!-- endbuild -->
    <!-- build:css(.tmp) styles/main.css -->
    <link rel="stylesheet" href="styles/main.css" />
    <link rel="stylesheet" href="styles/chem-tree.css" />
    <link rel="stylesheet" href="styles/tree-slider.css" />
    <link rel="stylesheet" href="styles/tooltip.css" />
    <link rel="stylesheet" href="styles/info.css" />
    <link rel="stylesheet" href="styles/colorbar.css">
    <!-- endbuild -->
</head>

<body ng-app="frontendApp">
<!--[if lt IE 7]>
<p class="browsehappy">You are using an <strong>outdated</strong> browser. Please
<a href="http://browsehappy.com/">upgrade your browser</a> to improve your
experience.</p>
<![endif]-->
```

```
<div id="wrapper">
    <div id="nav" ng-controller="NavController">
        <top-bar is-hover>
            <ul class="left">
                <li class="name">
                    <h1><a href="#/">ChemTreeMap</a></h1>
                </li>
                <li toggle-top-bar class="menu-icon">
                    <a>Menu</a>
                </li>
            </ul>

            <top-bar-section>
                <!-- Right Nav Section -->
                <ul class="right">
                    <li has-dropdown>
                        <a>
                            <div class="dropdown-type">Tree Type</div>
                            <div class="dropdown-
value">{{dataService.current.treeType}}</div>
                        </a>
                        <ul top-bar-dropdown>
                            <li class="{{treeType ===
dataService.current.treeType ? 'active' : ''}}"
                                ng-repeat="treeType in
dataService.available.treeTypes">
                                <a ng-click="dataService.setTreeType(treeType)">
                                    <span class="dropdown-name">{{treeType |
truncate : 12 : '...'}}</span>
                        <span class="dropdown-info" ng-click="getInfo($event,
treeType)">
                          <fa name="info-circle"></fa>
                        </span>
                                </a>
                            </li>
                        </ul>
                    </li>
                    <li has-dropdown>
                        <a>
                            <div class="dropdown-type">Circle Size</div>
                            <div class="dropdown-
value">{{dataService.current.circleSizeType}}</div>
                        </a>
                        <ul top-bar-dropdown>
                            <li class="{{circleSizeType ===
dataService.current.circleSizeType ? 'active' : ''}}"
                                ng-repeat="circleSizeType in
dataService.available.circleSizeTypes">
                                <a ng-
click="dataService.setCircleSizeType(circleSizeType)">
                                    <span class="dropdown-name">{{circleSizeType |
truncate : 12 : '...'}}</span>
                                    <span class="dropdown-info" ng-
click="getInfo($event, circleSizeType)"><fa name="info-circle"></fa></span>
                                </a>
                            </li>
                        </ul>
                    </li>
                    <li has-dropdown>
                        <a>
                        <a>
                            <div class="dropdown-type">Circle Border</div>
                            <div class="dropdown-
value">{{dataService.current.circleBorderType}}</div>
                        </a>
                        <ul top-bar-dropdown>
                            <li class="{{circleBorderType ===
dataService.current.circleBorderType ? 'active' : ''}}"
                                ng-repeat="circleBorderType in
dataService.available.circleBorderTypes">
                                <a ng-
click="dataService.setCircleBorderType(circleBorderType)">
                                    <span class="dropdown-name">{{circleBorderType
| truncate : 12 : '...'}}</span>
                                    <span class="dropdown-info" ng-
click="getInfo($event, circleBorderType)"><fa name="info-circle"></fa></span>
                                </a>
                            </li>
                        </ul>
                    </li>
                    <li has-dropdown>
                        <a>
                            <div class="dropdown-type">Activity Metric</div>
                            <div class="dropdown-
value">{{dataService.current.activityType | truncate : 12 : '...'}}</div>
                        </a>
                        <ul top-bar-dropdown>
                            <li class="{{activityType ===
dataService.current.activityType ? 'active' : ''}}"
                                ng-repeat="activityType in
dataService.available.activityTypes">
                                <a ng-
click="dataService.setActivityType(activityType)">
                                    <span class="dropdown-
name">{{activityType}}</span><span class="dropdown-info" ng-click="getInfo($event,
activityType)"><fa name="info-circle"></fa></span>
                                </a>
                            </li>
                        </ul>
                    </li>
                    <li class="has-form">
                        <fa name="search" class="magnifying-glass"></fa>
                        <script type="text/ng-template" id="customTemplate.html">
                            <a class="typeahead-item">
                                <img ng-src="images/{{match.model.orig_id}}.svg"
width="32">
                                <div class="typeahead-label" ng-bind-
html="'BindingDB:' + match.label | truncate:20:'...' |
typeaheadHighlight:query"></div>
                                <div class="typeahead-id" ng-bind-html="'PubChem:'
+ match.model.PubChem"></div>
                            </a>
                        </script>
                        <input class="dream-search" type="text"
```

```
                               ng-model="currentSearch"
                               typeahead='compound as compound.orig_id for
compound in dataService.data.compounds | filter:$viewValue | limitTo:8'
                               placeholder="Search"
                               typeahead-on-select="select($item)"
                               typeahead-loading="loading"
                               typeahead-template-url="customTemplate.html">
                        <i class="fa fa-loading" ng-show="loading"></i>
                    </li>
                    <li>
                        <a ng-click="openSettings()">
                            <fa name="cog"></fa>
                            <span class="icon-label">Settings</span>
                        </a>
                    </li>
                    <li>
                        <a ng-click="openInfo()">
                            <fa name="info-circle"></fa>
                            <span class="icon-label">Info</span>
                        </a>
                    </li>
                </ul>
            </top-bar-section>
        </top-bar>
    </div>

    <div id="content" ng-view=""></div>

</div>
<script src="bower_components/modernizr/modernizr.js"></script>
<script src="bower_components/jquery/dist/jquery.js"></script>
<script src="bower_components/es5-shim/es5-shim.js"></script>
<script src="bower_components/angular/angular.js"></script>
<script src="bower_components/json3/lib/json3.js"></script>
<script src="bower_components/angular-animate/angular-animate.js"></script>
<script src="bower_components/angular-cookies/angular-cookies.js"></script>
<script src="bower_components/angular-resource/angular-resource.js"></script>
<script src="bower_components/angular-route/angular-route.js"></script>
<script src="bower_components/angular-sanitize/angular-sanitize.js"></script>
<script src="bower_components/angular-touch/angular-touch.js"></script>
<script src="bower_components/waypoints/waypoints.js"></script>
<script src="bower_components/SHA-1/sha1.js"></script>
<script src="bower_components/angulartics/src/angulartics.js"></script>
<script src="bower_components/angulartics/src/angulartics-adobe.js"></script>
<script src="bower_components/angulartics/src/angulartics-chartbeat.js"></script>
<script src="bower_components/angulartics/src/angulartics-cnzz.js"></script>
<script src="bower_components/angulartics/src/angulartics-flurry.js"></script>
<script src="bower_components/angulartics/src/angulartics-ga-cordova.js"></script>
<script src="bower_components/angulartics/src/angulartics-ga.js"></script>
<script src="bower_components/angulartics/src/angulartics-gtm.js"></script>
<script src="bower_components/angulartics/src/angulartics-
kissmetrics.js"></script>
<script src="bower_components/angulartics/src/angulartics-mixpanel.js"></script>
<script src="bower_components/angulartics/src/angulartics-piwik.js"></script>
<script src="bower_components/angulartics/src/angulartics-scroll.js"></script>
<script src="bower_components/angulartics/src/angulartics-segmentio.js"></script>
<script src="bower_components/angulartics/src/angulartics-splunk.js"></script>
<script src="bower_components/angulartics/src/angulartics-woopra.js"></script>
<script src="bower_components/angulartics/src/angulartics-marketo.js"></script>
```

```
<!-- build:js({.tmp,app}) scripts/scripts.js -->
<script src="scripts/app.js"></script>
<script src="scripts/controllers/nav.js"></script>
<script src="scripts/directives/chem-tree.js"></script>
<script src="scripts/services/data-service.js"></script>
<script src="scripts/controllers/tree.js"></script>
<script src="scripts/controllers/settings.js"></script>
<script src="scripts/services/settings.js"></script>
<script src="scripts/directives/tooltip.js"></script>
<script src="scripts/directives/tree-slider.js"></script>
<script src="scripts/controllers/info.js"></script>
<!-- endbuild -->
</body>
</html>
```

info.js

```
'use strict';

/**
 * @ngdoc function
 * @name frontendApp.controller:InfoCtrl
 * @description
 * # InfoCtrl
 * Controller of the frontendApp
 */
angular.module('frontendApp')
  .controller('InfoCtrl', function ($scope, $modalInstance) {

    $scope.dismiss = function () {
        $modalInstance.dismiss('cancel');
    };

});
```

info.html

```
<h1>ChemTreeMap</h1>

<p class="copyright"><fa name="copyright"></fa> Jing Lu 2015</p>

<p class="quote">A tree visualization for compounds</p>

<p>It was developed by Jing Lu, a PhD student in the Carlson Group at the
University of Michigan, Ann Arbor.</p>

<p>Source code is <a href="https://github.com/ajing/ChemTreeMap"
target="_blank">available</a>.</p>

<a class="close-reveal-modal dismiss" ng-click="dismiss()">&#215;</a>
```

settings.html

```html
<h1>Settings</h1>
<form name="myForm">
    <h4>Activate the force directed graph.</h4>
    <p>* Please note the dynamic graph can be slow for a data set with more
than 2,000 molecules.</p>
    <label>
        <input type="radio" ng-model="forceAct.value" ng-value="false">
        Deactivate
    </label>
    <label>
        <input type="radio" ng-model="forceAct.value" ng-value="true">
        Activate
    </label>
</form>


<button ng-click="reset()">Reset</button>
<a class="close-reveal-modal dismiss" ng-click="dismiss()">&#215;</a>
```

tooltip.html

```
<div class="tooltip-close dismiss" ng-click="close()"><i class="fa fa-
close"></i></div>
<h2 class="tooltip-title">{{data.object.orig_id}}</h2>
<a target="_blank" ng-repeat="ext in data.external" ng-href="{{ext.link +
data.object[ext.name]}}"> {{ext.name}} </a>

<!-- compound specific properties -->
<accordion ng-show="data.compound" close-others="oneAtATime">
    <accordion-group is-open="structureIsOpen" ng-init="structureIsOpen =
true">
        <accordion-heading>
            <span>structure</span>
            <i class="tooltip-accordion-toggle" ng-class="{'fa fa-chevron-
down': structureIsOpen, 'fa fa-chevron-right': !structureIsOpen}"></i>
        </accordion-heading>
        <img class="tooltip-structure" ng-
src="images/{{data.object.orig_id}}.svg">
    </accordion-group>

    <accordion-group is-open="activityIsOpen" ng-init="activityIsOpen =
false">
        <accordion-heading>
            <span>activities</span>
            <i class="tooltip-accordion-toggle" ng-class="{'fa fa-chevron-
down': activityIsOpen, 'fa fa-chevron-
right': !activityIsOpen}"></i></accordion-heading>
        <table class="tooltip-table">
            <tr ng-repeat="(activity, value) in data.object.activities">
                <td class="data-key">{{activity}}</td>
                <td class="data-value">{{value | number:4}}</td>
            </tr>
        </table>
    </accordion-group>

    <accordion-group ng-show="true" is-open="propertiesIsOpen" ng-
init="propertiesIsOpen = false">
        <accordion-heading>
            <span>properties</span>
            <i class="tooltip-accordion-toggle" ng-class="{'fa fa-chevron-
down': propertiesIsOpen, 'fa fa-chevron-right': !propertiesIsOpen}"></i>
        </accordion-heading>
        <table class="tooltip-table">
           <tr ng-repeat="(property, value) in data.object.properties">
                <td class="data-key">{{property}}</td>
                <td ng-show="angular.isNumber(value)" class="data-
value">{{value | number:4}}</td>
                <td ng-hide="angular.isNumber(value)" class="data-
value">{{value}}</td>
            </tr>
        </table>
    </accordion-group>
</accordion>
```

tree.html

```
<tree-slider    min="gravitySlider.min"
                max="gravitySlider.max"
                value="gravitySlider.value"
                left-name="gravitySlider.leftName"
                name = "gravitySlider.name"
                id = "gravitySlider.id"
                right-name="gravitySlider.rightName"></tree-slider>

<tooltips       class="tooltip-detail"
                visibility="tooltip.visibility"
                data="tooltip.data"
                click="select"></tooltips>

<chem-tree      data="data"
                data-selected="model.selected"
                force-act="settings.forceAct"
                tree-type="current.treeType"
                circle-size-type="current.circleSizeType"
                circle-border-type="current.circleBorderType"
                activity-type="current.activityType"
                gravity-value="gravitySlider.value"
                link-strength-value="linkStrengthSlider.value"></chem-tree>
```

chem-tree.css

```
svg.viz {
    position: absolute;
    top: 45px;
    width: 100%;
    height: calc(100% - 45px);
}

circle.node {
    opacity: 1;
    transition: fill 0.2s ease-in-out;
    -webkit-transition: fill 0.2s ease-in-out;
    -moz-transition: fill 0.2s ease-in-out;
    -o-transition: fill 0.2s ease-in-out;
    -ms-transition: fill 0.2s ease-in-out;
}

circle.highlighted {
    stroke: #999999;
    fill: #666666;
}

circle.node.selected {
    stroke: #ca0a00;
    fill: #ca0a00;
}

circle.hidden {
    display: none;
    visibility: none;
    opacity: 0.0;
}

line.link {
    stroke-opacity: 0.6;
    stroke: #D3D3D3;
    stroke-width: 5px;
    transition: stroke-opacity 0.2s ease-in-out;
    -webkit-transition: stroke-opacity 0.2s ease-in-out;
    -moz-transition: stroke-opacity 0.2s ease-in-out;
    -o-transition: stroke-opacity 0.2s ease-in-out;
    -ms-transition: stroke-opacity 0.2s ease-in-out;
}
```

colorbar.css

```
.colorbarA .axis text {
    font-family: sans-serif;
    font-size: 11px;
}

.colorbarB .axis text {
    font-family: sans-serif;
    font-size: 11px;
}

.colorbarA .axis line{
    fill: none;
    /*
    stroke: black;
    */
    shape-rendering: crispEdges;
}

.colorbarA .axis .domain{
    fill: none;
    stroke: black;
    shape-rendering: crispEdges;
}

.colorbarB .axis line{
    fill: none;
    /*
    stroke: black;
    */
    shape-rendering: crispEdges;
}

.colorbarB .axis .domain{
    fill: none;
    stroke: black;
    shape-rendering: crispEdges;
}
```

info.css

```
.copyright {
    opacity: 0.7;
}

.quote {
  font-style: italic;
  font-size: 18px;
  background: #f9f9f9;
  border-left: 10px solid #ccc;
  margin: 1.5em 10px;
  padding: 0.5em 10px;
  quotes: "\201C""\201D""\2018""\2019";
}
```

main.css

```
/* Sorry, should have used sass!!! First time using css :(*/

@import
url(http://fonts.googleapis.com/css?family=Source+Sans+Pro:200,300,400,600,
700,900,200italic,300italic,400italic,600italic,700italic,900italic);

html, body, div, span, button, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed,
figure, figcaption, footer, header, hgroup,
menu, nav, output, ruby, section, summary,
time, mark, audio, video {
  font-family: 'Source Sans Pro', sans-serif;

}

.top-bar .name h1 {
  font-size: 24px;
}

.top-bar-section ul li > a {
  font-family: 'Source Sans Pro';
}

#wrapper:before {
    content:'';
    float: left;
    height: 100%;
}
#wrapper {
    height: 100%;

}

#content {

}

.has-form {
    padding: 5px 0 5px 0;
}

.has-form ul li {
  float: none;
  color: black;
  background: none;
}

/* nav bar */
```

```
.top-bar, .top-bar-section li:not(.has-form) a:not(.button), .top-bar-
section .has-form, .top-bar-section .dropdown li:not(.has-
form):not(.active) > a:not(.button) {
  background: #444;
  }

.top-bar-section .dropdown li:not(.has-form) a:not(.button):hover {
  background: #363636;
}

.top-bar-section .dropdown li.active:not(.has-form) a:not(.button) {
  background: #008CBA;
}

.top-bar-section .dropdown li.active:not(.has-form) a:not(.button):hover {
  background: #007BA9;
}


.dropdown-type {
  opacity: 0.75;
  font-weight: lighter;
  padding: .85em 0 0.5em 0;
  line-height: 0.8125em;
  font-size: 0.8125em;
}

.dropdown-info {
  position: absolute;
  z-index: 5;
  opacity: 0.6;
  right: 13px;
}

.dropdown-info:hover {
  opacity: 1.0;
}

.dropdown-value {
  line-height: 1.0em;
  padding-bottom: 0.75em;
}

/* typeahead list */

.has-form li:not(.has-form) a:not(.button), .has-form .dropdown {
  color: black;
  background: none;
}

.typeahead-id {
  position:absolute;
  left: 55px;
  bottom: 0.75em;
  line-height: 0.8123em;
  font-weight: lighter;
  font-size: 0.8123em;
  opacity: 0.75;
```

```
}

.typeahead-label {
  position: absolute;
  line-height: 1.0em;
  font-size: 1.0em;
  left: 55px;
  top: 0.75em;

}

.typeahead-item {
  position: relative;
}

ul[typeahead-popup] {
  width: 12em;
}

/* active typeahead*/

/*

.has-form li.active:not(.has-form) a:not(.button), .has-form li:not(.has-form):hover a:not(.button):hover, .has-form .dropdown {
  color: white;
  background: blue;
}

*/
.cp {
  padding: 1em 1em 1em 1em;
}


.top-bar {
    z-index: 100;
}

/* settings modal */

table.color-settings tbody tr {
  background: none;
}

.dismiss {
    opacity: 0.5;
    transition: opacity 0.25s;
    -webkit-transition: opacity 0.25s;
    -moz-transition: opacity 0.25s;
    -o-transition: opacity 0.25s;
}

.dismiss:hover {
    opacity: 1.0;
}

.magnifying-glass {
  position: absolute;
  z-index:2;
  margin-left: 5px;
  margin-top: 15px;
  color: #CCC;
}

input[type="text"].dream-search {
  width: 12em;
  padding-left: 2em;
}

ul[typeahead-popup] {
  z-index: 30;
}


@media only screen and (max-width: 40.063em) {

    .magnifying-glass {

      margin-top: 6px;

    }

    input[type="text"].dream-search {
      width: 100%;

    }

    input[type="text"].dream-search:focus {
      width: 100%;

    }
}

label {
    color: black;
}

.fade {
  opacity: 0;
  -webkit-transition: opacity .15s linear;
          transition: opacity .15s linear;
}
.fade.in {
  opacity: 1;
}

.reveal-modal.fade {
  -webkit-transition: -webkit-transform .3s ease-out;
     -moz-transition:    -moz-transform .3s ease-out;
       -o-transition:      -o-transform .3s ease-out;
          transition:         transform .3s ease-out;
  -webkit-transform: translate(0, -25%);
      -ms-transform: translate(0, -25%);
          transform: translate(0, -25%);
}
```

```
.reveal-modal.in {
  -webkit-transform: translate(0, 0);
      -ms-transform: translate(0, 0);
          transform: translate(0, 0);
}

.reveal-modal-bg.fade {
  filter: alpha(opacity=0);
  opacity: 0;
}
.reveal-modal-bg.in {
  filter: alpha(opacity=50);
  opacity: .5;
}

@media only screen and (min-width: 40.063em) and (max-width: 66em) {

  .top-bar-section .has-dropdown > a {
    padding-right: 15px !important;
  }
  .top-bar-section .has-dropdown > a::after {
    display: none;
  }
}

@media only screen and (min-width: 40.063em) and (max-width: 61em) {
  .icon-label {
    display: none;
  }
}

@media only screen and (min-width: 40.063em) and (max-width: 56em) {
  input[type="text"].dream-search {
    width: 7em;
  }
}

@media only screen and (min-width: 40.063em) and (max-width: 52em) {
  .top-bar .name h1 a {
    padding: 0 0 0 0;
  }
}

@media only screen and (min-width: 40.063em) and (max-width: 50em) {
  .top-bar .name {
    display: none;
  }
}


.top-bar .toggle-topbar.menu-icon {
  top: 22px;
}

.top-bar .toggle-topbar a {
  -webkit-touch-callout: none;
  -webkit-user-select: none;
  -khtml-user-select: none;
  -moz-user-select: none;
  -ms-user-select: none;
  user-select: none;
}

.top-bar.expanded .toggle-topbar a {
  color: #FFF;
}
.top-bar .name h1 a {
  width: 100%;
}
nav.top-bar.expanded {
  background: #393939;
}

.main-section {
  text-align: center;
  padding-top: 60px;
  width: 50%;
  margin: auto;
}

@media only screen and (max-width: 600px) {
  .main-section {
    width: 300px;
  }
}

@media only screen and (max-width: 40.063em) {
  .main-section {
    padding-top: 20px;
  }
}

.main-section ul {
  border-radius: 10px;
  list-style-type: none;
  margin: 20px 0 20px 0;
  padding: 0;
  background: #EEE;
}

.main-section li:first-child {
  border-top-left-radius: 10px;
  border-top-right-radius: 10px;
}

.main-section li:last-child {
  border: none;
  border-bottom-left-radius: 10px;
  border-bottom-right-radius: 10px;
}

.main-section li {

  cursor: default;
  -webkit-user-select: none; /* Chrome/Safari */
  -moz-user-select: none; /* Firefox */
  -ms-user-select: none; /* IE10+ */

  /* Rules below not implemented in browsers yet */
  -o-user-select: none;
  user-select: none;
  text-decoration: none;
  color: #000;
  display: block;
  text-align: center;
  font: 200 20px/1.5 Helvetica, Verdana, sans-serif;
  border-bottom: 1px solid #CCC;

  -webkit-transition: font-size 0.3s ease, background-color 0.3s ease;
  -moz-transition: font-size 0.3s ease, background-color 0.3s ease;
  -o-transition: font-size 0.3s ease, background-color 0.3s ease;
  -ms-transition: font-size 0.3s ease, background-color 0.3s ease;
  transition: font-size 0.3s ease, background-color 0.3s ease;
}

.main-section li a {
  color: black;
  display:block;
}

.main-section li:hover {
  background: #F6F6F6;
}

.main-section li.current {
  background: #EBFFE0;
}
```

tooltip.css

```
.tooltip-detail {
    display: none;
    max-height: calc(100% - 110px);
    overflow: auto;
    position: absolute;
    opacity: 0.9;
    background: white;
    user-select:none;
    top: 80px;
    right: 20px;
    width: 280px;
    z-index: 10;
    -webkit-box-shadow: rgba(0, 0, 0, 0.188235) 0px 10px 30px 0px;
    -webkit-font-smoothing: antialiased;
    border-bottom-left-radius: 2px;
    border-bottom-right-radius: 2px;
    border-top-left-radius: 2px;
    border-top-right-radius: 2px;
    box-shadow: rgba(0, 0, 0, 0.188235) 0px 10px 30px 0px;
    box-sizing: border-box;

}

.tooltip-accordion-toggle {
    opacity: 0.4;
    float:right;
    vertical-align: middle;
    line-height: 1.5em;
}

.tooltip-title {
    padding: 0.25em 0.5em 0.0em 0.5em;
    font-size: 20pt;
    font-weight: 600;
}

.tooltip-table {
    margin: 0 5% 0 5%;
    width: 90%;
    border: none;
    border-spacing: 0;
}

.tooltip-table tr:nth-child(even) {
    background-color: white;
}

.synergy-table tr td {
    cursor: pointer;
    color: #666666;
    max-width: 120px;
    transition: color 0.25s;
    -webkit-transition: color 0.25s;
    -moz-transition: color 0.25s;
    -o-transition: color 0.25s;
    transition: background-color 0.1s;
    -webkit-transition: background-color 0.1s;
    -moz-transition: background-color 0.1s;
    -o-transition: background-color 0.1s;
}

.compound-image-thumbnail {
    padding: 0.25em;
}

.accordion dd > .content {
    padding: 0.4em;
}

.synergy-table tr:hover td {
    color: #111111;
    background-color: #F9F9F9;
}

.tooltip-table td.data-key {
    text-align: left;
    color: #333333;
    font-weight: bold;
    border-left: gray;

}

.tooltip-table td.data-value {
    text-align: right;
    color: #666666;

}

.accordion dd a {
    padding: 0.5em 1em 0.5em 1em;
    font-weight: lighter;
}

.accordion dd div.content {

}

.tooltip-structure {
    margin: -5px auto -5px auto;
    display: block;
    width: 10em;
    height: 10em;
}

.tooltip-components-container {
    text-align: center;
}

.tooltip-component-container {
    color: #888888;
    opacity: 0.6;
    cursor: pointer;
    display: inline-block;
    width: 120px;
    transition: opacity 0.25s;
    -webkit-transition: opacity 0.25s;
    -moz-transition: opacity 0.25s;
    -o-transition: opacity 0.25s;
}

.tooltip-component-container:hover {
    color: #111111;
    opacity: 1.0;
}

.tooltip-component {
    display: block;
    margin-left: auto;
    margin-right: auto;
    width: 6em;
    height: 6em;
}

.tooltip-close {
    font-size: 14pt;
    position: absolute;
    top: 0.4em;
    right: 0.8em;
    color: #999999;
}
```

tree-slider.css

```
.slider {
    position: relative;
    z-index: 1;
    width: 150px;
    height: 17px;
    top: 0px;
    left: 20px;
    opacity: 0.8;
}

.ui-slider .ui-slider-handle {
    margin-top: 2px;
}

.ui-slider-range {
    background: white;
}

.slider .section {
    display: inline-block;
    position: absolute;
    height: 100%;
    font-size: 10px;
}

.slider .left {
    border-bottom-left-radius: 3px;
    border-top-left-radius: 3px;
    padding-left: 2.5%;
    text-align: left;
    float:left;
}

.slider .right {
    border-bottom-right-radius: 3px;
    border-top-right-radius: 3px;
    padding-right: 2.5%;
    text-align: right;
    right: 0px;
}

.sm-label {
    opacity: 0.0;
    cursor: default;

    user-select: none;
    -o-user-select: none;
    -webkit-user-select: none; /* Chrome/Safari */
    -moz-user-select: none; /* Firefox */
    -ms-user-select: none; /* IE10+ */

    transition: all 0.2s ease-in-out;
    -webkit-transition: all 0.2s ease-in-out;
    -moz-transition: all 0.2s ease-in-out;
    -o-transition: all 0.2s ease-in-out;
    -ms-transition: all 0.2s ease-in-out;
}

.slider .section:hover .sm-label {
    opacity: 1.0;
}



svg.axis {
    position: absolute;
    left: 0px;
    top: 17px;
    width: 280px;
    height: 20px;
    margin: 0 auto;
    margin-top: -1px;
    opacity: 0.3;
}

svg.axis line {
  stroke: #000;
}

svg.axis path {
  display: none;
}

svg.axis .tick {
    font-size: 10px;
}
```

app.js

```
'use strict';


/**
 * @ngdoc overview
 * @name frontendApp
 * @description
 * # frontendApp
 *
 * Main module of the application.
 */

angular
  .module('frontendApp', [
    'oitozero.ngSweetAlert',
    'angular.filter',
    'colorpicker.module',
    'picardy.fontawesome',
    'mm.foundation',
    'angulartics',
    'angulartics.google.analytics',
    'ngAnimate',
    'ngCookies',
    'ngResource',
    'ngRoute',
    'ngSanitize',
    'ngTouch'
  ])
  .config(function ($routeProvider) {
    $routeProvider
      .when('/:dataset', {
          templateUrl: 'views/tree.html',
          controller: 'TreeController'
      })
      .otherwise({
        redirectTo: '/aff'
      });
  });
```

info.js

```
'use strict';


/**
 * @ngdoc overview
 * @name frontendApp
 * @description
 * # frontendApp
 *
 * Main module of the application.
 */

angular
  .module('frontendApp', [
    'oitozero.ngSweetAlert',
    'angular.filter',
    'colorpicker.module',
    'picardy.fontawesome',
    'mm.foundation',
    'angulartics',
    'angulartics.google.analytics',
    'ngAnimate',
    'ngCookies',
    'ngResource',
    'ngRoute',
    'ngSanitize',
    'ngTouch'
  ])
  .config(function ($routeProvider) {
    $routeProvider
      .when('/:dataset', {
          templateUrl: 'views/tree.html',
          controller: 'TreeController'
      })
      .otherwise({
        redirectTo: '/aff'
      });
  });
```

nav.js

```
/**
 * Created by ajing on 9/10/15.
 */


'use strict';

/**
 * @ngdoc function
 * @name frontendApp.controller:NavController
 * @description
 * # NavController
 * Controller of the frontendApp
 */
angular.module('frontendApp')
  .controller('NavController', function ($scope, $modal, dataService,
SweetAlert) {

    $scope.dataService = dataService;

    $scope.currentSearch = '';


    $scope.getInfo = function(e, infoObject) {
      e.stopPropagation();
      //console.log(dataService.metadata);
      SweetAlert.swal({
        title: infoObject,
        html: dataService.metadata[infoObject],
        allowOutsideClick: true
      });
    };

    $scope.select = function(a) {
      dataService.model.selected = a;
      console.log(dataService.model.selected);
    };

    //open the settings modal
    $scope.openSettings = function () {

      $modal.open({
          templateUrl: 'views/settings.html',
          controller: 'SettingsCtrl'
        }
      );
    };

    //open the info modal
    $scope.openInfo = function () {

      $modal.open({
          templateUrl: 'views/info.html',
          controller: 'InfoCtrl'
        }
      );
```

```
   };
});
```

settings.js

```
'use strict';

/**
 * @ngdoc function
 * @name frontendApp.controller:SettingsCtrl
 * @description
 * # SettingsCtrl
 * Controller of the frontendApp
 * Used to control the settings page
 */

angular.module('frontendApp')
  .controller('SettingsCtrl', function ($scope, $modalInstance, settings) {


    //dataService.current.forceAct = {value: false};
    $scope.forceAct = settings.forceAct;

    $scope.reset = function () {
      $scope.forceAct.value = settings.defaultForce.value;
    };

    $scope.dismiss = function () {
      $modalInstance.dismiss('cancel');
    };
  });
```

tree.js

```
'use strict';

/**
 * @ngdoc function
 * @name frontendApp.controller:TreeController
 * @description
 * # TreeController
 * Controller of the frontendApp
 */
angular.module('frontendApp')
  .controller('TreeController', function ($scope, $routeParams, 
dataService, settings) {

    //locate database from the route
    $scope.datasetName = $routeParams.dataset;

    //selected information is currently nothing
    $scope.model = dataService.model;

    $scope.current = dataService.current;

    $scope.flatten = dataService.flatten;

    $scope.tooltip = {visibility: false};

    $scope.settings = settings;

    $scope.$watch('model.selected', function(selected) {

      if (selected === null) {
        $scope.tooltip.visibility = false;
      } else {
        //set up the tooltip for the specific selected item
        $scope.tooltip.visibility = true;
        $scope.tooltip.data = {
          compound: true,
          object: selected,
          external: dataService.data.metadata.external
        };
      }
    });

    $scope.$watch('tooltip.visibility', function(newVis) {
      if ( newVis === false ) {
        $scope.model.selected = null;
      }
    });

    $scope.select = function(d) {
      $scope.model.selected = d;
    };



    $scope.gravitySlider      = {min: 0, max: 0.2, value:0.1, id: 
'gravity', name: 'Radius of Display', leftName: 'Lager', rightName:
```

```
'Smaller'};

    $scope.linkStrengthSlider = {min: 0, max: 10, value: 1, id:
'linkstrength', name: 'Compactness', leftName: 'Looser', rightName:
'Tight'};

    dataService.loadExample($routeParams.dataset, function() {
        $scope.data = dataService.data;
/*        // i deleted all slider parameters, but I need to add color bar
parameters here.
        $scope.$watch('dataService.current.circleBorderType', function() {
            var extent = d3.extent($scope.data.nodes, function(d) { return
d.strock; });
            $scope.borderColorbar.min = extent[0];
            $scope.borderColorbar.max = extent[1];
        });*/

    });
});
```

chem-tree.js

```
'use strict';

/**
 * @ngdoc directive
 * @name frontendApp.directive:chem-tree
 * @description
 * # chem-tree
 */

function colorBar(){
  var orient = 'right',
    lineWidth = 40,
    size_ = 300,
    tickFormat = d3.format('3e'),
    color = d3.scale.linear().domain([0, 0.5, 1]).range(['blue', 'green',
'red']), //v -> color
    line = d3.svg.line().interpolate('basis'),
    precision = 8,
    points_,
    tickSize_;

  function component(selection){
    selection.each(function(){
      var container = d3.select(this),
        tickSize = tickSize_ || lineWidth,
        n,
        points = points_ || (((orient === 'left') || (orient ===
'right'))?[[0,size_],[0,0]]:[[size_,0],[0,0]]),
        quads = quad(sample(line(points),precision)),
        size = (points)?n:size_,
        aScale =
color.copy().interpolate(d3.interpolate).domain(color.domain()).range([size
,0]), //v -> px
        colorExtent = color.domain(),
        normScale =
color.copy().domain(color.domain().map(function(d){ return (d -
colorExtent[0])/ (colorExtent[1] - colorExtent[0]);})),

      //Save values for transitions
        oldLineWidth = this.__lineWidth__ || lineWidth,
        oldQuads = this.__quads__ || quads;
      this.__quads__ = quads;
      this.__lineWidth__ = lineWidth;

      //Enters
      var bar = container.selectAll('path.c').data(d3.range(quads.length),
function(d){return d;}),
        bEnter = bar.enter().insert('path','g.axis').classed('c',true),
        bExit = d3.transition(bar.exit()).remove(),
        bUpdate = d3.transition(bar),
        bTransform = function(selection,f,lw){
          selection.style('fill', function(d) { return
normScale(f(d).t); })
            .style('stroke', function(d) { return normScale(f(d).t); })
      .attr('d', function(d) { var p = f(d); return lineJoin(p[0],
p[1], p[2], p[3], lw); });};

   bEnter.call(bTransform,function(d){return oldQuads[oldQuads.length -
1];},oldLineWidth); // enter from last of oldQuad
   bExit.call(bTransform,function(d){return quads[quads.length -
1];},lineWidth); //exit from last of quads
   bUpdate.call(bTransform,function(d){return quads[d];},lineWidth);

   var colorBarAxis = d3.svg.axis().scale(aScale).orient(orient)
     .tickSize(tickSize).tickFormat(tickFormat),
    a = container.selectAll('g.axis').data(function(d){return
(aScale)?[1]:[];}), //axis container
    aEnter = a.enter().append('g').classed('axis',true),
    aExit = d3.transition(a.exit()).remove(),
    aUpdate = d3.transition(a).call(colorBarAxis),
    aTransform = function(selection,lw){
     selection.attr('transform', 'translate(' + (((orient === 'right')
|| (orient === 'left'))?-lw/2:0) + ',' + (((orient === 'right') || (orient
==='left'))?0:lw/2) + ')');};

   aEnter.call(aTransform,oldLineWidth);
   aExit.call(aTransform,lineWidth);
   aUpdate.call(aTransform,lineWidth);

   // Compute stroke outline for segment p12.
   function lineJoin(p0, p1, p2, p3, width) {
    var u12 = perp(p1, p2),
     r = width / 2, e,
     a = [p1[0] + u12[0] * r, p1[1] + u12[1] * r],
     b = [p2[0] + u12[0] * r, p2[1] + u12[1] * r],
     c = [p2[0] - u12[0] * r, p2[1] - u12[1] * r],
     d = [p1[0] - u12[0] * r, p1[1] - u12[1] * r];

    if (p0) { // clip ad and dc using average of u01 and u12
     var u01 = perp(p0, p1);
     e = [p1[0] + u01[0] + u12[0], p1[1] + u01[1] + u12[1]];
     a = lineIntersect(p1, e, a, b);
     d = lineIntersect(p1, e, d, c);
    }

    if (p3) { // clip ab and dc using average of u12 and u23
     var u23 = perp(p2, p3);
     e = [p2[0] + u23[0] + u12[0], p2[1] + u23[1] + u12[1]];
     b = lineIntersect(p2, e, a, b);
     c = lineIntersect(p2, e, d, c);
    }

    return 'M' + a + 'L' + b + ' ' + c + ' ' + d + 'Z';
   }

   // Compute intersection of two infinite lines ab and cd.
   function lineIntersect(a, b, c, d) {
    var x1 = c[0], x3 = a[0], x21 = d[0] - x1, x43 = b[0] - x3,
     y1 = c[1], y3 = a[1], y21 = d[1] - y1, y43 = b[1] - y3,
     ua = (x43 * (y1 - y3) - y43 * (x1 - x3)) / (y43 * x21 - x43 *
y21);
    return [x1 + ua * x21, y1 + ua * y21];
```

```
      }

      // Compute unit vector perpendicular to p01.
      function perp(p0, p1) {
        var u01x = p0[1] - p1[1], u01y = p1[0] - p0[0],
          u01d = Math.sqrt(u01x * u01x + u01y * u01y);
        return [u01x / u01d, u01y / u01d];
      }


      // Sample the SVG path string 'd' uniformly with the specified
precision.
      function sample(d,pre) {
        var path = document.createElementNS(d3.ns.prefix.svg, 'path');
        path.setAttribute('d', d);

        n = path.getTotalLength();

        var t = [0], i = 0;
        while ((i += pre) < n) {
          t.push(i);
        }
        t.push(n);

        return t.map(function(t) {
          var p = path.getPointAtLength(t), a = [p.x, p.y];
          a.t = t / n;
          return a;
        });

      }

      // Compute quads of adjacent points [p0, p1, p2, p3].
      function quad(pts) {
        return d3.range(pts.length - 1).map(function(i) {
          var a = [pts[i - 1], pts[i], pts[i + 1], pts[i + 2]];
          a.t = (pts[i].t + pts[i + 1].t) / 2;
          return a;
        });
      }


    });}

  component.orient = function(_) {
    if (!arguments.length) {
      return orient;
    }
    orient = _;
    return component;
  };

  component.lineWidth = function(_) {
    if (!arguments.length) {
      return lineWidth;
    }
    lineWidth = _;
    return component;
```

```
};

component.size = function(_) {
  if (!arguments.length) {
    return size_;
  }
  size_ = _;
  return component;
};

component.tickFormat = function(_) {
  if (!arguments.length) {
    return tickFormat;
  }
  tickFormat = _;
  return component;
};

component.tickSize = function(_) {
  if (!arguments.length) {
    return tickSize_;
  }
  tickSize_ = _;
  return component;
};

component.color = function(_) {
  if (!arguments.length) {
    return color;
  }
  color = _;
  return component;
};

component.precision = function(_) {
  if (!arguments.length) {
    return precision;
  }
  precision = _;
  return component;
};

component.points = function(_) {
  if (!arguments.length) {
    return points_;
  }
  points_ = _;
  return component;
};

component.line = function(_) {
  if (!arguments.length) {
    return line;
  }
  line = _;
  return component;
};
```

```
  return component;
}

angular.module('frontendApp')
  .directive('chemTree', function () {
    function link($scope, $elements) {
      var xScale, yScale, activityScale, activityColor, slogpScale,
ligeffScale,
        borderScale, borderColor, sizeScale, force, nodes, linkDOM,
nodeDOM;

      //setup
      var el = $elements[0];

      //append the svg element
      var svg = d3.select(el)
        .append('svg')
        .attr({class: 'viz'});

      //clicking anywhere should set selected to none.  This should default
away if clicking on an object
      svg.on('click', function(){
        if (d3.event.defaultPrevented) { return; }
        $scope.selected = null;
        $scope.$apply();
      });

      var vis = svg.append('g'); // the zoom container

      d3.selectAll('.viz').append('g').append('svg')
        .attr('x', '80')
        .attr('y', '100')
        .append('g')
        .attr('transform', 'translate(0, 10)').classed('colorbarA',true);//
color bar Activyty

      var barB = d3.selectAll('.viz').append('g').append('svg')
        .attr('x', '20')
        .attr('y', '100')
        .append('g')
        .attr('transform', 'translate(0, 10)').classed('barB', true);

      barB.append('text').classed('barBtext', true);
      barB.append('g')
        .attr('transform', 'translate(0, 10)').classed('colorbarB',true);//
color bar border

      function flatten(root){
        var nodes = [];

        function recurse(node) {
          if (node.children) { node.size = node.children.reduce(function(p,
v) { return p + recurse(v); }, 0); }
          nodes.push(node);
          return node.size;
        }

        root.size = recurse(root);
        return nodes;
      }

      function tick() {
        linkDOM
          .attr('x1', function(d) { return xScale(d.source.x); })
          .attr('y1', function(d) { return yScale(d.source.y); })
          .attr('x2', function(d) { return xScale(d.target.x); })
          .attr('y2', function(d) { return yScale(d.target.y); });

        nodeDOM
          .attr('cx', function(d) { return xScale(d.x); })
          .attr('cy', function(d) { return yScale(d.y); });
      }

      activityScale = d3.scale.linear()
        .domain([4, 9])
        .clamp(true)
        .range(['hsl(300,80%,50%)', 'hsl(0,80%,50%)'])
        .interpolate(d3.interpolateString);

      slogpScale = d3.scale.linear()
        .domain([5, -5])
        .clamp(true)
        .range(['hsl(300,80%,50%)', 'hsl(0,80%,50%)'])
        .interpolate(d3.interpolateString);


      ligeffScale = d3.scale.linear()
        .domain([0, 0.5])
        .clamp(true)
        .range(['hsl(300,80%,50%)', 'hsl(0,80%,50%)'])
        .interpolate(d3.interpolateString);

      // Activity colorbar
      var colorbarA = colorBar()
        .color(activityScale).size(350).lineWidth(20).precision(4).tickForm
at(d3.format('g'));

      d3.select('.colorbarA')
        .insert('text',':first-child')
        .text('Activity');

      d3.select('.colorbarA')
        .append('g')
        .attr('transform', 'translate(0, 10)').call(colorbarA);


      function changeBorderColorBar(nodes) {
        var colorExtent, colorbarB; // for border color
        if ($scope.circleBorderType === 'SLogP') {
          borderScale = slogpScale;
          colorbarB = colorBar()
            .color(borderScale).size(350).lineWidth(20).precision(4).tickFo
rmat(d3.format('g'));
        } else if ($scope.circleBorderType === 'Lig_Eff') {
          borderScale = ligeffScale;
          colorbarB = colorBar()
```

```
            .color(borderScale).size(350).lineWidth(20).precision(4).tickFo
rmat(d3.format('g'));
        } else {
          colorExtent = d3.extent(nodes, function(d) { if (d.name[0] ===
'B') { return d.stroke; } });
          borderScale = d3.scale.linear()
            .domain([colorExtent[0], d3.mean(colorExtent), colorExtent[1]])
            .range(['#008000', '#FFFF00', '#FF0000']);
        }


        if ($scope.circleBorderType === 'None') {
          d3.select('.barB').style('visibility','hidden');
          d3.select('.colorbarB').style('visibility','hidden');
        } else {
          d3.select('.barB').style('visibility','visible');
          d3.select('.colorbarB').style('visibility','visible');
          d3.select('.barBtext')
            .text($scope.circleBorderType);
          d3.select('.colorbarB').call(colorbarB);
        }
      }

      function addForce(nodes) {
        var linkRange = d3.scale.linear()
          .domain([0, 0.5])
          .range([5, 100]);

        force = d3.layout.force()
          .charge(function(d) { return d._children ? -d.size * 100 : -
50; })
          .linkDistance(function(d) {  return
linkRange(Number(d.target.dist)); })
          .size([ 0.7 * window.innerWidth / 2, window.innerHeight / 2]);
        var links = d3.layout.tree().links(nodes);

        force.nodes(nodes).links(links);
        force.on('tick', tick);
        force.start();
      }

      $scope.$watch('forceAct.value', function(newForce) {
        console.log('new force act');
        console.log(newForce);
        if (newForce === undefined){ return; }

        if (newForce) {
          addForce(nodes);
        } else if (force) {
          force.stop();
          force = null;
        }
      });

      //update data
      $scope.$watch('treeType', function(newTreeType) {
        //function($scope) { return $scope.treeType === null; }, function()
{
        if (newTreeType === undefined || $scope.data === undefined) {
          return;
        }

        var root  = $scope.data.trees[newTreeType];
        nodes = flatten(root);
        //  force = $scope.data.forces[$scope.treeType];

        //extract the scales
        xScale = d3.scale.linear()
          .domain(d3.extent(nodes, function(d) { return d.x; }))
          .range([0.1 * window.innerWidth, 0.9 * window.innerWidth]);

        yScale = d3.scale.linear()
          .domain(d3.extent(nodes, function(d) { return d.y; }))
          .range([0.1 * window.innerHeight, 0.9 * window.innerHeight]);

        sizeScale = d3.scale.linear()
          .domain(d3.extent(nodes, function(d) { return d.r; }))
          .range([4, 10]);

        changeBorderColorBar(nodes);

        activityColor = function(d) {
          return d._children ? '#3182bd' : d.children ? '#D3D3D3' :
activityScale(d.fill);
        };

        borderColor = function(d) {
          return d._children ? '#CCC' : d.children ? '#D3D3D3' :
borderScale(d.stroke);
        };

        /** Drag behavior configuration **/
        // zoom and drag may interfere with each other, so here redefine
drag function
        function dragstarted(){
          d3.event.sourceEvent.stopPropagation();
          d3.select(this).classed('dragged', true);
        }

        function dragged(d){
          var mouselocation = d3.mouse(vis.node());
          d.x = xScale.invert(mouselocation[0]);
          d.y = yScale.invert(mouselocation[1]);
          tick(); // re-position this node and links connected to this node
        }

        function dragended(){
          d3.select(this).classed('dragging', false);
          //force.resume();
        }

        var drag = d3.behavior.drag()
          .origin(function(d){ return d; }) // identify function
          .on('dragstart', dragstarted)
          .on('drag', dragged)
          .on('dragend', dragended);

        function update() {
          var links = d3.layout.tree().links(nodes);

          //create the selections and bind them to the data
          //console.log(links);
          nodeDOM = vis.selectAll('circle').data(nodes, function(d)
{ return d.name; });

          // Enter any new nodes.
          nodeDOM.enter().append('svg:circle')
            .attr('class', 'node')
            .attr('cx', function(d) { return xScale(d.x); })
            .attr('cy', function(d) { return yScale(d.y); })
            .attr('r', function(d) { return d.children ? 2 :
sizeScale(d.r); })
            .style('fill', activityColor)
            .style('stroke', borderColor)
            .style('stroke-width', function(d) { return d.strokeWidth; })
            //.on('click', click)
            // .on('mouseover', mouseover)
            .call(drag) // attach drag behavior to new circles
            .append('svg:title')
            .text( function(d){ return d.name; });
          // .classed('selected', function (d) { return d.name ===
$scope.id; });


          // transition from old to new
          nodeDOM.filter(function(d) { return d.name[0] === 'B' ? this :
null; })
            //.transition().duration(750)
            .attr('cx', function(d) { return xScale(d.x); })
            .attr('cy', function(d) { return yScale(d.y); });
          nodeDOM.filter(function(d) { return d.name[0] === 'B' ? null :
this; })
            .attr('cx', function(d) { return xScale(d.x); })
            .attr('cy', function(d) { return yScale(d.y); });

          // Exit any old nodes.
          nodeDOM.exit().remove();

          // 2. update new links
          linkDOM = vis.selectAll('line')
            .data(links, function(d) { return d.target.name; });

          // transition from old to new
          linkDOM
            //     .transition(0).duration(750)
            .attr('x1', function(d) { return xScale(d.source.x); })
            .attr('y1', function(d) { return yScale(d.source.y); })
            .attr('x2', function(d) { return xScale(d.target.x); })
            .attr('y2', function(d) { return yScale(d.target.y); });

          // Enter any new links.
          linkDOM.enter().insert('svg:line', '.node')
            .attr('class', 'link')
```

```
      .attr('x1', function(d) { return xScale(d.source.x); })
      .attr('y1', function(d) { return yScale(d.source.y); })
      .attr('x2', function(d) { return xScale(d.target.x); })
      .attr('y2', function(d) { return yScale(d.target.y); })
      .style('stroke', '#D3D3D3')
      .style('stroke-width', '5px');

     // 2. Exit previous links
     linkDOM.exit().remove();

     // 3. transition
     //linkDOM.transition().duration(750)
     //  .style('stroke-opacity', 0.5);

    }

    update();

    //click - select the element that was clicked
    nodeDOM.on('click', function (compound) {
     //  if (compound.name[0] !== 'B') { return click(compound); }
     if (d3.event.defaultPrevented) { return; }
     d3.event.preventDefault();
     if (compound === $scope.selected) {
      $scope.selected = null;
     } else {
      $scope.selected =
$scope.data.compounds[parseInt(compound.name.substring(1))];
     }
     $scope.$apply();
    });



    //zoomer
    /** Zoom behavior configuration **/

    function zoom() {
     tick(); // update position by tick, so the actual d.x, d.y won't
change
    }

    var zoomer = d3.behavior.zoom()
     // allow only 10 times zoom in or out
     .scaleExtent([0.1, 10])
     // attach zoom function for variable modification
     .on('zoom', zoom);

    // let zoomer ajust coordinates by xScale and yScale
    zoomer.x(xScale).y(yScale);

    svg.call(zoomer);

    if ($scope.forceAct.value) {
     addForce(nodes);
    } else if (force) {
     force.stop();
     force = null;
```

```
        }
      });

      $scope.$watch('circleSizeType', function(newCircleSizeType) {

        if (newCircleSizeType === undefined || $scope.data === undefined) {
          return;
        }

        var extent = d3.extent(nodes, function(d) { if (d.name[0] === 'B')
{ return d.r; } });

        var sizeLowerbound = 4;
        var sizeScale = d3.scale.linear()
          .domain(extent)
          .range([sizeLowerbound, 10]);

        // transition from old to new
        if (nodeDOM) {
          nodeDOM.filter(function(d) { return d.name[0] === 'B' ? this :
null; })
            .transition().delay(100).duration(750)
            .attr('r', function(d) { if (extent[0] === extent[1]) { return
sizeLowerbound; } else { return sizeScale(d.r);}});
        }
      });

      $scope.$watch('circleBorderType', function(newCircleBorderType) {

        if (newCircleBorderType === undefined || $scope.data === undefined)
{
          return;
        }

        //nodeDOM = vis.selectAll('circle').data(nodes, function(d)
{ return d.name; });
        changeBorderColorBar(nodes);

        // transition from old to new
        if (nodeDOM) {
          nodeDOM.filter(function(d) { return d.name[0] === 'B' ? this :
null; })
            .transition().delay(100).duration(750)
            .style('stroke', function(d) { return borderScale(d.stroke); })
            .style('stroke-width', function(d) { return d.strokeWidth; });
        }
      });

      $scope.$watch('activityType', function(newActivityType) {

        if (newActivityType === undefined || $scope.data === undefined) {
          return;
        }

        // transition from old to new
        if (nodeDOM) {
          nodeDOM.filter(function(d) { return d.name[0] === 'B' ? this :
null; })
```

```
            .transition().delay(100).duration(750)
            .style('fill', function(d) { return activityScale(d.fill); });
        }
      });

      $scope.$watch('gravityValue', function(newGravity) {

        if (newGravity === undefined || $scope.data === undefined) {
          return;
        }

        //console.log('new gravity:' + newGravity);

        force.resume();
        force.gravity( newGravity );

      });

      $scope.$watch('linkStrengthValue', function(newLinkStrength) {

        if (newLinkStrength === undefined || $scope.data === undefined) {
          return;
        }

        console.log('new linkStrength:' + newLinkStrength);

        force.linkStrength( newLinkStrength );
        force.resume();
        $scope.$apply();

        console.log('tree type is: ', $scope.treeType);
        console.log('setted linkStrength:' + force.linkStrength());
      });


      ////watch the selected scope variable - this can be controlled from
outside the directive or inside the directive
      $scope.$watch('selected', function (selected) {

          if (selected === undefined || $scope.data === undefined) {
            return;
          }

          //if nothing is selected set the $elements to not be selected
          if (selected === null) {
            nodeDOM
              .filter(function(d) { return d.name[0] === 'B' ? this :
null; })
              .style('fill', function (d) {
                return activityScale(d.fill);
              });
          } else {
            nodeDOM
              .filter(function(d) { return d.name[0] === 'B' ? this :
null; })
              .style('fill', function (d) {
                return d.name === selected.id ? 'black' :
activityScale(d.fill);
```

```
            });
        }
      }
    );

  }
  return {
    restrict: 'E',
    link: link,
    scope: {
      data: '=',
      current: '=',
      gravityValue: '=',
      linkStrengthValue: '=',
      treeType: '=',
      circleSizeType: '=',
      circleBorderType: '=',
      activityType: '=',
      selected: '=',
      forceAct: '='
    }
  };
});
```

tooltip.js

```
'use strict';

/**
 * @ngdoc directive
 * @name frontendApp.directive:tooltip
 * @description
 * # tooltip
 */
angular.module('frontendApp')
  .directive('tooltips', function (dataService) {

    return {
        scope: {visibility: '=', data: '=', height: '@', click: '='},
        templateUrl: 'views/tooltip.html',
        restrict: 'E',
        link: function (scope, elements) {

            var parent = $(elements[0]);

            scope.dataService = dataService;

            scope.close = function() {
                scope.visibility = false;
            };


            scope.$watch('visibility', function(newVisibility) {
                if (newVisibility) {
                    parent.show(400);
                } else {
                    parent.hide(400);
                }
            });


        }
    };
  });
```

tree-slider.js

```
'use strict';

/**
 * @ngdoc directive
 * @name frontendApp.directive:treeSlider
 * @description
 * # treeSlider
 */

angular.module('frontendApp')
  .directive('treeSlider', function () {
    return {
      scope: {
        min: '=',
        max: '=',
        value: '=',
        leftName: '=',
        rightName: '=',
        name: '=',
        id: '='
      },
      templateUrl: 'views/tree-slider.html',
      restrict: 'E',
      link: function postLink(scope, elements) {

        var nameTag = $('<div class="name section"></div>'),
          leftSection = $('<div class="left section"></div>'),
          leftLabel = $('<span class="sm-label">left</span>'),
          rightSection = $('<div class="right section"></div>'),
          rightLabel = $('<span class="sm-label">right</span>'),
          slider= $('<div class="slider"  title="Please activate the force 
directed graph first."></div>');

        //console.log('#' + scope.id);
        //console.log(slider);
        var el = elements.find('div');

        $(el).append([nameTag, slider]);

        slider.append([leftSection, rightSection]);
        rightLabel.appendTo(rightSection);
        leftLabel.appendTo(leftSection);

        nameTag.text(scope.name);
        rightLabel.text(scope.rightName);
        leftLabel.text(scope.leftName);

        slider.slider({
          //range: true,
          min: scope.min,
          max: scope.max,
          step: 0.01,
          value: scope.value,
          //animate: 'fast',
          slide: function(evt, ui) {
              scope.value = ui.value;
              scope.$apply();
          }
        });

      }
    };
});
```

data-service.js

```
'use strict';

/**
 * @ngdoc service
 * @name frontendApp.dataService
 * @description
 * # dataService
 * Factory in the frontendApp.
 */
angular.module('frontendApp')
  .factory('dataService', function ($http, $q) {
    // Service logic
    // ...

    // create the service to be returned by the factory
    var dataService = {};

    //selected molecules
    // also a variable dataService.model.selected
    dataService.model = {selected: null};

    //initially, there is no data
    dataService.data = null;

    //initially, there are no representations in data
    dataService.available = {
      treeTypes: [],
      circleSizeTypes: ['None'],
      circleBorderTypes: ['None'],
      activityTypes: []
    };

    //initially, there is no active spaces
    dataService.current = {
      treeType: null,
      circleSizeType: null,
      circleBorderType: null,
      activityType: null
    };

    dataService.flatten = function (root) {
      var nodes = [];

      function recurse(node) {
        if (node.children) { node.r = node.children.reduce(function(p, v) { return p + recurse(v); }, 0);}
        nodes.push(node);
        return node.size;
      }

      root.size = recurse(root);
      return nodes;
    };

    //
    // api for changing the data
    //

    // doesn't touch the data directly, calls
setDimensionalityReductionType to actually change the data
    dataService.setTreeType = function (newTreeType) {

      console.log('Set Tree to ' + newTreeType);
      dataService.current.treeType = newTreeType;

      this.setActivityType(this.current.activityType);

    };

    //
    dataService.setCircleSizeType = function (newCircleSize) {

      console.log('Set circle size:' + newCircleSize);

      // should check if is a member of available
      this.current.circleSizeType = newCircleSize;

      var root  = this.data.trees[this.current.treeType];
      var nodes = dataService.flatten(root);

      if (newCircleSize === 'None') {
        nodes.forEach(function(d) {
          d.r = 2;
        });
      } else {
        nodes.forEach(function(d) {
          if (d.name.startsWith('B')) {
            d.r =
dataService.data.compounds[parseInt(d.name.substring(1))].properties[newCir
cleSize];
          }
        });
      }

    };

    // use to change the border type
    dataService.setCircleBorderType = function (newCircleBorderType) {

      console.log('Set circleBorderType:', newCircleBorderType);

      // should check if is a member of available
      dataService.current.circleBorderType = newCircleBorderType;

      var root  = this.data.trees[this.current.treeType];
      var nodes = dataService.flatten(root);

      if (newCircleBorderType === 'None') {
        nodes.forEach(function(d) {
          d.stroke = 0;
          d.strokeWidth = 0;
        });
      } else {
        nodes.forEach(function(d) {
          if (d.name.startsWith('B')) {
            d.stroke =
dataService.data.compounds[parseInt(d.name.substring(1))].properties[newCir
cleBorderType];
            d.strokeWidth = 3;
          }
        });
      }

    };

    // use to change the activity type
    dataService.setActivityType = function (newActivityType) {

      console.log('Set activity type:', newActivityType);

      // should check if is a member of available
      dataService.current.activityType = newActivityType;

      var root  = this.data.trees[this.current.treeType];
      var nodes = dataService.flatten(root);

      nodes.forEach(function(d) {
        if (d.name.startsWith('B')) {
          d.fill =
dataService.data.compounds[parseInt(d.name.substring(1))].activities[newAct
ivityType];
        }
      });

    };

    //remove all the data
    dataService.empty = function () {
      this.available.treeTypes.length = 0;
      this.available.circleSizeTypes = ['None'];
      this.available.circleBorderTypes = ['None'];
      this.available.activityTypes.length = 0;
      this.current.treeType = null;
      this.current.circleSizeType = null;
      this.current.circleBorderType = null;
      this.current.activityType = null;
      return this;
    };

    dataService.initializeData = function () {

      // remove old representation types if any previously existed
      this.empty();

      // add metadata for each option
      this.metadata = {};

      // add tree types
      this.data.metadata.treeTypes.forEach(function(d) {
        dataService.metadata[d.name] = d.metadata;
        dataService.available.treeTypes.push(d.name); });
      this.setTreeType(this.available.treeTypes[0]);

      // add circle size
      this.data.metadata.circleSizeTypes.forEach(function(d) {
        dataService.metadata[d.name] = d.metadata;
        dataService.available.circleSizeTypes.push(d.name); });

      this.setCircleSizeType(this.available.circleSizeTypes[0]);

      // add tree types
      this.data.metadata.circleBorderTypes.forEach(function(d) {
        dataService.metadata[d.name] = d.metadata;
        dataService.available.circleBorderTypes.push(d.name); });

      this.setCircleBorderType(this.available.circleBorderTypes[0]);

      // add activity types
      this.data.metadata.activityTypes.forEach(function(d) {
        dataService.metadata[d.name] = d.metadata;
        dataService.available.activityTypes.push(d.name); });

      this.setActivityType(this.available.activityTypes[0]);

      // set first in list to be active

      return this;
    };

    // load up an example dataset
    dataService.loadExample = function(name, callback) {

      var delay = $q.defer();

      $http.get('data/' + name + '.json')
        .then(function(response) {
          //retrieve the data as a property
          dataService.datasetName = name;
          dataService.data = response.data;

          //process the data
          dataService.initializeData();

          return delay.resolve(response);
        })
        .then(callback);

      return delay.promise;
    };

    return dataService;
  });
```

settings.js

```
'use strict';

/**
 * @ngdoc service
 * @name frontendApp.settings
 * @description
 * # settings
 * Service in the frontendApp.
 */
angular.module('frontendApp')
  .service('settings', function () {

    this.defaultForce = {value: false};

    this.forceAct = {value: false};

  });
```

# Appendix C.	ChemTreeMap Back End Code

setup.py

```
#! /usr/bin/env python
#
# Copyright (C) 2016 Jing Lu <ajingnk@gmail.com>
# License: Apache

# -*- coding: utf-8 -*-

# pylint: disable=too-few-public-methods

from distutils.core import setup, Extension

from setuptools.command.install import install
from distutils import log # needed for outputting information messages

import os


class OverrideInstall(install):

    def run(self):
        uid, gid = 0, 0
        mode = 0700
        install.run(self) # calling install.run(self) insures that
everything that happened previously still happens, so the installation does
not break!
        # here we start with doing our overriding and private magic ..
        print self.install_scripts
        for filepath in self.get_outputs():
            if filepath.endswith("rapidnj-linux-64"):
                log.info("Overriding setuptools mode of scripts ...")
                log.info("Changing ownership of %s to uid:%s gid %s" %
                         (filepath, uid, gid))
                os.chown(filepath, uid, gid)
                log.info("Changing permissions of %s to %s" %
                         (filepath, oct(mode)))
                os.chmod(filepath, mode)

setup(
    name='treebuild',
    version='0.1.0',
    packages=['treebuild'],
    url='https://github.com/ajing/ChemTreeMap',
    license='Apache 2.0',
    author='ajing',
    author_email='ajingnk@gmail.com',
    description='Generate Tree Structures for Biochemical Similarity in
Molecular Datasets. Please install rdkit and graphviz first, because they
are not pip installable.',
    install_requires=[
        #'rdkit',  ## currently rdkit is not pip installable
        #               ## so this dependency must be met by the installer
        'ete2'
    ],
    package_data={'treebuild': ['data/*.txt', 'data/*.py', 'lib/rapidnj-linux-64']},
    cmdclass={'install': OverrideInstall}
)
```

util.py

```
'''
    Provide utility functions
'''
import datetime
import json
from csv import DictReader
from rdkit import Chem, DataStructs

from ete2 import Tree

from .model import FILE_FORMAT


def GuessByFirstLine(firstline):
    """
    Guess the number of columns with floats by the first line of the file

    :param firstline:
    :return:
    """
    num_colnam = []
    for key in firstline:
        try:
            float(firstline[key])
            num_colnam.append(key)
        except:
            continue
    return num_colnam

def ConvertToFloat(line, colnam_list):
    """
    Convert some columns (in colnam_list) to float, and round by 3 decimal.

    :param line: a dictionary from DictReader.
    :param colnam_list: float columns
    :return: a new dictionary
    """
    for name in colnam_list:
        line[name] = round(float(line[name]), 3)
    return line

def ParseLigandFile(infile, identifier):
    """
    Parse ligand file to an dictionary, key is ligand id and valud is a
dictionary with properties and property values.
    This program will guess the type for each column based on the first
row. The program will assume there is only two types of data: number and
string.

    :param infile: input filename
    :param identifier: the identifier column name
    :return: a dictionray
    """
    '''

    '''
    mol_dict = dict()
    flag = 1 # first line flag
    id_count = 0
    for line in DictReader(open(infile), delimiter = "\t"):
        if flag:
            num_colnam = GuessByFirstLine({k:v for k,v in line.items() })
        new_id = "B" + str(id_count)
        id_count += 1
        mol_dict[new_id] = ConvertToFloat({k:v for k,v in line.items()},
num_colnam)
        mol_dict[new_id]["orig_id"] = line[identifier]
    return mol_dict


def WriteJSON(dict_obj, outfile, write_type):
    """
    Dump json object to a file.

    :param dict_obj: dictionary object
    :param outfile: output file name
    :param write_type: append or rewrite ('a' or 'w')
    :return: void
    """
    fileobj = open(outfile, write_type)
    fileobj.write(json.dumps(dict_obj))


def SelectColumn(lig_dict, colname):
    """
    Prune the dictionary, only attribute in colname will be left.

    :param lig_dict: a tree like dictionary
    :param colname: what attribute you want to keep.
    :return: a new dictionary
    """
    lig_new = dict()
    for k in lig_dict:
        lig_new[k] = {sk:v for sk, v in lig_dict[k].items() if sk in
colname}
    return lig_new


def WriteAsPHYLIPFormat(smile_list, fp_func):
    """
    Prepare the input for RapidNJ.

    :param smile_list: a list of smiles string
    :param fp_func: the fingerprint function
    :return: tje filename with PHYLIP format (input for rapidnj)
    """
    fp_list = ToFPObj(smile_list, fp_func)
    print "finish parsing smile list"
    list_len = len(fp_list)

    newfilename = datetime.datetime.now().strftime(FILE_FORMAT) + ".dist"
    fileobj  = open(newfilename, "w")
    fileobj.write( str(list_len) + "\n")
    for i in range(list_len):
        lig1 = fp_list[i]
        lig1list = []
        for j in range(list_len):
            lig2 = fp_list[j]
            sim  = getSimilarity(lig1[1], lig2[1])
            lig1list.append([lig2[0], 1 - sim])

        sim_values = [ "%.4f" % x[1] for x in lig1list]
        line = "\t".join([lig1[0], "\t".join(sim_values)]) + "\n"
        fileobj.write(line)

    fileobj.close()

    return newfilename


def ToFPObj(alist, fp_func):
    """
    A list of SMILE string object with (id, smiles) to a list of
fingerprint object with (id, fp_obj)

    :param alist: a list of two element list, the first item is ligand
name, the second is smile
    :param fp_func: the fingerprint function
    :return: a new list of two element list, with first item as ligand
name, second item as a fingerprint object.
    """
    newlist = []
    for each in alist:
        smile = each[1]
        m = Chem.MolFromSmiles(smile)
        if m is None:
            continue
        fp = fp_func(m)
        newlist.append([each[0], fp])
    return newlist


def WriteDotFile(newick):
    """
    Write newick string to a DOT file

    :param newick: a string with newick tree structure
    :return: DOT file name
    """
    tree = Tree(newick)

    dot_file_name = datetime.datetime.now().strftime(FILE_FORMAT) + ".gv"
    fileobj = open(dot_file_name, "w")

    # rename internal tree name
    i = 0
    for n in tree.traverse():
        if not n.name:
            n.name = "F" + str(i)
            i = i + 1
        else:
            n.name = n.name.replace("\'", "")

    aline = "graph G{\nnode [shape=circle, style=filled];"
    fileobj.write(aline + "\n")
    filecontent = []
    for n in tree.traverse():
        if n.up:
            filecontent.append(n.name + "--" + n.up.name + "[len=" +
"{:f}".format(n.dist).rstrip("0") + "]")
        else:
            filecontent.append(n.name)

    fileobj.write("\n".join(filecontent) + "}")
    return dot_file_name


def RemoveBackSlash(dotfile):
    """
    Rewrite dot file, with removing back slash of dot file.

    :param dotfile: DOT file name
    :return: void
    """
    # remove backslash and replace all " quote sign
    f = open(dotfile, 'r+')
    content = f.readlines()
    newcontent = []
    for line in content:
        line = line.replace("\"", "")
        if line.endswith("\\\n"):
            newcontent.append(line[:-2])
        elif line.endswith("\n") and line[-2] != ";":
            newcontent.append(line[:-1])
        else:
            newcontent.append(line)
    f.seek(0)
    f.write("".join(newcontent))
    f.truncate()
    f.close()


def Dot2Dict(dotfile, moldict):
    """
    Read a DOT file to generate a tree and save it to a dictionary.

    :param dotfile: DOT file name
    :param moldict: a dictionary with ligand information
    :return: a dictionary with the tree
    """
    rootname = "F0"
    # dotfile is a dot file
    contents = open(dotfile).readlines()
    # get the root of the network
    root = GetRoot(dotfile, rootname)
    curr_nodes = [root]
    curr_name_list = [root.name]
    next_nodes     = 1
    while next_nodes:
        next_nodes = []
        for each_node in curr_nodes:
            next_nodes += extendChildren(each_node, contents,
curr_name_list)
        curr_nodes = next_nodes
    rootdict = RecursiveNode2Dict(root, moldict)
    return rootdict


def getSimilarity(fp1, fp2):
    """
    Generate similarity score for two smiles strings.

    :param fp1: fingerprint object (rdkit)
    :param fp2: fingerprint object (rdkit)
    :return: Tanimoto similarity
    """
    if (fp1 is None or fp2 is None):
        return
    return DataStructs.TanimotoSimilarity(fp1, fp2)


def GetRoot(dotfile, rootname):
    """
    Return root name with rootname.

    :param dotfile: DOT file
    :param rootname: the name of the root
    :return: the object of the root
    """
    for eachline in open(dotfile):
        if NodeNameExist(eachline) and not IsEdge(eachline):
            name, attr = NameAndAttribute(eachline)
            name = name.strip()
            if name == rootname:
                name, size, position = GetNodeProperty(eachline)
                return Node(name, size = size, position = position)


def extendChildren(a_node, contents, cur_list):
    """
    Find all children of a node in a tree.

    :param a_node: a node in a tree
    :param contents: contents from DOT file
    :param cur_list: current children
    :return: a list of node objects (children)
    """
    children_list = []
    for eachline in contents:
        if IsEdge(eachline):
            name, attr = NameAndAttribute(eachline)
            fnode, snode = ProcessName(name, True)
            eachline = CleanAttribute(eachline)
            if fnode == a_node.name and not snode in cur_list:
                edge_len  = GetAttributeValue("len", eachline)
                AddNewChild(contents, a_node, snode, edge_len,
children_list, cur_list)
```

```
            if snode == a_node.name and not fnode in cur_list:
                edge_len  = GetAttributeValue("len", eachline)
                AddNewChild(contents, a_node, fnode, edge_len,
children_list, cur_list)
    return children_list


def IsEdge(line):
    """
    Whether this line in DOT file is an edge.

    :param line: a string line in DOT file
    :return: True or False
    """
    if "--" in line:
        return True
    else:
        return False


def RecursiveNode2Dict(node, info_dict):
    '''
    Recursively populate information to the tree object with info_dict.

    :param node: tree object with all info
    :param info_dict: information for each ligand.
    :return: a tree dictionary
    '''
    if not node.children:
        x, y   = map(float, node["position"].split("-"))
        result = {"name": node.name, "size": 1, "x": x, "y": y, "dist":
abs(float(node.dist))}
        if info_dict:
            result.update(info_dict[node.name])
    else:
        x, y   = map(float, node["position"].split("-"))
        result = {"name": node.name, "x": x, "y": y, "dist":
abs(float(node.dist))}
        if info_dict and node.name in info_dict:
            result.update(info_dict[node.name])
    children = [RecursiveNode2Dict(c, info_dict) for c in node.children]
    if children:
        result["children"] = children
    return result


def NodeNameExist(line):
    """
    Functions for parsing DOT file.

    :param line: a line from DOT file
    :return: whether there is a node name in this line
    """
    if "CHEMBL" in line or "ASD" in line or "Chk1" in line or "B" in line
or "F" in line:
        return True
    else:
        return False
```

```
def NameAndAttribute(line):
    """
    Split name and attribute.

    :param line: DOT file name
    :return: name string and attribute string
    """
    split_index = line.index("[")
    name   = line[:split_index]
    attr   = line[split_index:]
    return name, attr


def AddNewChild(contents, a_node, new_node_name, edge_length, children,
currentlist):
    """
    Add a new child to a node.

    :param contents: a string, a line from DOT
    :param a_node: a node object
    :param new_node_name: new node name
    :param edge_length: the length of edge
    :param children: existing children
    :param currentlist: current list of node name
    :return: void
    """
    # return a node object
    newnode = NodeByName(new_node_name, contents)
    newnode.set_dist(edge_length)
    a_node.add_child(newnode)
    children.append(newnode)
    currentlist.append(new_node_name)


def GetNodeProperty(line):
    """
    Get node property from a string.

    :param line: a string
    :return: name, size, and position of the node
    """
    name, attr = NameAndAttribute(line)
    name = ProcessName(name, False)
    position = GetAttributeValue("pos", attr)[:-1].replace(",", "-")
    attr = CleanAttribute(attr)
    width = GetAttributeValue("width", attr)
    #group = GetAttributeValue("color", attr)
    size = SizeScale(GetSize(width))
    return name, size, position


def ProcessName(name, isedge):
    """
    Process the name of the node.

    :param name: name of the node
    :param isedge: whether this is a edge
    :return: new name
    """
    if isedge:
        firstnode, secondnode = name.split("--")
        firstnode = firstnode.strip()
        secondnode = secondnode.strip()
        return firstnode, secondnode
    else:
        return name.strip()


def GetAttributeValue(attrname, attr):
    """
    Get node attribute.

    :param attrname: name of the attribute
    :param attr: the attribute string
    :return: the value for the attribute
    """
    left = attr.index("[") + 1
    right = attr.index("]")
    attr  = attr[left:right]
    attrlist = attr.split()
    for each in attrlist:
        if attrname in each:
            value = each.split("=")[1]
            if value.endswith("!"):
                return value[:-1]
            else:
                return value


def CleanAttribute(attr):
    """
    Clean attribute, remove ','.

    :param attr: old attribute string
    :return: new string
    """
    new_attr = attr.replace(",", "")
    return new_attr


def NodeByName(name, contents):
    """
    Create node with name name.

    :param name: a string with node name
    :param contents: a list of string from DOT file
    :return: node object
    """
    for eachline in contents:
        if not IsEdge(eachline) and NodeNameExist(eachline):
            nodename, attr = NameAndAttribute(eachline)
            if name == nodename.strip():
                name, size, position = GetNodeProperty(eachline)
                return Node(name, size = size, position = position)
```

```
def SizeScale(size):
    """
    Rescale the size (currently only convert to float).

    :param size: a string
    :return: a float
    """
    return float(size)


def GetSize(width):
    """
    Get the size.

    :param width:
    :return:
    """
    if isinstance(width, str):
        width = float(width)
    return width


class Node(dict):
    """
    class for node of tree, each node can only have one parent
    """
    def __init__(self, name, **attr):
        self.name = name
        self.parent = None
        self.children = []
        self.dist   = 0
        self.update(attr)

    def __str__(self):
        return "a node with name:" + self.name

    def get_dist(self, a_node):
        """
        get the node as a dictionary.

        :param a_node: Node object
        :return: a dictionary
        """
        if not isinstance(a_node, Node):
            raise TypeError("argument should be Node class")
        if a_node == self.parent:
            return self.dist
        if a_node in self.children:
            return a_node.dist
        else:
            return None

    def add_child(self, a_node):
        """
        Add child to the node.
        :param a_node: Node object
        :return: void
        """
        if not isinstance(a_node, Node):
            raise TypeError("argument should be Node class")
        self.children.append(a_node)
        a_node.set_parent(self)

    def set_parent(self, a_node):
        """
        Set the parent for a node.

        :param a_node: Node object
        :return: void
        """
        if not isinstance(a_node, Node):
            raise TypeError("argument should be Node class")
        self.parent = a_node

    def set_dist(self, dist):
        """
        set the dictionary attribute for the Node object.

        :param dist:
        :return:
        """
        self.dist = dist


if __name__ == "__main__":
    ParseLigandFile("./Data/thrombin_clean_ct.txt")
```

types.py

```
#! /usr/bin/env python
#
# Copyright (C) 2016 Jing Lu <ajingnk@gmail.com>
# License: Apache

# -*- coding: utf-8 -*-

# pylint: disable=too-few-public-methods
import math
from rdkit import Chem
from rdkit.Chem import AllChem
from rdkit.Chem.AtomPairs import Pairs

from .model import SMILE_COLUMNNAME, POTENCY


class FingerPrintType:
    """
    representing fingerprint types
    """
    def __init__(self, name, fp_func, metadata):
        """
        Initialize the fingerprint type

        :param name: name of fingerprint
        :param fp_func: the function to generate a fingerprint
        :param meta: string for fingerprint document
        :return:
        """
        self.name = name
        self.fp_func = fp_func
        self.metadata = metadata

    def to_dict(self):
        """
        Show the information for this fingerprint

        :return: dictionary with basic info
        """
        return {"name": self.name, "metadata": self.metadata}


class PropertyType:
    """
    representing biological or chemical properties
    """
    def __init__(self, name, metadata, transfunc=None, colname=None):
        """
        :param name: a string name of property
        :param metadata: a string with property meaning
        :param transfunc: a function to generate the property
        :return:
        """
        self.name = name
        self.metadata = metadata
        self.transfunc = transfunc  # the signiture for transfunc is 
transfunc(a_value, a_mol_dict)
        self.colname = colname

    def set_col_name(self, col_name):
        """
        Set the property name from the input file

        :param col_name: original column name in the input file
        :return:
        """
        self.colname = col_name

    def gen_property(self, mol_dict = None):
        """
        generate value for this property type

        :param prop_name: the name of the property
        :param mol_dict: other information about the molecule
        :return: a generated value for this property
        """
        # if self.colname is None:
        #     raise Exception("Please set the column name for this 
property")

        if self.colname in mol_dict:
            return mol_dict[self.colname]

        if self.name in mol_dict:
            return mol_dict[self.name]

        if not self.transfunc is None and not mol_dict is None:
            return self.transfunc(mol_dict)
        else:
            raise Exception("please provide the transformation function and 
molecule information for " + str(self))


    def to_dict(self):
        """
        Show the information

        :return: dictionary with basic info
        """
        return {"name": self.name, "metadata": self.metadata}

    def __str__(self):
        return self.name

# property functions
def _lig_eff(mol_dict):
    smile = mol_dict[SMILE_COLUMNNAME]
    m = Chem.MolFromSmiles(smile)
    num_heavy = m.GetNumHeavyAtoms()
    if POTENCY in mol_dict:
        ic50  = mol_dict[POTENCY]
        return round(1.37 * (9 - math.log10(ic50)) / num_heavy, 5)
    elif "pIC50" in mol_dict:
        pic50 = mol_dict["pIC50"]
        return round(1.37 * pic50 / num_heavy, 5)
    else:
        raise Exception("Cannot calculate ligand efficiency, please change
your input file column name to IC50 or pIC50.")


def _slogp(mol_dict):
    smile = mol_dict[SMILE_COLUMNNAME]
    m = Chem.MolFromSmiles(smile)
    return round(Chem.rdMolDescriptors.CalcCrippenDescriptors(m)[0], 5)

def _pic50(mol_dict):
    ic50 = mol_dict[POTENCY]
    return round(9 - math.log10(float(ic50)), 5)

ecfp6 = FingerPrintType(name = "ECFP6", fp_func= lambda mol:
AllChem.GetMorganFingerprint(mol, 3), metadata = "Extended Connectivity
fingerprint, implemented in <a href=\"http://www.rdkit.org\">RDKit</a>.
<br/>Parameters used: Radius = 3")

atom_pair = FingerPrintType(name = "AtomPair", fp_func=
Pairs.GetAtomPairFingerprint, metadata = "Atom Pairs as Molecular Features,
describe in  R.E. Carhart, D.H. Smith, R. Venkataraghavan; \"Atom Pairs as
Molecular Features in Structure-Activity Studies: Definition and
Applications\" JCICS 25, 64-73 (1985).implemented in <a
href=\"http://www.rdkit.org\">RDKit</a>. <br/>")

lig_eff = PropertyType(name = "Lig_Eff", metadata = "Ligand efficiency. The
value is calculated by the function 1.37 * pIC50 / a_heavy", transfunc =
_lig_eff)

slogp   = PropertyType(name = "SLogP", metadata = "SLogP, the coefficients
are a measure of the difference in solubility of the compound in water and
octanol. describe in    S. A. Wildman and G. M. Crippen JCICS 39 868-873
(1999) R.E. Carhart, D.H. Smith, R. Venkataraghavan; \"Atom Pairs as
Molecular Features in Structure-Activity Studies:", transfunc = _slogp)

ic50   = PropertyType(name = "IC50", colname = "IC50", metadata = "The half
maximal inhibitory concentration (IC50) is a measure of the effectiveness
of a substance in inhibiting a specific biological or biochemical
function.")

pic50 = PropertyType(name = "pIC50", metadata = "This number assumes IC50
in nM unit, so it is calculated by 9 - log(IC50). Please change your data
or the code to make it appropriate.", transfunc = _pic50)

bindingdb = {"name": "BindingDB", "link":
"https://www.bindingdb.org/bind/chemsearch/marvin/MolStructure.jsp?monomeri
d="}
chebi = {"name": "CHEBI", "link":
"https://www.ebi.ac.uk/chebi/searchId.do?chebiId="}
pubchem = {"name": "PubChem", "link":
"https://pubchem.ncbi.nlm.nih.gov/compound/"}

DEFAULT_FINGERPRINT_TYPES = [ecfp6, atom_pair]
DEFAULT_ACTIVITY_TYPES = [ic50, pic50]
```

```
DEFAULT_PROPERTY_TYPES = [lig_eff, slogp]
DEFAULT_EXTERNAL = [bindingdb, pubchem]
```

tree_build.py

```
#! /usr/bin/env python
#
# Copyright (C) 2016 Jing Lu <ajingnk@gmail.com>
# License: Apache

# -*- coding: utf-8 -*-

# pylint: disable=too-few-public-methods

import datetime
import os
import subprocess
import shutil
from rdkit import Chem
from rdkit.Chem.Draw import MolToFile

from .types import FingerPrintType
from .util import ParseLigandFile, WriteJSON, WriteAsPHYLIPFormat,
Dot2Dict, \
    WriteDotFile, RemoveBackSlash
from .model import IMG_DIR, SMILE_COLUMNNAME, RAPIDNJ_COMMAND, FILE_FORMAT,
TMP_FOLDER

class TreeBuild:
    """
    There are a few assumptions for the input file:
        1. potency unit is nM
        2. the file must have a id column, you can set the column name with
id_column
        3. the file must have a SMILES column, with 'Canonical_Smiles' as
column name
        4. the file must have at least one activity column.
    To build the tree
        1. the identity column needs to be specified with id_column
        2. a list of fingerprints and a list of properties need to be
specified
        3. the directories for input and output file are also needed to be
specified
    """
    def __init__(self, input_file, output_file, id_column, fps,
properties):
        """Setting parameters to build the tree.

        :param input_file: input file is a tab delimited text file.
        :param output_file: output file is a json file
        :param id_column: the id for each column, which will shown as the
identifier in the visualization.
        :param fps: a list of FingperPrintType
        :param properties: a list of PropertyType
        :return: void, the program will generate input file for the
visualization.
        """
        # initial setting
        self._RAPIDNJ_COMMAND = RAPIDNJ_COMMAND
        self._FILE_FORMAT = FILE_FORMAT

        # creating folders
        if not os.path.exists(TMP_FOLDER):
            os.makedirs(TMP_FOLDER)
        if not os.path.exists(IMG_DIR):
            os.makedirs(IMG_DIR)

        activities = properties["activities"]
        other_properties = properties["properties"]
        ext_links = properties["ext_links"]
        lig_dict = self.parse_lig_file(input_file, id_column)
        trees = dict()
        for fp in fps:
            assert isinstance(fp, FingerPrintType)
            trees[fp.name] = self.build_single_tree(lig_dict, fp)
        metadata = dict()
        metadata["activityTypes"] = [act.to_dict() for act in activities]
        metadata["treeTypes"] = [fp.to_dict() for fp in fps]
        metadata["circleSizeTypes"] = [prop.to_dict() for prop in
other_properties]
        metadata["circleBorderTypes"] = [prop.to_dict() for prop in
other_properties]
        metadata["external"] = ext_links

        ext_names = [ext["name"] for ext in ext_links]

        comp_info = self.gen_properties(lig_dict, activities,
other_properties, ext_names)
        final_dict = {"metadata": metadata, "trees": trees, "compounds":
comp_info}

        WriteJSON(final_dict, outfile=output_file, write_type="w")
         # make image file
        self.make_structures_for_smiles(lig_dict)

        # delete tmp folder
        shutil.rmtree(TMP_FOLDER)


    def build_single_tree(self, lig_dict, fp):
        """
        Build a single tree with fingerprint function

        :param lig_dict: all ligand information
        :param fp: fingerprint object
        :return: dot filename
        """
        distfile = self.gen_dist_file(lig_dict, fp.fp_func)
        newick_o = self.run_rapidnj(distfile)
        dot_inf = self.write_dotfile(newick_o)
        dot_out = self.sfdp_dot(dot_inf, 10)
        dot_dict = self.dot2dict(dot_out)
        return dot_dict

    @staticmethod
    def parse_lig_file(in_file, identifier):
        """
        parse ligand file and return a dictionary with identifier as IDs

        :param in_file: input file directory
        :param identifier: name for the identifier
        :return: a dictionray with ligand information
        """
        return ParseLigandFile(in_file, identifier)

    @staticmethod
    def gen_dist_file(liganddict, fp_func):
        """
        generate distance file which is the input of rapidnj program.

        :param liganddict: ligand information
        :param fp_func: fingerprint function
        :return: filename for distance file
        """
        smile_list = [ [lig_name, liganddict[lig_name][SMILE_COLUMNNAME]]
for lig_name in liganddict.keys()]
        print "finish smile list"
        filename   = WriteAsPHYLIPFormat(smile_list, fp_func)
        print "finish writing phyli file"
        return filename

    def run_rapidnj(self, distance_file):
        """
        run rapidnj program on distance_file

        :param distance_file: directory of distance file
        :return: newick string
        """
        proc = subprocess.Popen([self._RAPIDNJ_COMMAND, distance_file, "-
i", "pd"], stdout=subprocess.PIPE)
        newick = proc.stdout.read()
        return newick

    @staticmethod
    def write_dotfile(newick):
        """
        write newick string as dot file

        :param newick: newick string
        :return: dot file
        """
        return WriteDotFile(newick)

    def sfdp_dot(self, dot_infile, size):
        """
        run sdfp on dot file

        :param dot_infile: directory for dot file
        :param size: parameter for the sfdp
        :return: new filename
        """
        fmt= self._FILE_FORMAT + '_sfdp.gv'
        newfilename = datetime.datetime.now().strftime(fmt)
        if os.path.isfile(newfilename):
            os.remove(newfilename)
        command = "sfdp -Gsmoothing=triangle -Gsize={size} {infile} >
{outfile}".format(size=size, infile=dot_infile, outfile=newfilename)
        subprocess.Popen( command, shell = True, stdout =
subprocess.PIPE ).communicate()
        RemoveBackSlash(newfilename)
        return newfilename

    @staticmethod
    def dot2dict(dot_outfile):
        return Dot2Dict(dot_outfile, None)

    @staticmethod
    def gen_properties(ligand_dict, activities, properties, ext_cols):
        """
        Generate properties for each molecule.

        :param ligand_dict: ligand dictionary which keep all ligand
information
        :param activities: a list of PropertyType objects
        :param properties: a list of PropertyType objects
        :param ext_cols: the column name for external links
        :return:
        """
        compounds = []
        for idx in range(len(ligand_dict)):
            lid = "B" + str(idx)
            comp = dict()
            comp["id"] = lid
            comp["orig_id"] = ligand_dict[lid]["orig_id"]
            comp["activities"] = dict()
            comp["properties"] = dict()
            comp["external"] = dict()
            for act in activities:
                comp["activities"][act.name] =
act.gen_property(ligand_dict[lid])
            for prop in properties:
                comp["properties"][prop.name] =
prop.gen_property(ligand_dict[lid])
            for col in ext_cols:
                ext_val = ligand_dict[lid][col]
                if isinstance(ext_val, float):
                    comp[col] = str(int(ext_val))
                else:
                    comp[col] = str(ext_val)
            compounds.append(comp)

        return compounds

    @staticmethod
    def make_structures_for_smiles( ligand_dict ):
        """
        Make structure figures from smile strings. All image files will be
in the IMG_DIR

        :param ligand_dict: ligand dictionary which keep all ligand
information
        :return:
        """
        relative_dir = IMG_DIR
        for key in ligand_dict:
            smile = ligand_dict[key][ SMILE_COLUMNNAME ]
            filename = ligand_dict[ key ][ "orig_id" ]
            mol = Chem.MolFromSmiles(smile)
            try:
                MolToFile( mol, os.path.join(relative_dir,
'{}.svg'.format(filename)) )
            except:
                raise Exception("cannot write to file: " +
os.path.join(relative_dir, '{}.svg'.format(filename)))
```

model.py

```
#! /usr/bin/env python
#
# Copyright (C) 2016 Jing Lu <ajingnk@gmail.com>
# License: Apache

# -*- coding: utf-8 -*-

# pylint: disable=too-few-public-methods

"""
    Keep consistent among all files
"""
import os


SMILE_COLUMNNAME = "Canonical_Smiles"

RAPIDNJ_COMMAND = os.path.join(os.path.dirname(__file__), "lib/rapidnj-
linux-64")

# temperary files
TMP_FOLDER  = "./.tmp"
FILE_FORMAT = './.tmp/%Y-%m-%d-%Hh-%Mm-%Ss'
## image directory
IMG_DIR = "./images/"

# Potency
POTENCY = "IC50"
```

generate_ids.py

```
#! /usr/bin/env python
#
# Copyright (C) 2016 Jing Lu <ajingnk@gmail.com>
# License: Apache

# -*- coding: utf-8 -*-

# pylint: disable=too-few-public-methods

class GenerateIDs:
    """
    Retrieve other external ids for BindingDB IDs.
    """
    def __init__(self, infile, outfile):
        """
        Initialize the object with a spreadsheet input file, and output a
file with
        CHEBI ID and PubChem ID.

        :param infile:
        :param outfile:
        :return:
        """
        self.infile = infile
        self.outfile = outfile

        self.pubchem_dict =
self.parse_dict_file("./data/BindingDB_CID.txt")
        self.chebi_dict =
self.parse_dict_file("./data/BindingDB_CHEBI_ID.txt")

        self.cannot_find_pubchem = 0
        self.cannot_find_chebi   = 0
        self.run_for_file()

        print("# PubChem ID cannot find " + str(self.cannot_find_pubchem))
        print("# CHEBI ID cannot find " + str(self.cannot_find_chebi))


    @staticmethod
    def parse_dict_file(filename):
        """
        Parse BindingDB_CID and BindingDB_CHEBI_ID

        :param filename: ID mapping between BindingDB and other databases
        :return: dictionray with BindingDB ID as the key.
        """
        id_dict = dict()
        with open(filename) as fobj:
            for line in fobj:
                cont = line.split()
                id_dict[cont[0]] = cont[1]
        return id_dict
    def get_pubchemid_from_bdid(self, bdid):
        """
        Get PubCHEM ID with BindingDB ID

        :param bdid: bindingdb id
        :return: pubchem id
        """
        try:
            return self.pubchem_dict[bdid]
        except:
            self.cannot_find_pubchem += 1
            return None

    def get_chebiid_from_bdid(self, bdid):
        """
        Get CHEBI ID with BindingDB ID

        :param bdid: CHEBI ID
        :return: pubchem id
        """
        try:
            return self.chebi_dict[bdid]
        except:
            self.cannot_find_chebi += 1
            return None

    def run_for_file(self):
        """
        Create a new file based on the input file

        :return:
        """
        import csv
        out_obj = open(self.outfile, "w")
        with open(self.infile) as in_obj:
            reader = csv.DictReader(in_obj, delimiter = '\t')
            writer = csv.DictWriter(out_obj, reader.fieldnames +
["PubChem"], delimiter = '\t')
            writer.writeheader()
            for line in reader:
                line["PubChem"] =
self.get_pubchemid_from_bdid(line["BindingDB"])
                #line["CHEBI"] =
self.get_chebiid_from_bdid(line["BindingDB"])
                writer.writerow(line)

if __name__ == "__main__":
    import sys
    GenerateIDs(sys.argv[1], sys.argv[2])
```

__init__.py

```python
#! /usr/bin/env python
#
# Copyright (C) 2016 Jing Lu <ajingnk@gmail.com>
# License: Apache

# -*- coding: utf-8 -*-

# pylint: disable=too-few-public-methods

from .tree_build import TreeBuild
from .types import (
    FingerPrintType,
    PropertyType,
    DEFAULT_ACTIVITY_TYPES,
    DEFAULT_FINGERPRINT_TYPES,
    DEFAULT_PROPERTY_TYPES,
    DEFAULT_EXTERNAL
)
```

Appendix D. Allosteric versus Competitive Compound Supplementary Information

Table D-1 Protein Targets for Allosteric and Competitive Compounds

Allosteric compound targets	ligand count	Competitive compound targets	ligand count
Metabotropic glutamate receptor 1	1947	Acyl-CoA synthase	309
Metabotropic glutamate receptor 2	1195	Metabotropic glutamate receptor 5	204
Muscarinic acetylcholine receptor M1	969	Menin/Histone-lysine N-methyltransferase MLL	202
Alpha-7 nicotinic receptor	722	Estrogen receptor beta	184
Metabotropic glutamate receptor 6	502	Dopamine D2 receptor	170
Glutamate receptor ionotropic, AMPA	465	Adenosine A2b receptor	156
NMD-R1	402	Muscarinic acetylcholine receptor M1	118
ABPP	356	Thrombin	82
GABA receptor alpha-2 subunit	351	Androgen Receptor	75
GABA(A) receptor subunit gamma-2	349	Monoamine oxidase A	75
GABABR1	347	Histamine H3 receptor	74
GABA-B-R2	345	HERG	69
GABA-A receptor beta-3 subunit	340	Dopamine transporter	66
Adenosine receptor A2b	330	Progesterone receptor	63
Alpha-3 beta-4 nicotinic acetylcholine receptor	324	Acetylcholinesterase	56
Neuronal nicotinic acetylcholine receptor alpha-4 subunit	300	Dopamine D4 receptor	54
MAPK/ERK kinase 1	291	Inhibitor of apoptosis protein 3	54
Presenilin-1 CTF subunit	283	Thromboxane A2 receptor	52
RNA-directed RNA polymerase	275	Nociceptin receptor	51
PK	275	Mineralocorticoid receptor	49
PK-1	275	Alpha-1b adrenergic receptor	46
PtdIns-3-kinase subunit p110-gamma	270	Endothelin receptor ET-A	44
Cannabinoid CB1 receptor	247	Glutamate (NMDA) receptor subunit zeta 1	42
Avian erythroblastic leukemia viral (v-erb-b) oncogene homolog	239	Cytochrome P450 19A1	41

Drosophila relative of ERBB	232	Carbonic anhydrase II	41
G-protein coupled receptor 71	229	Receptor-type tyrosine-protein phosphatase beta	41
Sweet taste receptor T1R3	229	Glucocorticoid receptor	38
AMPK subunit alpha-2	228	MAP kinase p38 alpha	38
AMPK subunit beta-1	228	Glandular kallikrein	38
Protein kinase AMP-activated gamma	228	Integrin alpha-5/beta-1	38
Uncharacterized protein MJ1225 Protein-Kinase	228	Aldo-keto-reductase family 1 member C3	37
RT	219	Beta-glucocerebrosidase	36
Cell division protein kinase 2	217	Translocator protein	35
GLP-1 receptor	202	Histamine H2 receptor	33
CAMP-specific phosphodiesterase PDE4D6	201	Serine/threonine-protein kinase Chk1	33
HAC-1	198	Peroxisome proliferator-activated receptor alpha	31
Hyperpolarization-activated (Ih) channel	198	11-beta-hydroxysteroid dehydrogenase 1	30
Kinesin-like spindle protein HKSP	181	Adenosine A2a receptor	30
GL-R	178	Melatonin receptor	30
Interleukin-1 receptor type I	177	Muscarinic acetylcholine receptor M2	29
PN3	175	PI3-kinase p110-beta subunit	29
L-glutamine amidohydrolase	166	Histamine H4 receptor	27
HPDK1	162	Rap guanine nucleotide exchange factor 4	27
p21-activated kinase 1	157	Histamine H1 receptor	26
Muscarinic acetylcholine receptor M2	152	Muscarinic acetylcholine receptor M5	26
Sodium channel, voltage-gated, type I, alpha polypeptide	149	Dopamine D1 receptor	25
Protein kinase B	138	Prostanoid EP2 receptor	25
FBPase 1	136	Retinoid X receptor beta	24
D-fructose-1,6-bisphosphate 1-phosphohydrolase	136	Neuronal acetylcholine receptor protein alpha-7 subunit	23
FBPase class 1	136	Subtilisin	23
Non-structural protein 5	136	Cannabinoid CB1 receptor	22
MMP-13	124	3-phosphoinositide dependent protein kinase-1	22
S1P receptor Edg-3	124	Beta-2 adrenergic receptor	21
GluN2B	120	Trypanothione reductase	21
Integrase	120	Cannabinoid receptor	21

CAMP-dependent protein kinase type II-beta regulatory chain	119	Polymerase acidic protein	21
PKA C-beta	119	Serotonin 2a (5-HT2a) receptor	21
Tissue-specific extinguisher 1	119	FK506 binding protein 4	20
Follicle stimulating hormone receptor	118	Fatty acid-binding protein, liver	20
Biotin carboxylase	117	Glutamate receptor ionotropic, AMPA 2	20
Prealbumin	114	Plasminogen	20
DNA polymerase beta	103	Renin	19
GRO/MGSA receptor	100	Serotonin 1a (5-HT1a) receptor	19
Hsp90	99	Complement C1s	19
Bruton tyrosine kinase	96	PI3-kinase p110-alpha subunit	19
Serotonin receptor 2A	92	GABA-A receptor; anion channel	18
Rapamycin and FKBP12 target 1	86	Gamma-amino-N-butyrate transaminase	18
EK4	81	Acrosin	18
DA transporter	78	Aminopeptidase N	18
5-hydroxytryptamine receptor 3	74	Cholecystokinin B receptor	18
ALADH	72	Hepatocyte growth factor receptor	18
hSIRT1	69	Peptidyl-glycine alpha-amidating monooxygenase	18
Exoribonuclease H	63	Transthyretin	18
SDH	63	Apoptosis regulator Bcl-X	17
MAP kinase 14	58	Galectin-3	17
Short transient receptor potential channel 4	57	Galectin-9	17
Tyrosine kinase non-receptor protein 2	57	Melanin-concentrating hormone receptor 1	17
Heat shock 70 kDa protein 1/2	55	Neurokinin 2 receptor	17
5-HT1A	54	Platelet activating factor receptor	17
Cytochrome P450 NF-25	54	RAC-alpha serine/threonine-protein kinase	17
Farnesyl diphosphate synthase	54	CpG DNA methylase	17
B2R	53	Rho-associated protein kinase 1	17
Dopamine D3 receptor	53	Ryanodine receptor 1	17
Serotonin receptor 1A	53	Beta-1 adrenergic receptor	16
Parathyroid cell calcium-sensing receptor	52	Galectin-7	16
GR	52	Cyclooxygenase-2	16
ATP receptor	50	Ras-related C3 botulinum toxin substrate 1	16
Hemoglobin alpha-1 chain	50	T-cell protein-tyrosine phosphatase	16

US28	50	Choline acetylase	15
Beta globin chain	50	Matrix metalloproteinase 13	15
Muscarinic acetylcholine receptor M3	49	Dihydrofolate reductase	14
Ribonucleotide reductase small chain	49	Galectin-8	14
Adenosylcobalamin-dependent ribonucleoside-triphosphate reductase	49	Serotonin 7 (5-HT7) receptor	14
P2X4	49	Vitamin D receptor	14
Putative uncharacterized protein TRT1	49	3-dehydroquinate dehydratase	14
Ribonucleoside-diphosphate reductase 1 subunit alpha	49	Melanocortin receptor 5	14
Ribonucleoside-diphosphate reductase 2 subunit alpha	49	Peptide deformylase mitochondrial	14
Ribonucleotide reductase large subunit	49	Serine/threonine-protein kinase Chk2	14
Ribonucleotide reductase, B12-dependent	49	Serotonin (5-HT) receptor	14
TP2	49	Sialidase	14
Phosphatidylinositol 3-kinase p100 subunit	45	Thymidine kinase, cytosolic	14
c-Jun N-terminal kinase 1	45	Integrin alpha-4	13
NKR	44	Cytochrome P450 2D6	13
Janus kinase 1	44	Heat shock protein HSP 90-alpha	13
HIV-1 PR	41	Lipoxygenase	13
GPR-CY6	41	Sigma opioid receptor	13
Protein-tyrosine phosphatase 1B	40	Apoptosis regulator Bcl-2	12
Protein toll	40	Anandamide amidohydrolase	12
Free fatty acid activated receptor 2	38	Glycogen phosphorylase, muscle form	12
Integrin beta-1	38	Monoglyceride lipase	12
Abelson murine leukemia viral oncogene homolog 1	37	Autotaxin	12
RGS4	37	Casein kinase II alpha	12
Apoptosis regulator Bcl-2	36	Cathepsin B	12
SAMDC	36	Cathepsin L	12
Serpin C1	36	Gonadotropin-releasing hormone receptor	12
nPKC-epsilon	36	Kallikrein 5	12
Apoptosis inhibitor survivin	35	G protein-coupled receptor 44	11
CGS-PDE	35	Luciferin 4-monooxygenase	11
30S ribosomal protein S2	34	Alpha-chymotrypsin	11
A2 A adenosine receptor	33	Beta-lactamase	11

Cell surface glycoprotein MAC-1 subunit alpha	33	Botulinum neurotoxin type A	11
Neurotrophic tyrosine kinase receptor type 1	33	Calcitonin gene-related peptide 1	11
TER ATPase	33	Fibroblast activation protein alpha	11
CD11 antigen-like family member A	32	Nuclear receptor ROR-alpha	11
Nuclear receptor subfamily 3 group C member 4	32	Peptide deformylase	11
Ubiquitin carrier protein D3	31	Retinoic acid receptor beta	11
Thrombin light chain	30	Serine/threonine-protein kinase PIM1	11
GluK1	30	Tyrosinase	11
SR31747-binding protein	30	Alpha-L-fucosidase I	10
ET-A	28	Disks large homolog 4	10
Epoxide hydratase	28	Leukocyte elastase	10
Glucagon-like peptide 2 receptor	28	P-selectin	10
Toll-like receptor 9	28	Purine nucleoside phosphorylase	10
CD49 antigen-like family member E	26	Thymidine kinase	10
SMO	26	Tyrosine-protein kinase receptor FLT3	10
Thrombin receptor	26	Xanthine dehydrogenase	10
Flavin-containing amine oxidase domain-containing protein 2	24	Adenylate kinase 3 alpha like 1	10
Porphobilinogen synthase	24	Angiotensin II receptor	10
SL3/AKV core-binding factor alpha B subunit	24	CD209 antigen	10
UDP/CysLT receptor	24	Coagulation factor X	10
AChE	23	Glutathione reductase	10
Voltage-gated calcium channel subunit alpha Cav3.2	23	Histone-lysine N-methyltransferase, H3 lysine-79 specific	10
Aspartate carbamoyltransferase regulatory chain	22	Tyrosine-protein kinase Lyn	10
GSase	22	FK506-binding protein 1A	9
MC5-R	22	Thymidylate synthase	9
gp68	22	Growth factor receptor-bound protein 2	9
Plasmin	21	Histone deacetylase 3/Nuclear receptor corepressor 2 (HDAC3/NCoR2)	9

D-OR-1	21	Induced myeloid leukemia cell differentiation protein Mcl-1 homolog	9
GDH 2	21	Penicillin-binding protein 2a	9
PGE2 receptor EP2 subtype	20	Phosphodiesterase 4B	9
Ha-Ras	19	S-adenosylmethionine synthetase (MAT 1 and MAT 2)	9
Cannabinoid CB2 receptor	18	Thymidylate kinase	9
DNA-binding factor KBF1	18	Asialoglycoprotein receptor 1	8
Myosin II heavy chain	18	Baculoviral IAP repeat-containing protein 3	8
Nuclear factor of kappa light polypeptide gene enhancer in B-cells 3	18	Catechol O-methyltransferase	8
RNase P protein subunit	18	Cytochrome P450 3A4	8
TGase-2	18	Inositol 1,4,5-trisphosphate receptor type 1	8
Isocitric dehydrogenase subunit gamma	17	Phenylalanine-4-hydroxylase	8
P60-Src	17	Testis-specific androgen-binding protein	8
P94	17	Vesicular glutamate transporter 3	8
Beta-Ketoacyl-acyl carrier protein reductase	17	3-dehydroquinate dehydratase	8
Chloride channel Ka	17	Cytochrome P450 2C9	8
GABA-A receptor; anion channel	17	D-amino-acid oxidase	8
Glucose-1-phosphate thymidylyltransferase	17	Glycogen synthase kinase-3 alpha	8
Helicase with RNase motif	17	Histone deacetylase 10	8
IDH	17	Histone deacetylase 6	8
Isocitrate dehydrogenase [NAD] subunit alpha, mitochondrial	17	Histone deacetylase 8	8
Isocitrate dehydrogenase subunits 3/4	17	Integrin alpha-V/beta-5	8
Leukotriene A(4) hydrolase	17	Mitogen-activated protein kinase kinase kinase 5	8
NAD(+)-specific ICDH subunit beta	17	Serine/threonine-protein kinase Aurora-A	8
Tumor necrosis factor-alpha receptor	16	Serine/threonine-protein kinase Aurora-C	8
AtMEPCT	16	Cyclophilin A	7
CBF-beta	16	Adenylate kinase 2	7
Focal adhesion kinase 1	16	Leukotriene B4 receptor 1	7
NR2C	16	Mu opioid receptor	7
hsAC	16	Serotonin 2b (5-HT2b) receptor	7

Cu-NIR	15	Serotonin 6 (5-HT6) receptor	7
E-NPP 2	15	Acidic alpha-glucosidase	7
Glycine receptor 48 kDa subunit	15	Adrenergic receptor alpha-2	7
Glycine receptor 58 kDa subunit	15	Cruzipain	7
Hydroxylamine reductase	15	Cytosol aminopeptidase	7
MurI	15	Dual-specificity tyrosine-phosphorylation regulated kinase 1A	7
PEPCase 1	15	Histone deacetylase 11	7
Rd	15	Histone deacetylase 4	7
Scatter factor	15	Kallikrein 7	7
Estradiol receptor	14	Macrophage colony stimulating factor receptor	7
IKK-B	14	Neuronal acetylcholine receptor; alpha4/beta2	7
PEPCase	14	Protein tyrosine kinase 2 beta	7
Gelatinase B	14	Ribonuclease pancreatic	7
Malate dehydrogenase	14	Trypsin I	7
Malate dehydrogenase, decarboxylating	14	Vascular endothelial growth factor receptor 2	7
NADP-malic enzyme 2	14	Coagulation factor XI	6
SCFR	14	Adenylate kinase 1	6
Tryptophan synthase beta chain	14	Gamma-glutamyltranspeptidase 1	6
Voltage-gated potassium channel subunit Kv11.1	14	Glutamate [NMDA] receptor subunit epsilon 3	6
Calcium pump 1	13	Human immunodeficiency virus type 1 protease	6
Integrin alpha-4	13	Leukocyte adhesion molecule-1	6
SA	13	Prostanoid EP4 receptor	6
Tryptophan synthase alpha chain	13	Thermolysin	6
SPAAT	13	Adrenergic receptor alpha-1	6
CD49b	12	Cystine/glutamate transporter	6
GCS-beta-3	12	Furin	6
Huff	12	G-protein coupled bile acid receptor 1	6
Seed lipoxygenase-1	12	HLA class II histocompatibility antigen DRB3-1	6
CUL-5	12	Histone deacetylase 7	6
GC-C	12	Insulin receptor	6
GCS-alpha-2	12	L-lactate dehydrogenase A chain	6
Hemocyanin A chain	12	Leukotriene A4 hydrolase	6
LIMK-2	12	Ornithine decarboxylase	6

Threonine aspartase subunit alpha	12	Pyruvate kinase isozymes R/L	6
GPDH	11	Sphingosine kinase 1	6
Protein Yama	11	Tubulin	6
Aspartate carbamoyltransferase catalytic chain	11	Tyrosine-protein kinase JAK2	6
DNA deoxyribophosphodiesterase	11	p53-binding protein Mdm-2	6
Exonuclease I	11	Beta-galactosidase	5
Glyceraldehyde-3-phosphate dehydrogenase (NADP+)	11	Ghrelin receptor	5
Glyceraldehyde-3-phosphate dehydrogenase (Phosphorylating)	11	Neutral alpha-glucosidase AB	5
Monoamine oxidase type B	11	Prostanoid EP3 receptor	5
Nuclear receptor subfamily 2 group B member 1	11	Prostanoid IP receptor	5
ODC	11	Vanilloid receptor	5
Alpha-glucosidase I	10	WD repeat-containing protein 5	5
PLD	10	1-deoxy-D-xylulose-5-phosphate synthase	5
Coagulation factor XI	10	Adhesin protein fimH	5
PDC	10	Arachidonate 12-lipoxygenase	5
Phosphofructokinase-M	10	Beta amyloid A4 protein	5
OTCase	9	Beta-secretase 1	5
SSAO	9	C5a anaphylatoxin chemotactic receptor	5
TP	9	Cystinyl aminopeptidase	5
AK	9	Cytochrome P450 1A2	5
Beta-1 metal-binding globulin	9	Ephrin type-B receptor 2	5
Breast tumor-amplified kinase	9	Eukaryotic translation initation factor	5
CPT1-M	9	Glucagon receptor	5
CaMK-II subunit gamma	9	Glyoxalase I	5
Cathepsin O	9	Heat shock protein HSP 60	5
Cytochrome b-562	9	Indoleamine 2,3-dioxygenase	5
IP-10 receptor	9	Phosphodiesterase 10A	5
P2U purinoceptor 1	9	Phosphotyrosine-protein phosphatase PTPB	5
Serine/threonine protein kinase	9	Platelet-derived growth factor receptor beta	5
ATP-sulfurylase	8	Serine/threonine-protein kinase PLK1	5
CRF-R1	8	Tryptophan 2,3-dioxygenase	5
Deoxycytidylate aminohydrolase	8	Tyrosine-protein kinase ABL	5

UDP-GlcNAc-2-epimerase	8	Vitamin K-dependent protein C	5
VEGF Receptor 2	8	c-Jun N-terminal kinase 2	5
5-HT4	8	Seed lipoxygenase-1	4
ATP-PRT	8	Adenosine deaminase	4
Arachidonate 15-lipoxygenase B	8	DNA polymerase alpha subunit	4
Endoribonuclease	8	Diamine oxidase	4
GnRH-R	8	Dihydroorotate dehydrogenase	4
Phosphohexokinase	8	Kynureninase	4
cAMP-GEFII	8	Kynurenine 3-monooxygenase	4
CHK1	7	Neurotensin receptor 1	4
ICE-LAP3	7	Nitric oxide synthase, inducible	4
Na(+)/K(+) ATPase alpha-1 subunit	7	Prostanoid EP1 receptor	4
TS	7	Vascular endothelial growth factor receptor 1	4
1,4-alpha-D-glucan glucanohydrolase	7	2-dehydro-3-deoxyphosphooctonate aldolase	4
3-deoxy-D-arabino-heptulosonate 7-phosphate synthase	7	ALK tyrosine kinase receptor	4
ATP-dependent helicase E1	7	Aminopeptidase B	4
Caspase-6 subunit p11	7	Casein kinase I isoform alpha-like	4
G-protein coupled receptor 40	7	DNA-dependent protein kinase	4
GC-A	7	Discoidin domain-containing receptor 2	4
Inducible NOS	7	Dual specificity phosphatase Cdc25B	4
MOP-5	7	Glutamate receptor ionotropic, kainate	4
Na(+)/K(+) ATPase subunit gamma	7	Matrix metalloproteinase 14	4
Nuclear receptor subfamily 3 group C member 3	7	Max-like protein X	4
Penicillin binding protein 2 prime	7	Mycophenolic acid acyl-glucuronide esterase, mitochondrial	4
Sodium/potassium-dependent ATPase subunit beta-1	7	Neurotrophic tyrosine kinase receptor type 2	4
AMP deaminase isoform M	6	Nicotinamide phosphoribosyltransferase	4
ATP citrate synthase	6	Pantothenate synthetase	4
Insulin-like growth factor I receptor	6	Phosphodiesterase 7A	4
Lactose Repressor	6	Prolyl endopeptidase	4
Oxytocin-neurophysin 1	6	Protein kinase C epsilon	4

RNase P protein subunit DRpp40	6	Receptor protein-tyrosine kinase erbB-2	4
UGT1.1	6	Serine/threonine-protein kinase B-raf	4
Alpha-1C adrenergic receptor	6	Serotonin 1d (5-HT1d) receptor	4
Cyclic nucleotide-binding protein	6	Stem cell growth factor receptor	4
Dihydropteroate synthase	6	Subtilisin/kexin type 6	4
Estrogen synthase	6	Tyrosine-protein phosphatase yopH	4
Gamma-tubulin complex component 1	6	Uracil nucleotide/cysteinyl leukotriene receptor	4
Heterogeneous nuclear ribonucleoprotein methyltransferase-like protein 3	6	HMG-CoA reductase	3
Homoserine dehydrogenase	6	Adenosylhomocysteinase	3
Mitogen-activated protein kinase phosphatase 3	6	Angiotensin II type 2 (AT-2) receptor	3
Nicotinic acid receptor	6	Asparagine synthetase	3
Probable ribonuclease P/MRP protein subunit POP5	6	Bradykinin B2 receptor	3
RNase MRP protein subunit	6	Cyclin A2	3
Sensory neuron-specific G-protein coupled receptor 1	6	DNA topoisomerase I	3
Transformation-related protein 53	6	Folylpoly-gamma-glutamate synthetase	3
AE 1	5	Glutathione S-transferase A1	3
Beta-Carbonic Anhydrase	5	Hexokinase	3
Cytosolic 5'-nucleotidase II	5	Low molecular weight phosphotyrosine protein phosphatase	3
Glutamate [NMDA] receptor subunit epsilon 4	5	Selectin E	3
Phosphoglycerate dehydrogenase	5	Succinyl-diaminopimelate desuccinylase	3
Pur regulon repressor	5	Tubulin alpha chain	3
Pyruvate dehydrogenase, lipoamide, kinase isozyme 2, mitochondrial	5	UDP-glucuronosyltransferase 2B7	3
TEM-5	5	Vitamin D-binding protein	3
Threonine dehydratase	5	Activin receptor type-1B	3
1,25-dihydroxyvitamin D3 receptor	5	Apoptosis regulator Bcl-W	3
Exchange protein directly activated by cAMP 1	5	B-cell receptor CD22	3

G protein-coupled receptor kinase GRK6	5	Bcl-2-related protein A1	3
GABA(C) receptor	5	Bifunctional protein glmU	3
GLIC	5	Carbepenem-hydrolyzing beta-lactamase KPC	3
ICE-like apoptotic protease 6	5	Casein kinase I gamma 2	3
IMP dehydrogenase 1	5	Chymotrypsin C	3
PDT	5	Coagulation factor VII	3
PGDH	5	DNA topoisomerase II alpha	3
Solute carrier family 5 member 7	5	Dual specificity tyrosine-phosphorylation-regulated kinase 1B	3
nPKC-zeta	5	Enoyl-[acyl-carrier-protein] reductase [NADH]	3
p75	5	Inosine-5'-monophosphate dehydrogenase, probable	3
6-DEB hydroxylase	4	MAP kinase-activated protein kinase 2	3
ADP-glucose synthase	4	Macrophage migration inhibitory factor	3
Biotin--acetyl-CoA-carboxylase ligase	4	Mannosidase 2 alpha 1	3
CGMP-binding cGMP-specific phosphodiesterase	4	Monoacylglycerol lipase ABHD6	3
Carbamoyl-phosphate synthetase ammonia chain	4	Penicillin-binding protein 2	3
ER-beta	4	Phospho-2-dehydro-3-deoxyheptonate aldolase	3
PPDC	4	Probable low molecular weight protein-tyrosine-phosphatase	3
Phosphatidylcholine 2-acylhydrolase 1B	4	Prostaglandin E synthase	3
Type II Citrate Synthases	4	Proteasome Macropain subunit MB1	3
UDP-N-acetylglucosamine pyrophosphorylase, putative	4	Protein-arginine N-methyltransferase 1	3
UPRTase	4	Pyridoxal kinase	3
cGMP-dependent protein kinase I	4	Quinone reductase 2	3
1.3.1.20</ecNumber	4	Riboflavin-binding protein	3
235aa long hypothetical biotin--[acetyl-CoA-carboxylase] ligase	4	SHC-transforming protein 1	3
AGPase B	4	Serine/threonine-protein kinase RAF	3

ATP-PRTase	4	Signal transducer and activator of transcription 3	3
Acute-phase response factor	4	Sn1-specific diacylglycerol lipase beta	3
Adenylyl cyclase	4	Sphingosine kinase 2	3
Alpha-2AAR subtype C10	4	Sucrase-isomaltase	3
Bacteriophage N4 adsorption protein C	4	TRAIL receptor-1	3
Beta-thionase	4	Terminal deoxynucleotidyltransferase	3
Biotin repressor	4	Trace amine-associated receptor 1	3
Burkitt lymphoma receptor 1	4	Tyrosine-protein kinase ITK/TSK	3
CXXC-type zinc finger protein 9	4	Urease	3
Carbamoyl-phosphate synthetase glutamine chain	4	Uridine phosphorylase 1	3
Cytochrome P450-J	4	Vasopressin V2 receptor	3
DAHP synthase	4	Voltage-gated L-type calcium channel	3
FADD-like ICE	4	Epoxide hydratase	2
KDC	4	Alpha-1d adrenergic receptor	2
Lysis protein	4	Beta-3 adrenergic receptor	2
Methionyl-tRNA synthetase	4	Equilibrative nucleoside transporter 1	2
N-acetylglucosamine-1-phosphate uridyltransferase	4	Glyoxalase II	2
N-end-recognizing protein	4	Myelin-associated glycoprotein	2
PAS domain-containing protein 2	4	Neuropilin-1	2
PTH/PTHrP type I receptor	4	Orotidine 5'-phosphate decarboxylase	2
Prohormone convertase	4	Orotidine phosphate decarboxylase	2
Putative deoxycytidylate deaminase	4	Penicillin-binding protein 1A	2
Regulatory protein SIR2 homolog 3	4	Penicillin-binding protein 2B	2
Ribose-phosphate pyrophosphokinase 1	4	Penicillin-binding protein 2x	2
Stuart factor	4	Serotonin 5a (5-HT5a) receptor	2
TIS10 protein	4	Vasoactive intestinal peptide receptor	2
Trypanothione synthetase, putative	4	Vasopressin V1a receptor	2
Tyrosine 3-hydroxylase	4	1-aminocyclopropane-1-carboxylate synthase 5	2
UDP-glucose 6-dehydrogenase	4	5-enolpyruvylshikimate-3-phosphate synthase	2
UDPGDH	4	6-phospho-1-fructokinase	2
CAP	3	ATP-citrate synthase	2

CK II beta	3	Acetylcholine receptor protein delta chain	2
CaM	3	Aldehyde dehydrogenase	2
Cytochrome c oxidase subunit 2	3	Alpha-glucosidase	2
Cytosine aminohydrolase	3	Anthrax lethal factor	2
D-alanyl-D-alanine carboxypeptidase	3	Beta-hexosaminidase	2
ICE	3	Beta-hexosaminidase subunit beta	2
Kinesin-related protein CENPE	3	Beta-lactamase type II	2
Orphan nuclear receptor PXR	3	Bone morphogenetic protein receptor type-2	2
PGF receptor	3	Bradykinin B1 receptor	2
RNA polymerase subunit beta	3	Breast cancer type 1 susceptibility protein	2
Serotonin receptor 7	3	C-C motif chemokine 2	2
Tryptamin 2,3-dioxygenase	3	Catenin beta-1	2
4,5-PCD	3	Chitinase	2
AKIII	3	Cyclin-dependent kinase 5	2
Aspartate kinase	3	D-alanine--D-alanine ligase	2
Aspartate kinase 1	3	D-alanylalanine synthetase	2
CASP-2	3	Dipeptidyl peptidase VIII	2
CCK2-R	3	Dual specificity mitogen-activated protein kinase kinase 1	2
CK II alpha	3	Dual specificity protein kinase CLK2	2
Cytochrome c oxidase subunit 1	3	Dual specificty protein kinase CLK1	2
Cytochrome c oxidase subunit 3	3	Fibroblast growth factor receptor 2	2
Cytochrome c oxidase subunit 4 isoform 1, mitochondrial	3	FkbO	2
DHEA-ST	3	Hematopoietic cell protein-tyrosine phosphatase 70Z-PEP	2
DNA-directed RNA polymerase subunit alpha	3	Histone-lysine N-methyltransferase, H3 lysine-9 specific 3	2
Diguanylate kinase	3	Inhibitor of nuclear factor kappa B kinase alpha subunit	2
EK	3	Isoprenylcysteine carboxyl methyltransferase	2
Fructose-bisphosphate aldolase A	3	Lysine-specific demethylase 2A	2
G-protein coupled receptor HG11	3	MAP kinase signal-integrating kinase 2	2

HD2	3	MAP/microtubule affinity-regulating kinase 2	2
IMP--aspartate ligase	3	Matrix metalloproteinase 7	2
Inflammation-related G-protein coupled receptor EX33	3	Matrix metalloproteinase 9	2
Insulysin	3	Mitogen-activated protein kinase kinase kinase 1	2
M-calpain	3	Myosin light chain kinase family member 4	2
MIF	3	N(G),N(G)-dimethylarginine dimethylaminohydrolase 1	2
Mevalonate kinase	3	Neuropeptide FF receptor 1	2
NS3 protein	3	Neurotensin receptor	2
PCB	3	PI3-kinase p110-gamma subunit	2
PP-1A	3	Peptidyl-prolyl cis-trans isomerase NIMA-interacting 1	2
PPAR-gamma	3	Phosphodiesterase 11A	2
PelD	3	Phosphodiesterase 1A	2
RNAP subunit alpha	3	Phosphodiesterase 2A	2
RNAP subunit beta	3	Phosphodiesterase 3A	2
Receptor-interacting protein 1	3	Phosphodiesterase 5A	2
SK 1	3	Phosphodiesterase 8A	2
SPCA	3	Phosphodiesterase 9A	2
Serpin E1	3	Prostanoid FP receptor	2
Sodium-dependent glutamate/aspartate transporter 1	3	Proteasome component C5	2
Steroid Delta-isomerase	3	Protein kinase C theta	2
TRPO	3	Ribosomal protein S6 kinase alpha 6	2
Transmembrane protein 142A	3	RocR	2
Tryptase II	3	S-ribosylhomocysteine lyase	2
Tryptophan oxygenase	3	Serine/threonine-protein kinase DCLK2	2
U-PAR	3	Serine/threonine-protein kinase GAK	2
Uridylate kinase	3	Serine/threonine-protein kinase PAK 1	2
Voltage-gated calcium channel subunit alpha Cav1.2	3	Serine/threonine-protein kinase PLK4	2
hGPCR33	3	Somatostatin receptor	2
2-oxoglutarate dehydrogenase, mitochondrial	2	Squalene synthetase	2

6-phosphofructokinase II	2	Tumour suppressor p53/oncoprotein Mdm2	2
AAA+ ClpX hexamers	2	Tyrosine-protein kinase receptor RET	2
ATP-binding cassette sub-family B member 1	2	Tyrosine-protein kinase receptor Tie-1	2
Alpha-glucan phosphorylase	2	Tyrosine-protein kinase receptor UFO	2
Aspartate beta-decarboxylase	2	Urokinase-type plasminogen activator	2
Beta2-adrenoceptor	2	DNA polymerase beta	1
CFTR	2	4-hydroxyphenylpyruvate dioxygenase	1
CM	2	Atrial natriuretic peptide receptor C	1
Chitin synthase 2	2	Beta-xylosidase	1
Chloroplastic	2	Excitatory amino acid transporter 2	1
DegS	2	Fucosyltransferase 5	1
Dopamine D4 receptor	2	Galanin receptor 1	1
Glutamine phosphoribosylpyrophosphate amidotransferase	2	Galanin receptor 2	1
GroEL	2	Glutamate dehydrogenase	1
Guanase	2	Glycerol kinase	1
Hexokinase I	2	Heparanase	1
Hydroxymethylbilane synthase	2	Inosine-5'-monophosphate dehydrogenase 2	1
Influenza A M2 channel	2	Lactase-glycosylceramidase	1
L-LDH	2	Lanosterol synthase	1
N(1),N(8)-bis(glutathionyl)spermidine reductase	2	Nitric-oxide synthase, brain	1
N-WASP	2	Papain	1
NAGS	2	Phenylethanolamine N-methyltransferase	1
Oligomycin sensitivity conferral protein	2	Solute carrier family 22 member 12	1
P26	2	1-phosphatidylinositol 3-phosphate 5-kinase	1
Phosphatidylinositol-specific phospholipase C	2	3 beta-hydroxysteroid dehydrogenase/Delta 5-->4-isomerase	1

Phosphopentokinase 1	2	3-oxoacyl-acyl-carrier protein reductase	1
Phosphorylase kinase subunit gamma 1	2	6-phosphofructo-2-kinase/fructose-2,6-bisphosphatase 3	1
Prephenate dehydratase	2	ATP-dependent RNA helicase DDX3X	1
PriA helicase	2	Adaptor-associated kinase	1
RNA Polymerase	2	Aldehyde dehydrogenase dimeric NADP-preferring	1
RTX	2	Alpha-galactosidase	1
RecA	2	Alpha-galactosidase A	1
Srr	2	Alpha-glucosidase MAL62	1
Type II IPP	2	Aminoacyl-tRNA synthetase	1
hPanK1	2	Aryl hydrocarbon receptor	1
11R-lipoxygenase	2	BMP-2-inducible protein kinase	1
ATP-binding cassette, sub-family B, member 1	2	Baculoviral IAP repeat-containing protein 7	1
Abelson murine leukemia viral oncogene homolog 2	2	Beta-lactamase OXA-10	1
Allantoate deiminase	2	Bifunctional protein NCOAT	1
Amino acid biosynthesis regulatory protein	2	Bile salt export pump	1
AtCM1	2	Bombesin receptor subtype-3	1
Beta-galactoside-binding lectin L-14-I	2	Bromodomain adjacent to zinc finger domain protein 2B	1
Branched-chain alpha-ketoacid dehydrogenase kinase	2	Bromodomain-containing protein 2	1
CXC-R4	2	Bromodomain-containing protein 4	1
Chitin synthase 1	2	C-terminal processing protease of the D1 protein	1
Chitin-UDP acetyl-glucosaminyl transferase 3	2	C-type lectin domain family 7 member A	1
Cholesterol acyltransferase 1	2	CaM-kinase kinase beta	1
DHDPS	2	Carboxylesterase 1D	1
DHOase	2	Carnitine palmitoyltransferase 2	1
DNA double-strand break repair Rad50 ATPase	2	Casein kinase II beta	1
DNA-binding protein VF1	2	Cell division cycle 2-like protein kinase 6	1
Dihydrolipoyllysine-residue succinyltransferase component of 2-oxoglutarate dehydrogenase complex	2	Chaperone activity of bc1 complex-like, mitochondrial	1

Dihydroorotase	2	Citron Rho-interacting kinase	1
Ecto-5'-nucleotidase	2	Cyclin-dependent kinase 7	1
Env polyprotein	2	Cyclin-dependent kinase 9	1
FimX	2	Cyclin-dependent kinase-like 1	1
GMD	2	Cytochrome P450 11A1	1
GPATase	2	D-alanyl-D-alanine dipeptidase	1
Glycine hydroxymethyltransferase	2	DNA polymerase kappa	1
HD domain protein	2	Dual specificity protein kinase CLK3	1
Hexosephosphate aminotransferase	2	Dual-specificity tyrosine-phosphorylation regulated kinase 2	1
K-sam	2	EZH2/SUZ12/EED/RBBP7/RBBP4	1
Lactase	2	Endo-beta-N-acetylglucosaminidase	1
MGS	2	Endoplasmin	1
Mitochondrial oligopeptidase M	2	Enterobactin synthetase component E	1
Mitoxantrone resistance-associated protein	2	Eukaryotic translation initiation factor 2-alpha kinase 4	1
N-acetylglutamate kinase / N-acetylglutamate synthase	2	FK506 binding protein 12	1
Oleoyl-[acyl-carrier-protein] hydrolase	2	Fatty acid synthase	1
Oxoglutarate dehydrogenase (Succinyl-transferring), E1 component	2	Fe(3+)-Zn(2+) purple acid phosphatase	1
PHF-tau	2	G-protein coupled receptor 35	1
PKC-A	2	Glucosamine--fructose-6-phosphate aminotransferase [isomerizing] 1	1
PNPase	2	Glucose-6-phosphate translocase	1
Phosphorylase b kinase regulatory subunit alpha, skeletal muscle isoform	2	Glutamine synthetase	1
Phosphorylase kinase beta-subunit	2	Glutathione S-transferase Mu 1	1
Polyprotein	2	Glyceraldehyde-3-phosphate dehydrogenase, glycosomal	1
Prephenate dehydrogenase	2	Guanine deaminase	1
Protein IPGM-1, isoform a	2	Heat shock protein 75 kDa, mitochondrial	1
Protein phosphatase type 1A (formely 2C) Mg-dependent alpha isoform	2	Hepatitis C virus serine protease, NS3/NS4A	1
Putative uncharacterized protein PH0207	2	Histidase	1

RAR-epsilon	2	Histo-blood group ABO system transferase	1
RXR-interacting protein 14	2	Histone acetyltransferase KAT5	1
Recessive suppressor of secretory defect	2	Histone acetyltransferase PCAF	1
Regulatory protein SIR2	2	Histone acetyltransferase p300	1
Rhodobacter sphaeroides 2.4.1 chromosome 1, complete sequence	2	Histone deacetylase-like amidohydrolase	1
Rts protein	2	Histone-lysine N-methyltransferase, H3 lysine-9 specific 5	1
SH2 domain-containing inositol 5'-phosphatase 1	2	Homeodomain-interacting protein kinase 1	1
Ste2p	2	Homeodomain-interacting protein kinase 3	1
Sulfonylurea receptor 2	2	Homeodomain-interacting protein kinase 4	1
Toxin B	2	Hormonally up-regulated neu tumor-associated kinase	1
UPRT	2	Human immunodeficiency virus type 1 reverse transcriptase	1
Uncharacterized protein	2	IGF-like family receptor 1	1
Voltage-gated calcium channel subunit alpha Cav3.3	2	IgG receptor FcRn large subunit p51	1
WASp	2	Interferon-induced, double-stranded RNA-activated protein kinase	1
eIF-4E	2	Interleukin-1 receptor-associated kinase 3	1
nPKC-delta	2	Interleukin-2 receptor alpha chain	1
p72-Syk	2	Intestinal alkaline phosphatase	1
1-phosphatidylinositol-4,5-bisphosphate phosphodiesterase gamma 1	1	Kelch-like ECH-associated protein 1	1
2'-5'-oligoisoadenylate synthetase 1	1	Low affinity neurotrophin receptor p75NTR	1
3-hydroxybutyrate dehydrogenase	1	Lymphocyte differentiation antigen CD38	1
ALS-III	1	Lysine-specific histone demethylase 1	1
AP-TNAP	1	Lysophosphatidic acid receptor Edg-2	1

Adenylate nucleosidase	1	Mandelate racemase	1
Alpha-2CAR	1	Membrane-associated phosphatidylinositol transfer protein 1	1
Alpha-isopropylmalate synthase	1	Methylenetetrahydrofolate dehydrogenase	1
Aspartokinase I/homoserine dehydrogenase I	1	Mitogen-activated protein kinase 15	1
CTP synthetase	1	Mitogen-activated protein kinase 7	1
Complex III subunit III	1	Mitogen-activated protein kinase kinase kinase 10	1
Cyclophilin A	1	Mitogen-activated protein kinase kinase kinase 4	1
D-alanine:D-alanine ligase	1	Mitogen-activated protein kinase kinase kinase 7	1
DNA helicase Rep and single-stranded DNA-dependent ATPase	1	Mixed lineage kinase 7	1
Deadenylating nuclease	1	Myosin light chain kinase	1
G6PD	1	N-arachidonyl glycine receptor	1
GNPDA 1	1	N-lysine methyltransferase SMYD2	1
GPD-C	1	NAD-dependent deacetylase sirtuin 1	1
Gamma-glutamate kinase	1	NUAK family SNF1-like kinase 1	1
Glutamate carboxypeptidase-like protein 2	1	Neutral cholesterol ester hydrolase 1	1
Glutamine amido-transferase	1	O-acetylserine sulfhydrylase	1
Glutamine amidotransferase:cyclase	1	Oxytocin receptor	1
Glycerokinase	1	P-hydroxybenzoate hydroxylase	1
HMG-CoA reductase	1	P2X purinoceptor 4	1
HSCARG	1	Peptide N-myristoyltransferase 1	1
Iron(III) dicitrate transport protein fecA	1	Peptidyl-prolyl cis-trans isomerase D	1
L-asparaginase I	1	Phosphatidylinositol-4-phosphate 3-kinase C2 domain-containing beta polypeptide	1
LivG	1	Phosphatidylinositol-4-phosphate 5-kinase type-1 alpha	1
M.EcoDam	1	Phosphatidylinositol-5-phosphate 4-kinase type-2 beta	1
MetN	1	Phospho-N-acetylmuramoyl-pentapeptide-transferase	1

Mitochondrial uncoupling protein	1	Plasmepsin 2	1
Myosin-Va	1	Platelet-activating factor acetylhydrolase	1
NAD(P)(+) transhydrogenase [B-specific]	1	Pregnane X receptor	1
NADH:nitrate reductase	1	Proline racemase	1
OAS-TL A	1	Proteasome Macropain subunit	1
PTE	1	Protein VAC14 homolog	1
Parkingson disease protein 7	1	Protein farnesyltransferase	1
Protein Dhm1	1	Protein kinase C alpha	1
Pyrophosphate-dependent 6-phosphofructose-1-kinase	1	Protein-tyrosine phosphatase 2C	1
RBP	1	Protein-tyrosine phosphatase LC-PTP	1
Rab GG transferase alpha	1	S-methylmethionine--homocysteine S-methyltransferase BHMT2	1
RecA, E. coli, homolog of	1	Serine hydroxymethyltransferase, cytosolic	1
RuvB Protein	1	Serine-protein kinase ATR	1
S-Rnase	1	Serine/threonine-protein kinase 10	1
Teichoic acid biosynthesis protein D	1	Serine/threonine-protein kinase 11	1
Troponin I, cardiac muscle	1	Serine/threonine-protein kinase 17A	1
TrpRS	1	Serine/threonine-protein kinase 2	1
Tryptophan RNA-binding attenuator protein	1	Serine/threonine-protein kinase 32B	1
(p)ppGpp synthase	1	Serine/threonine-protein kinase LATS1	1
2.7.11.1</ecNumber	1	Serine/threonine-protein kinase MAK	1
5'-phosphoribosylglycinamide transformylase	1	Serine/threonine-protein kinase MST2	1
5-HT-1B	1	Serine/threonine-protein kinase NEK2	1
6-HDNO	1	Serine/threonine-protein kinase PAK 4	1
67 kDa protein	1	Serine/threonine-protein kinase PCTAIRE-3	1
76 kDa lysosomal alpha-glucosidase	1	Serine/threonine-protein kinase PLK3	1

ACAT-2	1	Serine/threonine-protein kinase RIO1	1
ACR-20, isoform a	1	Serine/threonine-protein kinase RIO2	1
ADP-ribosylation factors guanine nucleotide-exchange protein 100	1	Serine/threonine-protein kinase RIO3	1
AGK	1	Serine/threonine-protein kinase SRPK3	1
AHAS-I	1	Sphingosine 1-phosphate receptor Edg-1	1
AHAS-II	1	Squalene synthase	1
ALDH class 2	1	Steryl-sulfatase	1
ARF1-directed GTPase-activating protein	1	Sulfate anion transporter 1	1
ATP-dependent protease La	1	TRAF2- and NCK-interacting kinase	1
Acetohydroxy-acid synthase II small subunit	1	Telomerase reverse transcriptase	1
Acetokinase	1	Testis-specific serine/threonine-protein kinase 1	1
Alpha-NaCH	1	Thymidine kinase, mitochondrial	1
Amplified in liver cancer protein 1	1	Thymidine phosphorylase	1
Annexin V	1	Transient receptor potential cation channel subfamily M member 8	1
Anthranilate phosphoribosyltransferase	1	Tyrosine- and threonine-specific cdc2-inhibitory kinase	1
Anthranilate synthase component I	1	Tyrosine-protein kinase ABL2	1
Aurone synthase	1	Tyrosine-protein kinase FER	1
AvrA	1	Tyrosine-protein kinase SYK	1
Beta-chimerin	1	Tyrosine-protein phosphatase non-receptor type 9	1
Bifunctional phenylalanine ammonia-lyase	1	Tyrosyl-DNA phosphodiesterase 1	1
Brain-liver-intestine amiloride-sensitive Na(+) channel	1	UDP-N-acetylmuramoylalanine--D-glutamate ligase	1
C5a-R	1	Ubiquitin carboxyl-terminal hydrolase 5	1
CKI-alpha	1	Vascular endothelial growth factor receptor 3	1
CPSase I	1	Vasopressin V1b receptor	1
CXCR-7	1	Vesicular acetylcholine transporter	1
Carbon catabolite protein A	1		

Caspase-5 subunit p10	1		
Catechol oxidase	1		
Cav3.1c	1		
Chaperone protein MSI3	1		
Chymotrypsin-C	1		
Chymotrypsinogen B2	1		
Coagulation factor IXa light chain	1		
Complement C3b alpha' chain	1		
CooA protein	1		
Copper-sensitive operon repressor	1		
Cytosolic IMP-GMP specific 5'-nucleotidase	1		
D-methionine transport system permease protein metI	1		
DNA topoisomerase II	1		
Dimeric hemoglobin	1		
Dipeptidyl peptidase-like protein 9	1		
E3 ubiquitin ligase complex SCF subunit CDC4	1		
EF-G	1		
ELIC	1		
Ecto-NAD+ glycohydrolase	1		
GALR-2	1		
GMP-PDE gamma	1		
GSH-S	1		
GST class-pi	1		
Galactose operon repressor	1		
Galectin	1		
Gamma-ENaC	1		
Gibberellin-insensitive dwarf protein 1	1		
Glycerol dehydrase beta subunit	1		
Glycerol dehydrase gamma subunit	1		
Glycerol dehydratase large subunit	1		
Guanine nucleotide-binding protein alpha-q	1		
HBP23	1		
HD	1		
HD4	1		
HDC	1		
HGF activator	1		
HL-60 PAD	1		
HM63	1		

HPr kinase/phosphatase	1		
HPrK/P	1		
Heat shock protein HslU	1		
Hypothetical phosphoserine phosphatase	1		
ICD1	1		
IDH kinase/phosphatase	1		
IL-2	1		
IRK-1	1		
ImGP synthase subunit hisF	1		
Intracellular protease I	1		
Isopropylmalate/homocitrate/citramalate synthase	1		
Kinesin-related protein HSET	1		
L-serine dehydratase (Iron, sulfur-dependent)	1		
Lasalocid biosynthesis protein Lsd19	1		
MALT lymphoma-associated translocation	1		
MHC class II antigen DRA	1		
MHCK-A	1		
Metacaspase MCA2	1		
MutT/nudix family protein	1		
Mutated in multiple advanced cancers 1	1		
Myeloproliferative leukemia protein	1		
Myosin heavy chain 7	1		
N-acetyl-L-glutamate 5-phosphotransferase	1		
NAGSA dehydrogenase	1		
Na(+)/Ca(2+)-exchange protein 1	1		
Na(+)/PI cotransporter 1	1		
Nuclear matrix protein 265	1		
Nuclear receptor subfamily 1 group A member 1	1		
OTRPC1	1		
Oligosaccharyltransferase	1		
P-element transposase	1		
PGE2 receptor EP4 subtype	1		
PI3Kalpha	1		
PLC-delta-1	1		
PPIase FKBP1A	1		

Phospholipase A2	1		
PilZ domain protein	1		
Platelet membrane glycoprotein IIb	1		
Psi-conotoxin P3.8	1		
Pyrophosphate--fructose 6-phosphate 1-phosphotransferase subunit beta	1		
QAPRTase	1		
Rab geranylgeranyl transferase componenet, subunit beta	1		
Regulatory protein SIR2 homolog 2	1		
SAPKK1	1		
SK2	1		
STE20-like kinase MST	1		
Selenocysteine lyase	1		
Shaw2	1		
Spermidine n1-acetyltransferase	1		
StyR	1		
T4-binding globulin	1		
Transcortin	1		
Transcriptional regulator, MarR family	1		
Ubiquitin-conjugating enzyme E2-CDC34	1		
VAChT	1		
cAMP and cGMP phosphodiesterase 11A	1		
cGMP phosphodiesterase 6C	1		
hMSH2	1		
hPanK2	1		
hPanK3	1		
pMMO-H alpha subunit	1		

Table D-2. Medians (95%ci) of the 37 physicochemical properties for the original 2012 dataset. Numbers in bold are statistically significant differences between allosteric and competitive compounds. The list is ordered by largest differences. Red properties do not repeat in 2015 data. The analysis is done with center of clusters.

Clustering level	**60%/0.6**		**75%/0.75**		**90%/0.9**		**100%/1.0**	
Descriptors	**Allosteric**	**Competitive**	**Allosteric**	**Competitive**	**Allosteric**	**Competitive**	**Allosteric**	**Competitive**
chiral	**0(±0)**	**1(±0)**	**0(±0)**	**1(±0)**	**0(±0)**	**1(±0)**	**0(±0)**	**1(±0)**
chiral/HA	**0(±0)**	**0.0476(±0.005)**	**0(±0)**	**0.0400(±0.0035)**	**0(±0)**	**0.0400(±0.0035)**	**0(±0)**	**0.0435(±0.002)**
SlogP	**3.26(±0.09)**	**1.92(±0.23)**	**3.58(±0.05)**	**2.14(±0.16)**	**3.70(±0.04)**	**2.24(±0.1)**	**3.69(±0.04)**	**2.07(±0.13)**
b_ar/b_count	**0.30(±0.01)**	**0.21(±0.01)**	**0.310(±0.004)**	**0.222(±0.004)**	**0.309(±0.003)**	**0.214(±0.004)**	**0.309(±0.001)**	**0.214(±0.007)**
b_single/HA	**1.17(±0.01)**	**1.50(±0.04)**	**1.17(±0.01)**	**1.50(±0.02)**	**1.17(±0.01)**	**1.52(±0.01)**	**1.18(±0.01)**	**1.53(±0.01)**
b_single	**29(±1)**	**36(±2)**	**30(±0)**	**38(±1)**	**32(±0)**	**41(±1)**	**32(±1)**	**41(±1)**
b_ar/HA	**0.52(±0.01)**	**0.41(±0.01)**	**0.548(±0.003)**	**0.444(±0.016)**	**0.545(±0.003)**	**0.429(±0.003)**	**0.5484(±0.0029)**	**0.4286(±0)**
a_aro/HA	**0.51(±0.01)**	**0.41(±0.01)**	**0.54(±0.01)**	**0.43(±0.01)**	**0.55(±0.01)**	**0.43(±0.01)**	**0.55(±0.01)**	**0.42(±0.01)**
b_1rotN/HA	**0.174(±0.003)**	**0.200(±0.008)**	**0.172(±0.001)**	**0.192(±0.008)**	**0.172(±0.001)**	**0.200(±0.006)**	**0.172(±0.001)**	**0.211(±0.004)**
b_single/b_count	**0.67(±0.01)**	**0.76(±0.01)**	**0.654(±0.003)**	**0.758(±0.004)**	**0.655(±0.002)**	**0.763(±0.003)**	**0.655(±0.002)**	**0.764(±0.004)**
logS	**-4.63(±0.1)**	**-4.14(±0.14)**	**-4.99(±0.05)**	**-4.40(±0.08)**	**-5.25(±0.05)**	**-4.61(±0.07)**	**-5.32(±0.04)**	**-4.67(±0.06)**
b_count/HA	**1.80(±0.02)**	**1.98(±0.03)**	**1.80(±0.01)**	**1.97(±0.03)**	**1.81(±0.01)**	**2.00(±0.03)**	**1.81(±0.01)**	**1.98(±0.02)**
a_acc/HA	**0.107(±0.004)**	**0.097(±0.003)**	**0.107(±0.002)**	**0.091(±0.004)**	**0.107(±0.004)**	**0.095(±0.002)**	**0.107(±0.004)**	**0.097(±0.003)**
b_count	**43(±1)**	**47(±2)**	**46(±1)**	**50(±1)**	**49(±1)**	**54(±1)**	**49(±0)**	**55(±1)**
a_nC/HA	**0.739(±0.002)**	**0.762(±0.011)**	**0.75(±0.01)**	**0.77(±0.01)**	**0.750(±0.004)**	**0.767(±0.006)**	0.7500(±0)	0.7561(±0.0061)
b_1rotN	**4(±0)**	**5(±0)**	**4(±0)**	**5(±0)**	5(±0)	5(±0)	5(±0)	5(±0)
a_base	0(±0)	1(±1)	**0(±0)**	**1(±0)**	**0(±0)**	**1(±0)**	**0(±0)**	**1(±0)**
a_base/HA	0.0000(±0)	0.0208(±0.0208)	**0.0000(±0)**	**0.0244(±0.004)**	**0.0000(±0)**	**0.0222(±0.0094)**	**0.0000(±0)**	**0.0250(±0.002)**
a_aro	12(±0)	11(±1)	12(±0)	12(±0)	**14(±1)**	**12(±0)**	**15(±1)**	**12(±0)**
b_ar	12(±0)	11(±1)	12(±0)	12(±0)	**15(±1)**	**12(±0)**	**16(±1)**	**12(±0)**
a_don/HA	0.037(±0.001)	0.040(±0.004)	0.036(±0.001)	0.034(±0.003)	0.0345(±0)	0.0345(±0.0012)	0.034(±0.001)	0.034(±0.003)
b_rotN/HA	0.20(±0.01)	0.21(±0.01)	0.2(±0.0)	0.2083(±0.0083)	**0.2(±0.0)**	**0.2174(±0.0031)**	**0.2(±0.0)**	**0.2258(±0.0036)**
a_nC	18(±1)	19(±1)	19(±0)	20(±1)	**20(±0)**	**21(±0)**	**20(±0)**	**21(±0)**
b_1rotN/b_count	0.098(±0.002)	0.103(±0.003)	0.097(±0.001)	0.100(±0.002)	**0.098(±0.001)**	**0.103(±0.002)**	**0.097(±0.001)**	**0.106(±0.003)**
b_rotN/b_count	0.113(±0.003)	0.109(±0.006)	**0.111(±0.003)**	**0.105(±0.002)**	0.111(±0.002)	0.111(±0.003)	**0.111(±0.002)**	**0.118(±0.003)**
a_acc	2(±1)	2(±0)	**3(±0)**	**2(±0)**	3(±0)	3(±0)	3(±0)	3(±0)
a_acid	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
a_acid/HA	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
a_don	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
a_heavy	24(±0)	24(±1)	26(±0)	26(±0)	27(±0)	27(±1)	27(±0)	28(±1)
b_rotN	5(±0)	5(±0)	5(±0)	5(±0)	6(±1)	5(±1)	6(±1)	6(±1)
FCharge	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
FCharge/HA	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
lip_druglike	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
lip_violation	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
opr_leadlike	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
opr_violation	0(±0)	0(±1)	0(±0)	0(±0)	0(±0)	1(±0)	0(±0)	1(±0)
rings	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)

Table D-3. Medians (95%ci) of the 37 physicochemical properties for the original 2015 dataset. Numbers in bold are statistically significant differences between allosteric and competitive compounds. The list is ordered by largest differences. Red properties do not repeat in 2012 data. The analysis is done with center of clusters.

Clustering level	**60%/0.6**		**75%/0.75**		**90%/0.9**		**100%/1.0**	
Descriptors	**Allosteric**	**Competitive**	**Allosteric**	**Competitive**	**Allosteric**	**Competitive**	**Allosteric**	**Competitive**
a_don/HA	**0.037(±0.001)**	**0.045(±0.002)**	**0.0345(±0)**	**0.0435(±0.0018)**	**0.0323(±0)**	**0.0417(±0.0018)**	0.031(±0.001)	0.042(±0.002)
b_single	**29(±1)**	**34(±1)**	**31(±0)**	**36(±1)**	**34(±0)**	**39(±1)**	**34(±1)**	**39(±0)**
SlogP	**3.19(±0.03)**	**2.67(±0.07)**	**3.40(±0.01)**	**2.88(±0.04)**	**3.53(±0.01)**	**2.95(±0.04)**	**3.56(±0.01)**	**2.90(±0.04)**
b_single/HA	**1.22(±0.01)**	**1.38(±0.02)**	**1.20(±0.01)**	**1.38(±0.01)**	**1.20(±0.01)**	**1.40(±0.01)**	**1.212(±0.002)**	**1.421(±0.008)**
b_ar/b_count	**0.283(±0.004)**	**0.250(±0.006)**	**0.293(±0.001)**	**0.254(±0.004)**	**0.294(±0.001)**	**0.245(±0.005)**	**0.2927(±0.0004)**	**0.2391(±0.0038)**
b_count	**44(±0)**	**48(±1)**	**47(±1)**	**50(±1)**	**50(±1)**	**53(±1)**	**51(±0)**	**54(±0)**
b_1rotN/HA	**0.1667(±0)**	**0.1786(±0.0032)**	**0.1667(±0)**	**0.1795(±0.0023)**	**0.172(±0.001)**	**0.188(±0.002)**	**0.1739(±0)**	**0.1892(±0.0017)**
b_ar/HA	**0.51(±0.01)**	**0.48(±0.01)**	**0.524(±0.003)**	**0.486(±0.014)**	**0.526(±0.003)**	**0.474(±0.012)**	**0.5217(±0.0021)**	**0.4615(±0)**
b_count/HA	**1.810(±0.003)**	**1.915(±0.011)**	**1.800(±0.004)**	**1.923(±0.007)**	**1.806(±0.001)**	**1.933(±0.006)**	**1.815(±0.001)**	**1.944(±0.006)**
b_single/b_count	**0.681(±0.002)**	**0.720(±0.006)**	**0.667(±0.006)**	**0.720(±0.004)**	**0.671(±0.005)**	**0.727(±0.002)**	**0.673(±0.001)**	**0.732(±0.003)**
a_acc/HA	**0.118(±0.002)**	**0.111(±0.004)**	**0.120(±0.001)**	**0.107(±0.002)**	**0.121(±0.001)**	**0.107(±0.001)**	**0.120(±0.001)**	**0.107(±0.001)**
a_aro/HA	**0.5000(±0)**	**0.4737(±0.0121)**	**0.5217(±0)**	**0.4800(±0.0017)**	**0.5217(±0)**	**0.4615(±0.0072)**	**0.522(±0.004)**	**0.458(±0.006)**
b_rotN/HA	**0.1905(±0.003)**	**0.2(±0.0)**	**0.1905(±0.0018)**	**0.2(±0.0)**	**0.2(±0.0)**	**0.2105(±0.0038)**	**0.2(±0.0)**	**0.2143(±0.0038)**
b_1rotN/b_count	**0.093(±0.002)**	**0.097(±0.002)**	**0.094(±0.001)**	**0.096(±0.001)**	**0.0972(±0.0004)**	**0.1000(±0.002)**	**0.0978(±0.0003)**	**0.1000(±0.0013)**
a_nC/HA	**0.7333(±0.0026)**	**0.7500(±0)**	**0.733(±0.003)**	**0.750(±0.007)**	**0.7308(±0)**	**0.7500(±0)**	**0.7308(±0)**	**0.7500(±0)**
logS	**-4.48(±0.02)**	**-4.62(±0.07)**	-4.79(±0.02)	-4.85(±0.05)	**-5.16(±0.01)**	**-5.12(±0.04)**	**-5.17(±0.01)**	**-5.14(±0.03)**
a_nC	**18(±0)**	**19(±0)**	**19(±0)**	**20(±0)**	**20(±0)**	**21(±0)**	20(±1)	21(±0)
a_heavy	**24(±0)**	**25(±0)**	26(±0)	26(±0)	28(±0)	28(±0)	28(±0)	28(±0)
b_1rotN	4(±0)	5(±1)	**4(±0)**	**5(±0)**	5(±0)	5(±0)	5(±0)	5(±0)
b_rotN/b_count	0.106(±0.001)	0.105(±0.002)	0.106(±0.001)	0.106(±0.001)	0.110(±0.002)	0.111(±0.002)	0.1111(±0)	0.1111(±0.0021)
a_acc	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)	3(±0)
a_acid	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
a_acid/HA	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
a_aro	12(±0)	12(±0)	12(±0)	12(±0)	**15(±0)**	**12(±0)**	**15(±0)**	**12(±0)**
a_base	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
a_base/HA	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
a_don	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
b_ar	12(±0)	12(±0)	12(±0)	12(±0)	**16(±0)**	**12(±0)**	**16(±0)**	**12(±0)**
b_rotN	5(±0)	5(±0)	5(±0)	5(±0)	**5(±0)**	**6(±0)**	6(±0)	6(±0)
chiral	0(±0)	0(±0)	0(±0)	0(±0)	**0(±0)**	**1(±0)**	0(±0)	1(±0)
chiral/HA	0(±0)	0(±0)	0(±0)	0(±0)	**0(±0)**	**0.0256(±0.0039)**	0(±0)	0.0278(±0.0016)
FCharge	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	**0(±0)**	**0(±0)**
FCharge/HA	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	**0(±0)**	**0(±0)**
lip_druglike	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
lip_violation	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)	0(±0)
opr_leadlike	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)	1(±0)
opr_violation	0(±0)	0(±0)	0(±0)	1(±1)	1(±0)	1(±0)	1(±0)	1(±0)
rings	3(±0)	3(±0)	3(±0)	3(±0)	4(±0)	4(±1)	4(±0)	4(±0)

Reference

[1] A. Lehninger, D. Nelson, and M. Cox, *Lehninger Principles of Biochemistry*. W. H. Freeman, 2008.

[2] D. S. Wishart, C. Knox, A. C. Guo, S. Shrivastava, M. Hassanali, P. Stothard, Z. Chang, and J. Woolsey, “DrugBank: a comprehensive resource for in silico drug discovery and exploration,” *Nucleic Acids Res.*, vol. 34, no. suppl 1, pp. D668–D672, Jan. 2006.

[3] C. Knox, V. Law, T. Jewison, P. Liu, S. Ly, A. Frolkis, A. Pon, K. Banco, C. Mak, V. Neveu, Y. Djoumbou, R. Eisner, A. C. Guo, and D. S. Wishart, “DrugBank 3.0: a comprehensive resource for ‘omics’ research on drugs,” *Nucleic Acids Res.*, vol. 39, no. Database issue, pp. D1035–1041, Jan. 2011.

[4] D. S. Wishart, C. Knox, A. C. Guo, D. Cheng, S. Shrivastava, D. Tzur, B. Gautam, and M. Hassanali, “DrugBank: a knowledgebase for drugs, drug actions and drug targets,” *Nucleic Acids Res.*, vol. 36, no. Database issue, pp. D901–906, Jan. 2008.

[5] “Drug definition | Drugs.com.” [Online]. Available: http://www.drugs.com/dict/drug.html. [Accessed: 01-Apr-2016].

[6] T. Takenaka, “Classical vs reverse pharmacology in drug discovery,” *BJU Int.*, vol. 88 Suppl 2, pp. 7–10; discussion 49–50, Sep. 2001.

[7] P. Krogsgaard-Larsen, T. Liljefors, and U. Madsen, *Textbook of drug design and discovery*. London ; New York: Taylor & Francis, 2002.

[8] T. Cheng, Q. Li, Z. Zhou, Y. Wang, and S. H. Bryant, “Structure-Based Virtual Screening for Drug Discovery: a Problem-Centric Review,” *AAPS J.*, vol. 14, no. 1, pp. 133–141, Jan. 2012.

[9] N. A. Roberts, J. A. Martin, D. Kinchington, A. V. Broadhurst, J. C. Craig, I. B. Duncan, S. A. Galpin, B. K. Handa, J. Kay, and A. Kröhn, “Rational design of peptide-based HIV proteinase inhibitors,” *Science*, vol. 248, no. 4953, pp. 358–361, Apr. 1990.

[10] J. Erickson, D. J. Neidhart, J. VanDrie, D. J. Kempf, X. C. Wang, D. W. Norbeck, J. J. Plattner, J. W. Rittenhouse, M. Turon, and N. Wideburg, “Design, activity, and 2.8 A crystal structure of a C2 symmetric inhibitor complexed to HIV-1 protease,” *Science*, vol. 249, no. 4968, pp. 527–533, Aug. 1990.

[11] B. D. Dorsey, R. B. Levin, S. L. McDaniel, J. P. Vacca, J. P. Guare, P. L. Darke, J. A. Zugay, E. A. Emini, and W. A. Schleif, “L-735,524: The Design of a Potent and Orally Bioavailable HIV Protease Inhibitor,” *J. Med. Chem.*, vol. 37, no. 21, pp. 3443–3451, Oct. 1994.

[12] A. C. Anderson, “The Process of Structure-Based Drug Design,” *Chem. Biol.*, vol. 10, no. 9, pp. 787–797, Sep. 2003.

[13] V. Mountain, “Astex, Structural Genomix, and Syrrx. I can see clearly now: structural biology and drug discovery,” *Chem. Biol.*, vol. 10, no. 2, pp. 95–98, Feb. 2003.

[14] B. K. Shoichet and I. D. Kuntz, “Predicting the structure of protein complexes: a step in the right direction,” *Chem. Biol.*, vol. 3, no. 3, pp. 151–156, Mar. 1996.

[15] C. M. Dobson, “Chemical space and biology,” *Nature*, vol. 432, no. 7019, pp. 824–828, Dec. 2004.

[16] R. S. Bohacek, C. McMartin, and W. C. Guida, “The art and practice of structure-based drug design: A molecular modeling perspective,” *Med. Res. Rev.*, vol. 16, no. 1, pp. 3–50, Jan. 1996.

[17] M. Cummings, A. Maxwell, and R. DesJarlais, “Processing of Small Molecule Databases for Automated Docking,” *Med. Chem.*, vol. 3, no. 1, pp. 107–113, Jan. 2007.

[18] S. Dandapani, G. Rosse, N. Southall, J. M. Salvino, and C. J. Thomas, “Selecting, Acquiring, and Using Small Molecule Libraries for High-Throughput Screening,” *Curr. Protoc. Chem. Biol.*, vol. 4, pp. 177–191, 2012.

[19] C. A. Lipinski, F. Lombardo, B. W. Dominy, and P. J. Feeney, “Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings,” *Adv. Drug Deliv. Rev.*, vol. 46, no. 1–3, pp. 3–26, Mar. 2001.

[20] R. Perez-Pineiro, A. Burgos, D. C. Jones, L. C. Andrew, H. Rodriguez, M. Suarez, A. H. Fairlamb, and D. S. Wishart, “Development of a novel virtual screening cascade protocol to identify potential trypanothione reductase inhibitors,” *J. Med. Chem.*, vol. 52, no. 6, pp. 1670–1680, Mar. 2009.

[21] R. Gozalbes, L. Simon, N. Froloff, E. Sartori, C. Monteils, and R. Baudelle, “Development and experimental validation of a docking strategy for the generation of kinase-targeted libraries,” *J. Med. Chem.*, vol. 51, no. 11, pp. 3124–3132, Jun. 2008.

[22] C. Sage, R. Wang, and G. Jones, “G-protein coupled receptors virtual screening using genetic algorithm focused chemical space,” *J. Chem. Inf. Model.*, vol. 51, no. 8, pp. 1754–1761, Aug. 2011.

[23] D. Kireev, T. J. Wigle, J. Norris-Drouin, J. M. Herold, W. P. Janzen, and S. V. Frye, “Identification of Non-Peptide Malignant Brain Tumor (MBT) Repeat Antagonists by Virtual Screening of Commercially Available Compounds,” *J. Med. Chem.*, vol. 53, no. 21, pp. 7625–7631, Nov. 2010.

[24] D. Kireev, T. J. Wigle, J. Norris-Drouin, J. M. Herold, W. P. Janzen, and S. V. Frye, “Identification of Non-Peptide Malignant Brain Tumor (MBT) Repeat Antagonists by Virtual Screening of Commercially Available Compounds,” *J. Med. Chem.*, vol. 53, no. 21, pp. 7625–7631, Nov. 2010.

[25] Y. Bustanji, I. M. Al-Masri, A. Qasem, A. G. Al-Bakri, and M. O. Taha, “In silico screening for non-nucleoside HIV-1 reverse transcriptase inhibitors using physicochemical filters and high-throughput docking followed by in vitro evaluation,” *Chem. Biol. Drug Des.*, vol. 74, no. 3, pp. 258–265, Sep. 2009.

[26] Q. Shen, G. Wang, S. Li, X. Liu, S. Lu, Z. Chen, K. Song, J. Yan, L. Geng, Z. Huang, W. Huang, G. Chen, and J. Zhang, “ASD v3.0: unraveling allosteric regulation with structural mechanisms and biological networks,” *Nucleic Acids Res.*, vol. 44, no. D1, pp. D527–D535, Jan. 2016.

[27] C. J. Wenthur, P. R. Gentry, T. P. Mathews, and C. W. Lindsley, “Drugs for allosteric sites on receptors,” *Annu. Rev. Pharmacol. Toxicol.*, vol. 54, pp. 165–184, 2014.

[28] H. Eckert and J. Bajorath, “Molecular similarity analysis in virtual screening: foundations, limitations and novel approaches,” *Drug Discov. Today*, vol. 12, no. 5–6, pp. 225–233, Mar. 2007.

[29] H. Eckert and J. Bajorath, “Design and Evaluation of a Novel Class-Directed 2D Fingerprint to Search for Structurally Diverse Active Compounds,” *J. Chem. Inf. Model.*, vol. 46, no. 6, pp. 2515–2526, Nov. 2006.

[30] M. Wawer, E. Lounkine, A. M. Wassermann, and J. Bajorath, “Data structures and computational tools for the extraction of SAR information from large compound sets,” *Drug Discov. Today*, vol. 15, no. 15–16, pp. 630–639, Aug. 2010.

[31] M. Wawer, E. Lounkine, A. M. Wassermann, and J. Bajorath, “Data structures and computational tools for the extraction of SAR information from large compound sets,” *Drug Discov. Today*, vol. 15, no. 15–16, pp. 630–639, Aug. 2010.

[32] M. Vogt, Y. Huang, and J. Bajorath, “From Activity Cliffs to Activity Ridges: Informative Data Structures for SAR Analysis,” *J. Chem. Inf. Model.*, vol. 51, no. 8, pp. 1848–1856, Aug. 2011.

[33] J. Weigelt, L. D. B. McBroom-Cerajewski, M. Schapira, Y. Zhao, C. H. Arrowsmith, and C. H. Arrowmsmith, “Structural genomics and drug discovery: all in the family,” *Curr. Opin. Chem. Biol.*, vol. 12, no. 1, pp. 32–39, Feb. 2008.

[34] H.-Y. Sun, F.-Q. Ji, L.-Y. Fu, Z.-Y. Wang, and H.-Y. Zhang, “Structural and Energetic Analyses of SNPs in Drug Targets and Implications for Drug Therapy,” *J. Chem. Inf. Model.*, vol. 53, no. 12, pp. 3343–3351, Dec. 2013.

[35] G.-F. Hao, G.-F. Yang, and C.-G. Zhan, “Structure-based methods for predicting target mutation-induced drug resistance and rational drug design to overcome the problem,” *Drug Discov. Today*, vol. 17, no. 19–20, pp. 1121–1126, Oct. 2012.

[36] J. L. Medina-Franco, G. M. Maggiora, M. A. Giulianotti, C. Pinilla, and R. A. Houghten, “A similarity-based data-fusion approach to the visual characterization and comparison of compound databases,” *Chem. Biol. Drug Des.*, vol. 70, no. 5, pp. 393–412, Nov. 2007.

[37] D. M. Maniyar, I. T. Nabney, B. S. Williams, and A. Sewing, “Data Visualization during the Early Stages of Drug Discovery,” *J. Chem. Inf. Model.*, vol. 46, no. 4, pp. 1806–1818, Jul. 2006.

[38] A. M. Clark, K. Dole, A. Coulon-Spektor, A. McNutt, G. Grass, J. S. Freundlich, R. C. Reynolds, and S. Ekins, “Open Source Bayesian Models. 1. Application to ADME/Tox and Drug Discovery Datasets,” *J. Chem. Inf. Model.*, vol. 55, no. 6, pp. 1231–1245, Jun. 2015.

[39] C. A. Nicolaou and N. Brown, “Multi-objective optimization methods in drug design,” *Drug Discov. Today Technol.*, vol. 10, no. 3, pp. e427–e435, Sep. 2013.

[40] Y. C. Martin, J. L. Kofron, and L. M. Traphagen, “Do structurally similar molecules have similar biological activity?,” *J. Med. Chem.*, vol. 45, no. 19, pp. 4350–4358, Sep. 2002.

[41] M. Vogt, D. Stumpfe, G. M. Maggiora, and J. Bajorath, “Lessons learned from the design of chemical space networks and opportunities for new applications,” *J. Comput. Aided Mol. Des.*, vol. 30, no. 3, pp. 191–208, Mar. 2016.

[42] B.-O. Gohlke, T. Overkamp, A. Richter, A. Richter, P. T. Daniel, B. Gillissen, and R. Preissner, “2D and 3D similarity landscape analysis identifies PARP as a novel off-target for the drug Vatalanib,” *BMC Bioinformatics*, vol. 16, p. 308, 2015.

[43] R. Lewis, R. Guha, T. Korcsmaros, and A. Bender, “Synergy Maps: exploring compound combinations using network-based visualization,” *J. Cheminformatics*, vol. 7, no. 1, p. 36, Aug. 2015.

[44] R. Liu, N. Singh, G. J. Tawa, A. Wallqvist, and J. Reifman, “Exploiting large-scale drug-protein interaction information for computational drug repurposing,” *BMC Bioinformatics*, vol. 15, no. 1, p. 210, Jun. 2014.

[45] H. Huang, T. Nguyen, S. Ibrahim, S. Shantharam, Z. Yue, and J. Y. Chen, “DMAP: a connectivity map database to enable identification of novel drug repositioning candidates,” *BMC Bioinformatics*, vol. 16, no. Suppl 13, p. S4, Sep. 2015.

[46] A. Bender and R. C. Glen, “Molecular similarity: a key technique in molecular informatics,” *Org. Biomol. Chem.*, vol. 2, no. 22, pp. 3204–3218, Nov. 2004.

[47] G. Maggiora, M. Vogt, D. Stumpfe, and J. Bajorath, “Molecular Similarity in Medicinal Chemistry,” *J. Med. Chem.*, vol. 57, no. 8, pp. 3186–3204, Apr. 2014.

[48] A. K. Ghose, V. N. Viswanadhan, and J. J. Wendoloski, “Prediction of Hydrophobic (Lipophilic) Properties of Small Organic Molecules Using Fragmental Methods: An Analysis of ALOGP and CLOGP Methods,” *J. Phys. Chem. A*, vol. 102, no. 21, pp. 3762–3772, May 1998.

[49] T. S. Rush, J. A. Grant, L. Mosyak, and A. Nicholls, “A shape-based 3-D scaffold hopping method and its application to a bacterial protein-protein interaction,” *J. Med. Chem.*, vol. 48, no. 5, pp. 1489–1495, Mar. 2005.

[50] X. Liu, H. Jiang, and H. Li, “SHAFTS: A Hybrid Approach for 3D Molecular Similarity Calculation. 1. Method and Assessment of Virtual Screening,” *J. Chem. Inf. Model.*, vol. 51, no. 9, pp. 2372–2385, Sep. 2011.

[51] Y. C. Martin, “Distance Comparisons: A New Strategy for Examining Three-Dimensional Structure?Activity Relationships,” in *Classical and Three-Dimensional*

QSAR in Agrochemistry, vol. 606, 0 vols., American Chemical Society, 1995, pp. 318–329.

[52] J. W. Raymond and P. Willett, “Maximum common subgraph isomorphism algorithms for the matching of chemical structures,” *J. Comput. Aided Mol. Des.*, vol. 16, no. 7, pp. 521–533, Jul. 2002.

[53] A. L. Teixeira and A. O. Falcao, “Noncontiguous atom matching structural similarity function,” *J. Chem. Inf. Model.*, vol. 53, no. 10, pp. 2511–2524, Oct. 2013.

[54] R. D. Brown and Y. C. Martin, “The Information Content of 2D and 3D Structural Descriptors Relevant to Ligand-Receptor Binding,” *J. Chem. Inf. Comput. Sci.*, vol. 37, no. 1, pp. 1–9, Jan. 1997.

[55] G. B. McGaughey, R. P. Sheridan, C. I. Bayly, J. C. Culberson, C. Kreatsoulas, S. Lindsley, V. Maiorov, J.-F. Truchon, and W. D. Cornell, “Comparison of Topological, Shape, and Docking Methods in Virtual Screening,” *J. Chem. Inf. Model.*, vol. 47, no. 4, pp. 1504–1519, Jul. 2007.

[56] D. Rogers and M. Hahn, “Extended-Connectivity Fingerprints,” *J. Chem. Inf. Model.*, vol. 50, no. 5, pp. 742–754, May 2010.

[57] D. J. Rogers and T. T. Tanimoto, “A Computer Program for Classifying Plants,” *Science*, vol. 132, no. 3434, pp. 1115–1118, Oct. 1960.

[58] P. Jaccard, “Distribution de la Flore Alpine dans le Bassin des Dranses et dans quelques régions voisines.,” *Bull. Soc. Vaudoise Sci. Nat.*, vol. 37, no. 140, pp. 241–72, 1901.

[59] L. Dice, “Measures of the Amount of Ecologic Association Between Species,” *Ecology*, vol. 26, no. 3, pp. 297–302, Jul. 1945.

[60] A. Tversky, “Features of similarity,” *Psychol. Rev.*, vol. 84, no. 4, pp. 327–352, 1977.

[61] S. Wollenhaupt and K. Baumann, “inSARa: Intuitive and Interactive SAR Interpretation by Reduced Graphs and Hierarchical MCS-Based Network Navigation,” *J. Chem. Inf. Model.*, Jun. 2014.

[62] I. Herman, G. Melancon, and M. S. Marshall, “Graph visualization and navigation in information visualization: A survey,” *IEEE Trans. Vis. Comput. Graph.*, vol. 6, no. 1, pp. 24–43, Jan. 2000.

[63] D. A. Keim, “Information Visualization and Visual Data Mining,” *IEEE Trans. Vis. Comput. Graph.*, vol. 8, no. 1, pp. 1–8, Jan. 2002.

[64] M. Wawer, L. Peltason, N. Weskamp, A. Teckentrup, and J. Bajorath, “Structure−Activity Relationship Anatomy by Network-like Similarity Graphs and Local Structure−Activity Relationship Indices,” *J. Med. Chem.*, vol. 51, no. 19, pp. 6075–6084, Oct. 2008.

[65] M. Wawer and J. Bajorath, “Similarity−Potency Trees: A Method to Search for SAR Information in Compound Data Sets and Derive SAR Rules,” *J. Chem. Inf. Model.*, vol. 50, no. 8, pp. 1395–1409, Aug. 2010.

[66] T. Sander, J. Freyss, M. von Korff, and C. Rufener, “DataWarrior: An Open-Source Program For Chemistry Aware Data Visualization And Analysis,” *J. Chem. Inf. Model.*, vol. 55, no. 2, pp. 460–473, Feb. 2015.

[67] M. Awale, R. van Deursen, and J.-L. Reymond, “MQN-Mapplet: Visualization of Chemical Space with Interactive Maps of DrugBank, ChEMBL, PubChem, GDB-11, and GDB-13,” *J. Chem. Inf. Model.*, vol. 53, no. 2, pp. 509–518, Feb. 2013.

[68] P. Lind, “Construction and Use of Fragment-Augmented Molecular Hasse Diagrams,” *J. Chem. Inf. Model.*, vol. 54, no. 2, pp. 387–395, Feb. 2014.

[69] S. J. Wilkens, J. Janes, and A. I. Su, “HierS: hierarchical scaffold clustering using topological chemical graphs,” *J. Med. Chem.*, vol. 48, no. 9, pp. 3182–3193, May 2005.

[70] S. Wetzel, K. Klein, S. Renner, D. Rauh, T. I. Oprea, P. Mutzel, and H. Waldmann, “Interactive exploration of chemical space with Scaffold Hunter,” *Nat. Chem. Biol.*, vol. 5, no. 8, pp. 581–583, Aug. 2009.

[71] E. Lounkine and J. Bajorath, “Core trees and consensus fragment sequences for molecular representation and similarity analysis,” *J. Chem. Inf. Model.*, vol. 48, no. 6, pp. 1161–1166, Jun. 2008.

[72] A. M. Clark, “2D depiction of fragment hierarchies,” *J. Chem. Inf. Model.*, vol. 50, no. 1, pp. 37–46, Jan. 2010.

[73] M. A. Koch, A. Schuffenhauer, M. Scheck, S. Wetzel, M. Casaulta, A. Odermatt, P. Ertl, and H. Waldmann, “Charting biologically relevant chemical space: A structural classification of natural products (SCONP),” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 102, no. 48, pp. 17272–17277, Nov. 2005.

[74] A. Schuffenhauer, P. Ertl, S. Roggo, S. Wetzel, M. A. Koch, and H. Waldmann, “The scaffold tree--visualization of the scaffold universe by hierarchical scaffold classification,” *J. Chem. Inf. Model.*, vol. 47, no. 1, pp. 47–58, Feb. 2007.

[75] P. Ertl and B. Rohde, “The Molecule Cloud - compact visualization of large collections of molecules,” *J. Cheminformatics*, vol. 4, no. 1, p. 12, Jul. 2012.

[76] G. M. Maggiora and J. Bajorath, “Chemical space networks: a powerful new paradigm for the description of chemical space,” *J. Comput. Aided Mol. Des.*, vol. 28, no. 8, pp. 795–802, Jun. 2014.

[77] M. Gütlein, A. Karwath, and S. Kramer, “CheS-Mapper 2.0 for visual validation of (Q)SAR models,” *J. Cheminformatics*, vol. 6, no. 1, pp. 1–18, 2014.

[78] M. Gütlein, A. Karwath, and S. Kramer, “CheS-Mapper - Chemical Space Mapping and Visualization in 3D,” *J. Cheminformatics*, vol. 4, no. 1, pp. 1–16, 2012.

[79] J. B. Kruskal Jr., “On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem,” *Proc. Am. Math. Soc.*, vol. 7, no. 1, pp. 48–50, Feb. 1956.

[80] M. Zwierzyna, M. Vogt, G. M. Maggiora, and J. Bajorath, “Design and characterization of chemical space networks for different compound data sets,” *J. Comput. Aided Mol. Des.*, vol. 29, no. 2, pp. 113–125, Feb. 2015.

[81] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” *J. Stat. Mech. Theory Exp.*, vol. 2008, no. 10, p. P10008, Oct. 2008.

[82] M. E. J. Newman, “Detecting community structure in networks,” *Eur. Phys. J. B - Condens. Matter Complex Syst.*, vol. 38, no. 2, pp. 321–330, Mar. 2004.

[83] A. J. Alvarez, C. E. Sanz-Rodríguez, and J. L. Cabrera, “Weighting dissimilarities to detect communities in networks,” *Phil Trans R Soc A*, vol. 373, no. 2056, p. 20150108, Dec. 2015.

[84] M. E. J. Newman, “Fast algorithm for detecting community structure in networks,” *Phys. Rev. E*, vol. 69, no. 6, Jun. 2004.

[85] S. G. Kobourov, “Spring Embedders and Force Directed Graph Drawing Algorithms,” *ArXiv12013011 Cs*, Jan. 2012.

[86] Y. Koren, “Drawing graphs by eigenvectors: theory and practice,” *Comput. Math. Appl.*, vol. 49, no. 11–12, pp. 1867–1888, Jun. 2005.

[87] O. Bastert and C. Matuszewski, “Layered Drawings of Digraphs,” in *Drawing Graphs*, M. Kaufmann and D. Wagner, Eds. Springer Berlin Heidelberg, 2001, pp. 87–120.

[88] Y. A. Ivanenkov, N. P. Savchuk, S. Ekins, and K. V. Balakin, “Computational mapping tools for drug discovery,” *Drug Discov. Today*, vol. 14, no. 15–16, pp. 767–775, Aug. 2009.

[89] G. M. Keserü and G. M. Makara, “The influence of lead discovery strategies on the properties of drug candidates,” *Nat. Rev. Drug Discov.*, vol. 8, no. 3, pp. 203–212, Mar. 2009.

[90] A. C. Cheng, R. G. Coleman, K. T. Smyth, Q. Cao, P. Soulard, D. R. Caffrey, A. C. Salzberg, and E. S. Huang, “Structure-based maximal affinity model predicts small-molecule druggability,” *Nat. Biotechnol.*, vol. 25, no. 1, pp. 71–75, Jan. 2007.

[91] P. Imming, C. Sinning, and A. Meyer, “Drugs, their targets and the nature and number of drug targets,” *Nat. Rev. Drug Discov.*, vol. 5, no. 10, pp. 821–834, Oct. 2006.

[92] A. L. Hopkins and C. R. Groom, “The druggable genome,” *Nat. Rev. Drug Discov.*, vol. 1, no. 9, pp. 727–730, Sep. 2002.

[93] D. Brown and G. Superti-Furga, “Rediscovering the sweet spot in drug discovery,” *Drug Discov. Today*, vol. 8, no. 23, pp. 1067–1077, Dec. 2003.

[94] S. K. Schreyer, C. N. Parker, and G. M. Maggiora, “Data Shaving: A Focused Screening Approach,” *J. Chem. Inf. Comput. Sci.*, vol. 44, no. 2, pp. 470–479, Mar. 2004.

[95] N. Y. Mok and R. Brenk, “Mining the ChEMBL Database: An Efficient Chemoinformatics Workflow for Assembling an Ion Channel-Focused Screening Library,” *J. Chem. Inf. Model.*, vol. 51, no. 10, pp. 2449–2454, Oct. 2011.

[96] G. Chen, S. Zheng, X. Luo, J. Shen, W. Zhu, H. Liu, C. Gui, J. Zhang, M. Zheng, C. M. Puah, K. Chen, and H. Jiang, “Focused combinatorial library design based on structural diversity, druglikeness and binding affinity score,” *J. Comb. Chem.*, vol. 7, no. 3, pp. 398–406, Jun. 2005.

[97] Z. Deng, C. Chuaqui, and J. Singh, “Knowledge-based design of target-focused libraries using protein-ligand interaction constraints,” *J. Med. Chem.*, vol. 49, no. 2, pp. 490–500, Jan. 2006.

[98] D. Schnur, B. R. Beno, A. Good, and A. Tebben, “Approaches to target class combinatorial library design,” *Methods Mol. Biol. Clifton NJ*, vol. 275, pp. 355–378, 2004.

[99] D. Lagorce, O. Sperandio, J. B. Baell, M. A. Miteva, and B. O. Villoutreix, "FAF-Drugs3: a web server for compound property calculation and chemical library design," *Nucleic Acids Res.*, vol. 43, no. W1, pp. W200–W207, Jul. 2015.

[100] V. Khanna and S. Ranganathan, "Physicochemical property space distribution among human metabolites, drugs and toxins," *BMC Bioinformatics*, vol. 10, no. 15, pp. 1–18, 2009.

[101] O. A. Raevsky, S. V. Trepalin, H. P. Trepalina, V. A. Gerasimenko, and O. E. Raevskaja, "SLIPPER-2001 -- software for predicting molecular properties on the basis of physicochemical descriptors and structural similarity," *J. Chem. Inf. Comput. Sci.*, vol. 42, no. 3, pp. 540–549, Jun. 2002.

[102] L. Di and E. H. Kerns, "Profiling drug-like properties in discovery research," *Curr. Opin. Chem. Biol.*, vol. 7, no. 3, pp. 402–408, Jun. 2003.

[103] K. Horio, H. Muta, J. Goto, and N. Hirayama, "A simple method to improve the odds in finding 'lead-like' compounds from chemical libraries," *Chem. Pharm. Bull. (Tokyo)*, vol. 55, no. 7, pp. 980–984, Jul. 2007.

[104] T. I. Oprea, A. M. Davis, S. J. Teague, and P. D. Leeson, "Is There a Difference between Leads and Drugs? A Historical Perspective," *J. Chem. Inf. Comput. Sci.*, vol. 41, no. 5, pp. 1308–1315, Sep. 2001.

[105] T. I. Oprea, "Current trends in lead discovery: Are we looking for the appropriate properties?," *Mol. Divers.*, vol. 5, no. 4, pp. 199–208, Dec. 2000.

[106] T. I. Oprea and J. Gottfries, "Chemography: The Art of Navigating in Chemical Space," *J. Comb. Chem.*, vol. 3, no. 2, pp. 157–166, Mar. 2001.

[107] D. F. Veber, S. R. Johnson, H.-Y. Cheng, B. R. Smith, K. W. Ward, and K. D. Kopple, "Molecular properties that influence the oral bioavailability of drug candidates," *J. Med. Chem.*, vol. 45, no. 12, pp. 2615–2623, Jun. 2002.

[108] K. V. Balakin, S. E. Tkachenko, S. A. Lang, I. Okun, A. A. Ivashchenko, and N. P. Savchuk, "Property-Based Design of GPCR-Targeted Library," *J. Chem. Inf. Comput. Sci.*, vol. 42, no. 6, pp. 1332–1342, Nov. 2002.

[109] N. Singh, H. Sun, S. Chaudhury, M. D. M. AbdulHameed, A. Wallqvist, and G. Tawa, "A physicochemical descriptor-based scoring scheme for effective and rapid filtering of kinase-like chemical space," *J. Cheminformatics*, vol. 4, p. 4, Feb. 2012.

[110] G. Kéri, Z. Székelyhidi, P. Bánhegyi, Z. Varga, B. Hegymegi-Barakonyi, C. Szántai-Kis, D. Hafenbradl, B. Klebl, G. Muller, A. Ullrich, D. Erös, Z. Horváth, Z. Greff, J. Marosfalvi, J. Pató, I. Szabadkai, I. Szilágyi, Z. Szegedi, I. Varga, F. Wáczek, and L. Orfi,

“Drug discovery in the kinase inhibitory field using the Nested Chemical Library technology,” *Assay Drug Dev. Technol.*, vol. 3, no. 5, pp. 543–551, Oct. 2005.

[111] H. Xi and E. A. Lunney, “The design, annotation, and application of a kinase-targeted library,” *Methods Mol. Biol. Clifton NJ*, vol. 685, pp. 279–291, 2011.

[112] C. Zhang and G. Bollag, “Scaffold-based design of kinase inhibitors for cancer therapy,” *Curr. Opin. Genet. Dev.*, vol. 20, no. 1, pp. 79–86, Feb. 2010.

[113] H. Decornez, A. Gulyás-Forró, Á. Papp, M. Szabó, G. Sármay, I. Hajdú, S. Cseh, G. Dormán, and D. B. Kitchen, “Design, Selection, and Evaluation of a General Kinase-Focused Library,” *ChemMedChem*, vol. 4, no. 8, pp. 1273–1278, Aug. 2009.

[114] F. Deanda, E. L. Stewart, M. J. Reno, and D. H. Drewry, “Kinase-Targeted Library Design through the Application of the PharmPrint Methodology,” *J. Chem. Inf. Model.*, vol. 48, no. 12, pp. 2395–2403, Dec. 2008.

[115] J. N. C. Kew, “Positive and negative allosteric modulation of metabotropic glutamate receptors: emerging therapeutic potential,” *Pharmacol. Ther.*, vol. 104, no. 3, pp. 233–244, Dec. 2004.

[116] S. R. J. Hoare, “Mechanisms of peptide and nonpeptide ligand binding to Class B G-protein-coupled receptors,” *Drug Discov. Today*, vol. 10, no. 6, pp. 417–427, Mar. 2005.

[117] F. A. Kruger and J. P. Overington, “Global Analysis of Small Molecule Binding to Related Protein Targets,” *PLOS Comput Biol*, vol. 8, no. 1, p. e1002333, Jan. 2012.

[118] E. De Clercq, “The role of non-nucleoside reverse transcriptase inhibitors (NNRTIs) in the therapy of HIV-1 infection,” *Antiviral Res.*, vol. 38, no. 3, pp. 153–179, Jun. 1998.

[119] R. Pauwels, K. Andries, J. Desmyter, D. Schols, M. J. Kukla, H. J. Breslin, A. Raeymaeckers, J. Van Gelder, R. Woestenborghs, and J. Heykants, “Potent and selective inhibition of HIV-1 replication in vitro by a novel series of TIBO derivatives,” *Nature*, vol. 343, no. 6257, pp. 470–474, Feb. 1990.

[120] P. J. Conn, A. Christopoulos, and C. W. Lindsley, “Allosteric modulators of GPCRs: a novel approach for the treatment of CNS disorders,” *Nat. Rev. Drug Discov.*, vol. 8, no. 1, pp. 41–54, Jan. 2009.

[121] G.-F. Hao, G.-F. Yang, and C.-G. Zhan, “Structure-based methods for predicting target mutation-induced drug resistance and rational drug design to overcome the problem,” *Drug Discov. Today*, vol. 17, no. 19–20, pp. 1121–1126, Oct. 2012.

[122] L. Kruglyak and D. A. Nickerson, “Variation is the spice of life,” *Nat. Genet.*, vol. 27, no. 3, pp. 234–236, Mar. 2001.

[123] T. 1000 G. P. Consortium, “A map of human genome variation from population-scale sequencing,” *Nature*, vol. 467, no. 7319, pp. 1061–1073, Oct. 2010.

[124] K. A. Frazer, S. S. Murray, N. J. Schork, and E. J. Topol, “Human genetic variation and its contribution to complex traits,” *Nat. Rev. Genet.*, vol. 10, no. 4, pp. 241–251, Apr. 2009.

[125] C. Kimchi-Sarfaty, J. M. Oh, I.-W. Kim, Z. E. Sauna, A. M. Calcagno, S. V. Ambudkar, and M. M. Gottesman, “A ‘silent’ polymorphism in the MDR1 gene changes substrate specificity,” *Science*, vol. 315, no. 5811, pp. 525–528, Jan. 2007.

[126] A. E. Eriksson, W. A. Baase, X. J. Zhang, D. W. Heinz, M. Blaber, E. P. Baldwin, and B. W. Matthews, “Response of a protein structure to cavity-creating mutations and its relation to the hydrophobic effect,” *Science*, vol. 255, no. 5041, pp. 178–183, Jan. 1992.

[127] Z. Wang and J. Moult, “SNPs, protein structure, and disease,” *Hum. Mutat.*, vol. 17, no. 4, pp. 263–270, Apr. 2001.

[128] S. Sunyaev, V. Ramensky, and P. Bork, “Towards a structural basis of human non-synonymous single nucleotide polymorphisms,” *Trends Genet.*, vol. 16, no. 5, pp. 198–200, May 2000.

[129] S. Gong and T. L. Blundell, “Structural and functional restraints on the occurrence of single amino acid variations in human proteins,” *PloS One*, vol. 5, no. 2, p. e9186, 2010.

[130] I. A. Adzhubei, S. Schmidt, L. Peshkin, V. E. Ramensky, A. Gerasimova, P. Bork, A. S. Kondrashov, and S. R. Sunyaev, “A method and server for predicting damaging missense mutations,” *Nat. Methods*, vol. 7, no. 4, pp. 248–249, Apr. 2010.

[131] J. Bendl, J. Stourac, O. Salanda, A. Pavelka, E. D. Wieben, J. Zendulka, J. Brezovsky, and J. Damborsky, “PredictSNP: Robust and Accurate Consensus Classifier for Prediction of Disease-Related Mutations,” *PLoS Comput. Biol.*, vol. 10, no. 1, Jan. 2014.

[132] S. Velankar, J. M. Dana, J. Jacobsen, G. van Ginkel, P. J. Gane, J. Luo, T. J. Oldfield, C. O’Donovan, M.-J. Martin, and G. J. Kleywegt, “SIFTS: Structure Integration with Function, Taxonomy and Sequences resource,” *Nucleic Acids Res.*, vol. 41, no. Database issue, pp. D483–489, Jan. 2013.

[133] R. Calabrese, E. Capriotti, P. Fariselli, P. L. Martelli, and R. Casadio, “Functional annotations improve the predictive score of human disease-related mutations in proteins,” *Hum. Mutat.*, vol. 30, no. 8, pp. 1237–1244, 2009.

[134] A. David, R. Razali, M. N. Wass, and M. J. E. Sternberg, “Protein-protein interaction sites are hot spots for disease-associated nonsynonymous SNPs,” *Hum. Mutat.*, vol. 33, no. 2, pp. 359–363, Feb. 2012.

[135] P. Yue, E. Melamud, and J. Moult, “SNPs3D: candidate gene and SNP selection for association studies,” *BMC Bioinformatics*, vol. 7, p. 166, 2006.

[136] R. A. Laskowski, “PDBsum: summaries and analyses of PDB structures,” *Nucleic Acids Res.*, vol. 29, no. 1, pp. 221–222, Jan. 2001.

[137] M. Hendlich, A. Bergner, J. Günther, and G. Klebe, “Relibase: design and development of a database for comprehensive analysis of protein-ligand interactions,” *J. Mol. Biol.*, vol. 326, no. 2, pp. 607–620, Feb. 2003.

[138] A. Yamaguchi, K. Iida, N. Matsui, S. Tomoda, K. Yura, and M. Go, “Het-PDB Navi.: a database for protein-small molecule interactions,” *J. Biochem. (Tokyo)*, vol. 135, no. 1, pp. 79–84, Jan. 2004.

[139] E. Kellenberger, P. Muller, C. Schalon, G. Bret, N. Foata, and D. Rognan, “sc-PDB: an annotated database of druggable binding sites from the Protein Data Bank,” *J. Chem. Inf. Model.*, vol. 46, no. 2, pp. 717–727, Apr. 2006.

[140] M. L. Benson, R. D. Smith, N. A. Khazanov, B. Dimcheff, J. Beaver, P. Dresslar, J. Nerothin, and H. A. Carlson, “Binding MOAD, a high-quality protein-ligand database,” *Nucleic Acids Res.*, vol. 36, no. Database issue, pp. D674–D678, Jan. 2008.

[141] L. Hu, M. L. Benson, R. D. Smith, M. G. Lerner, and H. A. Carlson, “Binding MOAD (Mother Of All Databases).,” *Proteins*, vol. 60, no. 3, pp. 333–40, Aug. 2005.

[142] R. Wang, X. Fang, Y. Lu, and S. Wang, “The PDBbind database: collection of binding affinities for protein-ligand complexes with known three-dimensional structures,” *J. Med. Chem.*, vol. 47, no. 12, pp. 2977–2980, Jun. 2004.

[143] J. Yang, A. Roy, and Y. Zhang, “BioLiP: a semi-manually curated database for biologically relevant ligand-protein interactions,” *Nucleic Acids Res.*, vol. 41, no. Database issue, pp. D1096–1103, Jan. 2013.

[144] O. Roche, R. Kiyama, and C. L. Brooks, “Ligand-protein database: linking protein-ligand complex structures to binding data,” *J. Med. Chem.*, vol. 44, no. 22, pp. 3592–3598, Oct. 2001.

[145] D. Puvanendrampillai and J. B. O. Mitchell, “Protein Ligand Database (PLD): additional understanding of the nature and specificity of protein–ligand complexes,” *Bioinformatics*, vol. 19, no. 14, pp. 1856–1857, Sep. 2003.

[146] P. Block, C. A. Sotriffer, I. Dramburg, and G. Klebe, “AffinDB: a freely accessible database of affinities for protein–ligand complexes from the PDB,” *Nucleic Acids Res.*, vol. 34, no. suppl 1, pp. D522–D526, Jan. 2006.

[147] “Binding MOAD - The Mother of All.” [Online]. Available: http://www.bindingmoad.org/. [Accessed: 05-Apr-2016].

[148] “Welcome to PDBbind-CN Database.” [Online]. Available: http://www.pdbbind-cn.org/. [Accessed: 05-Apr-2016].

[149] Z. Liu, Y. Li, L. Han, J. Li, J. Liu, Z. Zhao, W. Nie, Y. Liu, and R. Wang, “PDB-wide collection of binding data: current status of the PDBbind database,” *Bioinformatics*, vol. 31, no. 3, pp. 405–412, Feb. 2015.

[150] R. Wang, X. Fang, Y. Lu, C.-Y. Yang, and S. Wang, “The PDBbind database: methodologies and updates,” *J. Med. Chem.*, vol. 48, no. 12, pp. 4111–4119, Jun. 2005.

[151] B. Lee and F. M. Richards, “The interpretation of protein structures: Estimation of static accessibility,” *J. Mol. Biol.*, vol. 55, no. 3, pp. 379–IN4, Feb. 1971.

[152] W. Kabsch and C. Sander, “Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features,” *Biopolymers*, vol. 22, no. 12, pp. 2577–2637, Dec. 1983.

[153] F. Eisenhaber and P. Argos, “Improved Strategy in Analytic Surface Calculation for Molecular Systems: Handling of Singularities and Computational Efficiency,” *J Comput Chem*, vol. 14, no. 11, pp. 1272–1280, Nov. 1993.

[154] S. Hubbard and J. Thornton, *NACCESS*. Department of Biochemistry and Molecular Biology: University College London.

[155] R. Fraczkiewicz and W. Braun, “Exact and efficient analytical calculation of the accessible surface areas and their gradients for macromolecules,” *J. Comput. Chem.*, vol. 19, no. 3, pp. 319–333, Feb. 1998.

[156] B. Lee and F. M. Richards, “The interpretation of protein structures: Estimation of static accessibility,” *J. Mol. Biol.*, vol. 55, no. 3, pp. 379–IN4, Feb. 1971.

[157] A. Shrake and J. A. Rupley, “Environment and exposure to solvent of protein atoms. Lysozyme and insulin,” *J. Mol. Biol.*, vol. 79, no. 2, pp. 351–371, Sep. 1973.

[158] M. H. Mucchielli-Giorgi, S. Hazout, and P. Tufféry, “PredAcc: prediction of solvent accessibility,” *Bioinforma. Oxf. Engl.*, vol. 15, no. 2, pp. 176–177, Feb. 1999.

[159] B. Rost and C. Sander, “Conservation and prediction of solvent accessibility in protein families,” *Proteins Struct. Funct. Bioinforma.*, vol. 20, no. 3, pp. 216–226, Nov. 1994.

[160] S. Pascarella, R. De Persio, F. Bossa, and P. Argos, “Easy method to predict solvent accessibility from multiple protein sequence alignments,” *Proteins*, vol. 32, no. 2, pp. 190–199, Aug. 1998.

[161] S. Ahmad, M. M. Gromiha, and A. Sarai, “Real value prediction of solvent accessibility from amino acid sequence,” *Proteins*, vol. 50, no. 4, pp. 629–635, Mar. 2003.

[162] J. A. Cuff, M. E. Clamp, A. S. Siddiqui, M. Finlay, and G. J. Barton, “JPred: a consensus secondary structure prediction server.,” *Bioinformatics*, vol. 14, no. 10, pp. 892–893, Jan. 1998.

[163] F. Jiang, “Prediction of protein secondary structure with a reliability score estimated by local sequence clustering,” *Protein Eng.*, vol. 16, no. 9, pp. 651–657, Sep. 2003.

[164] S. Montgomerie, S. Sundararaj, W. J. Gallin, and D. S. Wishart, “Improving the accuracy of protein secondary structure prediction using structural alignment,” *BMC Bioinformatics*, vol. 7, p. 301, 2006.

[165] H. Kaur and G. P. S. Raghava, “Role of evolutionary information in prediction of aromatic-backbone NH interactions in proteins,” *FEBS Lett.*, vol. 564, no. 1–2, pp. 47–57, Apr. 2004.

[166] C. Schaefer and B. Rost, “Predict impact of single amino acid change upon protein structure,” *BMC Genomics*, vol. 13, no. Suppl 4, p. S4, Jun. 2012.

[167] O. V. Tsodikov, M. T. Record, and Y. V. Sergeev, “Novel computer program for fast exact calculation of accessible and molecular surface areas and average surface curvature,” *J. Comput. Chem.*, vol. 23, no. 6, pp. 600–609, Apr. 2002.

[168] M. F. Sanner, A. J. Olson, and J. C. Spehner, “Reduced surface: an efficient way to compute molecular surfaces,” *Biopolymers*, vol. 38, no. 3, pp. 305–320, Mar. 1996.

[169] Y. Yamanishi, M. Kotera, M. Kanehisa, and S. Goto, “Drug-target interaction prediction from chemical, genomic and pharmacological data in an integrated framework,” *Bioinformatics*, vol. 26, no. 12, pp. i246–i254, Jun. 2010.

[170] B.-O. Gohlke, T. Overkamp, A. Richter, A. Richter, P. T. Daniel, B. Gillissen, and R. Preissner, “2D and 3D similarity landscape analysis identifies PARP as a novel off-target for the drug Vatalanib,” *BMC Bioinformatics*, vol. 16, no. 1, p. 308, Sep. 2015.

[171] M. A. Kuenemann, L. M. Bourbon, C. M. Labbé, B. O. Villoutreix, and O. Sperandio, “An exploration of the 3D chemical space has highlighted a specific shape profile for the compounds intended to inhibit protein-protein interactions,” *BMC Bioinformatics*, vol. 16, no. Suppl 3, p. A5, Feb. 2015.

[172] Y. Wang, J. Xiao, T. O. Suzek, J. Zhang, J. Wang, Z. Zhou, L. Han, K. Karapetyan, S. Dracheva, B. A. Shoemaker, E. Bolton, A. Gindulyte, and S. H. Bryant, “PubChem’s BioAssay Database,” *Nucleic Acids Res.*, vol. 40, no. Database issue, pp. D400–412, Jan. 2012.

[173] A. Gaulton, L. J. Bellis, A. P. Bento, J. Chambers, M. Davies, A. Hersey, Y. Light, S. McGlinchey, D. Michalovich, B. Al-Lazikani, and J. P. Overington, “ChEMBL: a large-scale bioactivity database for drug discovery,” *Nucleic Acids Res.*, vol. 40, no. D1, pp. D1100–D1107, Jan. 2012.

[174] R. E. Carhart, D. H. Smith, and R. Venkataraghavan, “Atom pairs as molecular features in structure-activity studies: definition and applications,” *J. Chem. Inf. Comput. Sci.*, vol. 25, no. 2, pp. 64–73, May 1985.

[175] J. L. Durant, B. A. Leland, D. R. Henry, and J. G. Nourse, “Reoptimization of MDL Keys for Use in Drug Discovery,” *J. Chem. Inf. Comput. Sci.*, vol. 42, no. 6, pp. 1273–1280, Nov. 2002.

[176] R. Nilakantan, N. Bauman, J. S. Dixon, and R. Venkataraghavan, “Topological torsion: a new molecular descriptor for SAR applications. Comparison with other descriptors,” *J. Chem. Inf. Comput. Sci.*, vol. 27, no. 2, pp. 82–85, May 1987.

[177] A. Gobbi and D. Poppinger, “Genetic optimization of combinatorial libraries,” *Biotechnol. Bioeng.*, vol. 61, no. 1, pp. 47–54, Dec. 1998.

[178] M. Levandowsky and D. Winter, “Distance between Sets,” *Nature*, vol. 234, no. 5323, pp. 34–35, Nov. 1971.

[179] N. Saitou and M. Nei, “The neighbor-joining method: a new method for reconstructing phylogenetic trees,” *Mol. Biol. Evol.*, vol. 4, no. 4, pp. 406–425, Jul. 1987.

[180] L. S. Vinh and A. von Haeseler, “Shortest triplet clustering: reconstructing large phylogenies using representative sets,” *BMC Bioinformatics*, vol. 6, p. 92, 2005.

[181] K. Tamura, M. Nei, and S. Kumar, “Prospects for inferring very large phylogenies by using the neighbor-joining method,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 101, no. 30, pp. 11030–11035, Jul. 2004.

[182] R. Mihaescu, D. Levy, and L. Pachter, “Why Neighbor-Joining Works,” *Algorithmica*, vol. 54, no. 1, pp. 1–24, Dec. 2007.

[183] M. Simonsen, T. Mailund, and C. N. S. Pedersen, “Rapid Neighbour-Joining,” in *Algorithms in Bioinformatics*, vol. 5251, K. A. Crandall and J. Lagergren, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 113–122.

[184] T. Liu, Y. Lin, X. Wen, R. N. Jorissen, and M. K. Gilson, “BindingDB: a web-accessible database of experimentally determined protein-ligand binding affinities,” *Nucleic Acids Res.*, vol. 35, no. Database issue, pp. D198–201, Jan. 2007.

[185] K. P. Seiler, G. A. George, M. P. Happ, N. E. Bodycombe, H. A. Carrinski, S. Norton, S. Brudz, J. P. Sullivan, J. Muhlich, M. Serrano, P. Ferraiolo, N. J. Tolliday, S. L. Schreiber, and P. A. Clemons, “ChemBank: a small-molecule screening and cheminformatics resource database,” *Nucleic Acids Res.*, vol. 36, no. Database issue, pp. D351–D359, Jan. 2008.

[186] D. Sculley, “Web-scale K-means Clustering,” in *Proceedings of the 19th International Conference on World Wide Web*, New York, NY, USA, 2010, pp. 1177–1178.

[187] M. Hassan, J. P. Bielawski, J. C. Hempel, and M. Waldman, “Optimization and visualization of molecular diversity of combinatorial libraries,” *Mol. Divers.*, vol. 2, no. 1–2, pp. 64–74, Oct. 1996.

[188] M. Hassan, R. D. Brown, S. Varma-O’brien, and D. Rogers, “Cheminformatics analysis and learning in a data pipelining environment,” *Mol. Divers.*, vol. 10, no. 3, pp. 283–299, Aug. 2006.

[189] B. Benatallah, F. Casati, and F. Toumani, “Web Service Conversation Modeling: A Cornerstone for E-Business Automation,” *IEEE Internet Comput.*, vol. 8, no. 1, pp. 46–54, Jan. 2004.

[190] S. A. Wildman and G. M. Crippen, “Prediction of Physicochemical Parameters by Atomic Contributions,” *J. Chem. Inf. Comput. Sci.*, vol. 39, no. 5, pp. 868–873, Sep. 1999.

[191] “Introducing JSON.” .

[192] J. Ellson, E. Gansner, L. Koutsofios, S. North, G. Woodhull, S. Description, and L. Technologies, “Graphviz — open source graph drawing tools,” in *Lecture Notes in Computer Science*, 2001, pp. 483–484.

[193] Chemical Computing Group Inc., “Molecular Operating Environment (MOE), 2014.10,” 2014. .

[194] C. W. Yap, “PaDEL-descriptor: an open source software to calculate molecular descriptors and fingerprints,” *J. Comput. Chem.*, vol. 32, no. 7, pp. 1466–1474, May 2011.

[195] E. J. Gardiner, V. J. Gillet, P. Willett, and D. A. Cosgrove, “Representing clusters using a maximum common edge substructure algorithm applied to reduced graphs and molecular graphs,” *J. Chem. Inf. Model.*, vol. 47, no. 2, pp. 354–366, Apr. 2007.

[196] J. J. Sutherland, L. A. O’Brien, and D. F. Weaver, “A comparison of methods for modeling quantitative structure-activity relationships,” *J. Med. Chem.*, vol. 47, no. 22, pp. 5541–5554, Oct. 2004.

[197] F. Fontaine, M. Pastor, I. Zamora, and F. Sanz, “Anchor-GRIND: filling the gap between standard 3D QSAR and the GRid-INdependent descriptors,” *J. Med. Chem.*, vol. 48, no. 7, pp. 2687–2694, Apr. 2005.

[198] H. a Carlson, R. D. Smith, N. a Khazanov, P. D. Kirchhoff, J. B. Dunbar, and M. L. Benson, “Differences between high- and low-affinity complexes of enzymes and nonenzymes.,” *J. Med. Chem.*, vol. 51, no. 20, pp. 6432–41, Oct. 2008.

[199] J. Bajorath, “Exploring Activity Cliffs from a Chemoinformatics Perspective,” *Mol. Inform.*, vol. 33, no. 6–7, pp. 438–442, Jun. 2014.

[200] A. Christopoulos, “Allosteric binding sites on cell-surface receptors: novel targets for drug discovery,” *Nat. Rev. Drug Discov.*, vol. 1, no. 3, pp. 198–210, Mar. 2002.

[201] P. Jeffrey Conn, A. Christopoulos, and C. W. Lindsley, “Allosteric modulators of GPCRs: a novel approach for the treatment of CNS disorders,” *Nat. Rev. Drug Discov.*, vol. 8, no. 1, pp. 41–54, Jan. 2009.

[202] T. Kenakin, “Collateral efficacy in drug discovery: taking advantage of the good (allosteric) nature of 7TM receptors,” *Trends Pharmacol. Sci.*, vol. 28, no. 8, pp. 407–415, Aug. 2007.

[203] W. Soudijn, I. van Wijngaarden, and A. P. IJzerman, “Allosteric modulation of G protein-coupled receptors: perspectives and recent developments,” *Drug Discov. Today*, vol. 9, no. 17, pp. 752–758, Sep. 2004.

[204] J.-P. Changeux, “Allostery and the Monod-Wyman-Changeux Model After 50 Years,” *Annu. Rev. Biophys.*, vol. 41, no. 1, pp. 103–133, 2012.

[205] J. Monod, J. Wyman, and J.-P. Changeux, “On the nature of allosteric transitions: A plausible model,” *J. Mol. Biol.*, vol. 12, no. 1, pp. 88–118, May 1965.

[206] C. Feng and C. B. Post, “Insights into the allosteric regulation of Syk association with receptor ITAM, a multi-state equilibrium,” *Phys Chem Chem Phys*, 2015.

[207] C. Malosh, M. Turlington, T. M. Bridges, J. M. Rook, M. J. Noetzel, P. N. Vinson, T. Steckler, H. Lavreysen, C. Mackie, J. M. Bartolomé-Nebreda, S. Conde-Ceide, C. M. Martínez-Viturro, M. Piedrafita, M. R. Sánchez-Casado, G. J. Macdonald, J. S. Daniels, C. K. Jones, C. M. Niswender, P. J. Conn, C. W. Lindsley, and S. R. Stauffer, “Acyl dihydropyrazolo[1,5-a]pyrimidinones as metabotropic glutamate receptor 5 positive allosteric modulators,” *Bioorg. Med. Chem. Lett.*, 2015.

[208] H. H. Nickols and P. J. Conn, “Development of allosteric modulators of GPCRs for treatment of CNS disorders,” *Neurobiol. Dis.*, vol. 61, pp. 55–71, Jan. 2014.

[209] Y. Yu, R. E. Savage, S. Eathiraj, J. Meade, M. J. Wick, T. Hall, G. Abbadessa, and B. Scwartz, “Targeting AKT1-E17K and the PI3K/AKT Pathway with an Allosteric AKT Inhibitor, ARQ 092,” 2015.

[210] N. M. Goodey and S. J. Benkovic, “Allosteric regulation and catalysis emerge via a common route,” *Nat. Chem. Biol.*, vol. 4, no. 8, pp. 474–482, Aug. 2008.

[211] Y. Wang, J. Xiao, T. O. Suzek, J. Zhang, J. Wang, and S. H. Bryant, “PubChem: a public information system for analyzing bioactivities of small molecules,” *Nucleic Acids Res.*, vol. 37, no. suppl 2, pp. W623–W633, 2009.

[212] C. Knox, V. Law, T. Jewison, P. Liu, S. Ly, A. Frolkis, A. Pon, K. Banco, C. Mak, V. Neveu, Y. Djoumbou, R. Eisner, A. C. Guo, and D. S. Wishart, “DrugBank 3.0: a comprehensive resource for ‘Omics’ research on drugs,” *Nucleic Acids Res.*, vol. 39, no. suppl 1, pp. D1035–D1041, Jan. 2011.

[213] A. Gaulton, L. J. Bellis, A. P. Bento, J. Chambers, M. Davies, A. Hersey, Y. Light, S. McGlinchey, D. Michalovich, B. Al-Lazikani, and others, “ChEMBL: a large-scale bioactivity database for drug discovery,” *Nucleic Acids Res.*, vol. 40, no. D1, pp. D1100–D1107, 2012.

[214] Z. Huang, L. Zhu, Y. Cao, G. Wu, X. Liu, Y. Chen, Q. Wang, T. Shi, Y. Zhao, Y. Wang, and others, “ASD: a comprehensive database of allosteric proteins and modulators,” *Nucleic Acids Res.*, vol. 39, no. suppl 1, pp. D663–D669, 2011.

[215] Z. Huang, L. Mou, Q. Shen, S. Lu, C. Li, X. Liu, G. Wang, S. Li, L. Geng, Y. Liu, and others, “ASD v2. 0: updated content and novel features focusing on allosteric regulation,” *Nucleic Acids Res.*, vol. 42, no. D1, pp. D510–D516, 2014.

[216] W. Cui, Y.-H. Cheng, L.-L. Geng, D.-S. Liang, T.-J. Hou, and M.-J. Ji, “Unraveling the Allosteric Inhibition Mechanism of PTP1B by Free Energy Calculation Based on Umbrella Sampling,” *J. Chem. Inf. Model.*, vol. 53, no. 5, pp. 1157–1167, May 2013.

[217] C. Malosh, M. Turlington, T. M. Bridges, J. M. Rook, M. J. Noetzel, P. N. Vinson, T. Steckler, H. Lavreysen, C. Mackie, J. M. Bartolomé-Nebreda, S. Conde-Ceide, C. M. Martínez-Viturro, M. Piedrafita, M. R. Sánchez-Casado, G. J. Macdonald, J. S. Daniels, C. K. Jones, C. M. Niswender, P. J. Conn, C. W. Lindsley, and S. R. Stauffer, “Acyl dihydropyrazolo[1,5-a]pyrimidinones as metabotropic glutamate receptor 5 positive allosteric modulators,” *Bioorg. Med. Chem. Lett.*

[218] C. Feng and C. B. Post, “Insights into the allosteric regulation of Syk association with receptor ITAM, a multi-state equilibrium,” *Phys. Chem. Chem. Phys. PCCP*, Oct. 2015.

[219] Y. Yu, R. E. Savage, S. Eathiraj, J. Meade, M. J. Wick, T. Hall, G. Abbadessa, and B. Schwartz, “Targeting AKT1-E17K and the PI3K/AKT Pathway with an Allosteric AKT Inhibitor, ARQ 092,” *PloS One*, vol. 10, no. 10, p. e0140479, 2015.

[220] O. N. A. Demerdash, M. D. Daily, and J. C. Mitchell, “Structure-based predictive models for allosteric hot spots,” *PLoS Comput. Biol.*, vol. 5, no. 10, p. e1000531, Oct. 2009.

[221] X. Li, Y. Chen, S. Lu, Z. Huang, X. Liu, Q. Wang, T. Shi, and J. Zhang, “Toward an understanding of the sequence and structural basis of allosteric proteins,” *J. Mol. Graph. Model.*, vol. 40, pp. 30–39, Mar. 2013.

[222] X. Ma, Y. Qi, and L. Lai, “Allosteric sites can be identified based on the residue-residue interaction energy difference,” *Proteins*, vol. 83, no. 8, pp. 1375–1384, Aug. 2015.

[223] A. Panjkovich and X. Daura, “Exploiting protein flexibility to predict the location of allosteric sites,” *BMC Bioinformatics*, vol. 13, no. 1, p. 273, Oct. 2012.

[224] Q. Wang, M. Zheng, Z. Huang, X. Liu, H. Zhou, Y. Chen, T. Shi, and J. Zhang, “Toward understanding the molecular basis for chemical allosteric modulator design,” *J. Mol. Graph. Model.*, vol. 38, pp. 324–333, Sep. 2012.

[225] “Accelrys Available Chemicals Directory (ACD).” Accelrys, Inc., San Diego, USA, 2005.

[226] "Comprehensive Medicinal Chemistry (CMC)." Accelrys, Inc., San Diego, USA, 2009.

[227] "Chinese Natural Product Database (CNPD)." NeoTrident Technology Ltd., Beijing, China, 2005.

[228] "MDDR." Accelrys, Inc., San Diego, USA, 2009.

[229] "NCI Open Database." National Cancer, Ins., Bethesda, USA, 2003.

[230] G. J. P. van Westen, A. Gaulton, and J. P. Overington, "Chemical, Target, and Bioactive Properties of Allosteric Modulation," *PLoS Comput Biol*, vol. 10, no. 4, p. e1003559, Apr. 2014.

[231] Q. Wang, M. Zheng, Z. Huang, X. Liu, H. Zhou, Y. Chen, T. Shi, and J. Zhang, "Toward understanding the molecular basis for chemical allosteric modulator design," *J. Mol. Graph. Model.*, vol. 38, pp. 324–333, Sep. 2012.

[232] G. J. P. van Westen, A. Gaulton, and J. P. Overington, "Chemical, Target, and Bioactive Properties of Allosteric Modulation," *PLoS Comput Biol*, vol. 10, no. 4, p. e1003559, Apr. 2014.

[233] D. J. Lipman and W. R. Pearson, "Rapid and sensitive protein similarity searches," *Science*, vol. 227, no. 4693, pp. 1435–1441, Mar. 1985.

[234] Pipeline Pilot, .

[235] J. Lu and H. A. Carlson, "ChemTreeMap: An Interactive Map of Biochemical Similarity in Molecular Datasets.," *Bioinformatics (in revision)*, 2016.

[236] W. S. Palmer, M. Alam, H. B. Arzeno, K.-C. Chang, J. P. Dunn, D. M. Goldstein, L. Gong, B. Goyal, J. C. Hermann, J. H. Hogg, G. Hsieh, A. Jahangir, C. Janson, S. Jin, R. Ursula Kammlott, A. Kuglstatter, C. Lukacs, C. Michoud, L. Niu, D. C. Reuter, A. Shao, T. Silva, T. A. Trejo-Martin, K. Stein, Y.-C. Tan, P. Tivitmahaisoon, P. Tran, P. Wagner, P. Weller, and S.-Y. Wu, "Development of amino-pyrimidine inhibitors of c-Jun N-terminal kinase (JNK): Kinase profiling guided optimization of a 1,2,3-benzotriazole lead," *Bioorg. Med. Chem. Lett.*, vol. 23, no. 5, pp. 1486–1492, Mar. 2013.

[237] F. Wilcoxon, "Individual Comparisons by Ranking Methods," *Biom. Bull.*, vol. 1, no. 6, pp. 80–83, Dec. 1945.

[238] R. C. Team, *R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria, 2012*. ISBN 3-900051-07-0, 2014.

[239] Z. Huang, L. Zhu, Y. Cao, G. Wu, X. Liu, Y. Chen, Q. Wang, T. Shi, Y. Zhao, Y. Wang, W. Li, Y. Li, H. Chen, G. Chen, and J. Zhang, “ASD: a comprehensive database of allosteric proteins and modulators,” *Nucleic Acids Res.*, vol. 39, no. suppl 1, pp. D663–D669, Jan. 2011.

[240] Z. Huang, L. Mou, Q. Shen, S. Lu, C. Li, X. Liu, G. Wang, S. Li, L. Geng, Y. Liu, J. Wu, G. Chen, and J. Zhang, “ASD v2.0: updated content and novel features focusing on allosteric regulation,” *Nucleic Acids Res.*, vol. 42, no. D1, pp. D510–D516, Jan. 2014.

[241] F. Zhu, Z. Shi, C. Qin, L. Tao, X. Liu, F. Xu, L. Zhang, Y. Song, X. Liu, J. Zhang, B. Han, P. Zhang, and Y. Chen, “Therapeutic target database update 2012: a resource for facilitating target-oriented drug discovery,” *Nucleic Acids Res.*, vol. 40, no. Database issue, pp. D1128–1136, Jan. 2012.

[242] Q. Wang, M. Zheng, Z. Huang, X. Liu, H. Zhou, Y. Chen, T. Shi, and J. Zhang, “Toward understanding the molecular basis for chemical allosteric modulator design,” *J. Mol. Graph. Model.*, vol. 38, pp. 324–333, Sep. 2012.

[243] R. Lappano and M. Maggiolini, “G protein-coupled receptors: novel targets for drug discovery in cancer,” *Nat. Rev. Drug Discov.*, vol. 10, no. 1, pp. 47–60, Jan. 2011.

[244] J. R. Lane and A. P. IJzerman, “Allosteric approaches to GPCR drug discovery,” *Drug Discov. Today Technol.*, vol. 10, no. 2, pp. e219–221, 2013.

[245] R. O. Dror, H. F. Green, C. Valant, D. W. Borhani, J. R. Valcourt, A. C. Pan, D. H. Arlow, M. Canals, J. R. Lane, R. Rahmani, J. B. Baell, P. M. Sexton, A. Christopoulos, and D. E. Shaw, “Structural basis for modulation of a G-protein-coupled receptor by allosteric drugs,” *Nature*, vol. 503, no. 7475, pp. 295–299, Nov. 2013.

[246] X. Li, Y. Chen, S. Lu, Z. Huang, X. Liu, Q. Wang, T. Shi, and J. Zhang, “Toward an understanding of the sequence and structural basis of allosteric proteins,” *J. Mol. Graph. Model.*, vol. 40, pp. 30–39, Mar. 2013.

[247] W. Huang, S. Lu, Z. Huang, X. Liu, L. Mou, Y. Luo, Y. Zhao, Y. Liu, Z. Chen, T. Hou, and J. Zhang, “Allosite: a method for predicting allosteric sites,” *Bioinformatics*, Jul. 2013.

[248] S. Li, Q. Shen, M. Su, X. Liu, S. Lu, Z. Chen, R. Wang, and J. Zhang, “Alloscore: a tool for predicting allosteric ligand-protein interactions,” *Bioinformatics*, p. btw036, Jan. 2016.

[249] C. Chothia and J. Janin, “Principles of protein-protein recognition,” *Nature*, vol. 256, no. 5520, pp. 705–708, Aug. 1975.

[250] J. Janin and C. Chothia, “The structure of protein-protein recognition sites,” *J. Biol. Chem.*, vol. 265, no. 27, pp. 16027–16030, 1990.

[251] J. Gruber, A. Zawaira, R. Saunders, C. P. Barrett, and M. E. M. Noble, “Computational analyses of the surface properties of protein–protein interfaces,” *Acta Crystallogr. D Biol. Crystallogr.*, vol. 63, no. Pt 1, pp. 50–57, Jan. 2007.

[252] W. Huang, S. Lu, Z. Huang, X. Liu, L. Mou, Y. Luo, Y. Zhao, Y. Liu, Z. Chen, T. Hou, and J. Zhang, “Allosite: a method for predicting allosteric sites,” *Bioinformatics*, vol. 29, no. 18, pp. 2357–2359, Sep. 2013.

[253] A. Panjkovich and X. Daura, “Exploiting protein flexibility to predict the location of allosteric sites,” *BMC Bioinformatics*, vol. 13, no. 1, p. 273, Oct. 2012.

[254] C. Perez-Iratxeta, P. Bork, and M. A. Andrade, “Association of genes to genetically inherited diseases using data mining,” *Nat. Genet.*, vol. 31, no. 3, pp. 316–319, Jul. 2002.

[255] The 1000 Genomes Project Consortium, “A global reference for human genetic variation,” *Nature*, vol. 526, no. 7571, pp. 68–74, Oct. 2015.

[256] International HapMap Consortium, “The International HapMap Project,” *Nature*, vol. 426, no. 6968, pp. 789–796, Dec. 2003.

[257] M. L. Metzker, “Sequencing technologies - the next generation,” *Nat. Rev. Genet.*, vol. 11, no. 1, pp. 31–46, Jan. 2010.

[258] 1000 Genomes Project Consortium, G. R. Abecasis, D. Altshuler, A. Auton, L. D. Brooks, R. M. Durbin, R. A. Gibbs, M. E. Hurles, and G. A. McVean, “A map of human genome variation from population-scale sequencing,” *Nature*, vol. 467, no. 7319, pp. 1061–1073, Oct. 2010.

[259] 1000 Genomes Project Consortium, G. R. Abecasis, A. Auton, L. D. Brooks, M. A. DePristo, R. M. Durbin, R. E. Handsaker, H. M. Kang, G. T. Marth, and G. A. McVean, “An integrated map of genetic variation from 1,092 human genomes,” *Nature*, vol. 491, no. 7422, pp. 56–65, Nov. 2012.

[260] V. A. McKusick, “Mendelian Inheritance in Man and Its Online Version, OMIM,” *Am. J. Hum. Genet.*, vol. 80, no. 4, pp. 588–604, Apr. 2007.

[261] S. T. Sherry, M. H. Ward, M. Kholodov, J. Baker, L. Phan, E. M. Smigielski, and K. Sirotkin, “dbSNP: the NCBI database of genetic variation,” *Nucleic Acids Res.*, vol. 29, no. 1, pp. 308–311, Jan. 2001.

[262] P. D. Stenson, M. Mort, E. V. Ball, K. Howells, A. D. Phillips, N. S. Thomas, and D. N. Cooper, “The Human Gene Mutation Database: 2008 update,” *Genome Med.*, vol. 1, no. 1, p. 13, 2009.

[263] Y. L. Yip, M. Famiglietti, A. Gos, P. D. Duek, F. P. A. David, A. Gateau, and A. Bairoch, “Annotating single amino acid polymorphisms in the UniProt/Swiss-Prot knowledgebase,” *Hum. Mutat.*, vol. 29, no. 3, pp. 361–366, Mar. 2008.

[264] S. K. Burley, S. C. Almo, J. B. Bonanno, M. Capel, M. R. Chance, T. Gaasterland, D. Lin, A. Šali, F. W. Studier, and S. Swaminathan, “Structural genomics: beyond the Human Genome Project,” *Nat. Genet.*, vol. 23, no. 2, pp. 151–157, Oct. 1999.

[265] H. Dingerdissen, M. Motwani, K. Karagiannis, V. Simonyan, and R. Mazumder, “Proteome-wide analysis of nonsynonymous single-nucleotide variations in active sites of human proteins,” *FEBS J.*, vol. 280, no. 6, pp. 1542–1562, Mar. 2013.

[266] M. Gao, H. Zhou, and J. Skolnick, “Insights into Disease-Associated Mutations in the Human Proteome through Protein Structural Analysis,” *Structure*, vol. 23, no. 7, pp. 1362–1369, Jul. 2015.

[267] Z. Wang and J. Moult, “SNPs, protein structure, and disease,” *Hum. Mutat.*, vol. 17, no. 4, pp. 263–270, Apr. 2001.

[268] B. Thibert, D. E. Bredesen, and G. del Rio, “Improved prediction of critical residues for protein function based on network and phylogenetic analyses,” *BMC Bioinformatics*, vol. 6, p. 213, 2005.

[269] C. Ferrer-Costa, M. Orozco, and X. de la Cruz, “Characterization of disease-associated single amino acid polymorphisms in terms of sequence and structure properties1,” *J. Mol. Biol.*, vol. 315, no. 4, pp. 771–786, Jan. 2002.

[270] S. Jones and J. M. Thornton, “Analysis of protein-protein interaction sites using surface patches,” *J. Mol. Biol.*, vol. 272, no. 1, pp. 121–132, Sep. 1997.

[271] S. Miller, J. Janin, A. M. Lesk, and C. Chothia, “Interior and surface of monomeric proteins,” *J. Mol. Biol.*, vol. 196, no. 3, pp. 641–656, Aug. 1987.

[272] T. Schmidt, J. Haas, T. Gallo Cassarino, and T. Schwede, “Assessment of ligand-binding residue predictions in CASP9,” *Proteins*, vol. 79 Suppl 10, pp. 126–136, 2011.

[273] Y. Choi, G. E. Sims, S. Murphy, J. R. Miller, and A. P. Chan, “Predicting the Functional Effect of Amino Acid Substitutions and Indels,” *PLoS ONE*, vol. 7, no. 10, p. e46688, Oct. 2012.

[274] M. R. Stratton, P. J. Campbell, and P. A. Futreal, “The cancer genome,” *Nature*, vol. 458, no. 7239, pp. 719–724, Apr. 2009.

[275] C. Greenman, P. Stephens, R. Smith, G. L. Dalgliesh, C. Hunter, G. Bignell, H. Davies, J. Teague, A. Butler, C. Stevens, S. Edkins, S. O’Meara, I. Vastrik, E. E. Schmidt, T. Avis, S. Barthorpe, G. Bhamra, G. Buck, B. Choudhury, J. Clements, J. Cole, E. Dicks, S. Forbes, K. Gray, K. Halliday, R. Harrison, K. Hills, J. Hinton, A. Jenkinson, D. Jones, A. Menzies, T. Mironenko, J. Perry, K. Raine, D. Richardson, R. Shepherd, A. Small, C. Tofts, J. Varian, T. Webb, S. West, S. Widaa, A. Yates, D. P. Cahill, D. N. Louis, P. Goldstraw, A. G. Nicholson, F. Brasseur, L. Looijenga, B. L. Weber, Y.-E. Chiew, A. DeFazio, M. F. Greaves, A. R. Green, P. Campbell, E. Birney, D. F. Easton, G. Chenevix-Trench, M.-H. Tan, S. K. Khoo, B. T. Teh, S. T. Yuen, S. Y. Leung, R. Wooster, P. A. Futreal, and M. R. Stratton, “Patterns of somatic mutation in human cancer genomes,” *Nature*, vol. 446, no. 7132, pp. 153–158, Mar. 2007.

[276] C. M. Santiveri and M. A. Jiménez, “Tryptophan residues: scarce in proteins but strong stabilizers of β-hairpin peptides,” *Biopolymers*, vol. 94, no. 6, pp. 779–790, 2010.

[277] D. G. Isom, C. A. Castañeda, B. R. Cannon, P. D. Velu, and B. G.-M. E, “Charges in the hydrophobic interior of proteins,” *Proc. Natl. Acad. Sci.*, vol. 107, no. 37, pp. 16096–16100, Sep. 2010.

[278] B. Forood, E. J. Feliciano, and K. P. Nambiar, “Stabilization of alpha-helical structures in short peptides via end capping.,” *Proc. Natl. Acad. Sci.*, vol. 90, no. 3, pp. 838–842, Feb. 1993.

[279] S. Khan and M. Vihinen, “Spectrum of disease-causing mutations in protein secondary structures,” *BMC Struct. Biol.*, vol. 7, p. 56, Aug. 2007.

[280] G. Bellesia, A. I. Jewett, and J.-E. Shea, “Sequence periodicity and secondary structure propensity in model proteins,” *Protein Sci. Publ. Protein Soc.*, vol. 19, no. 1, pp. 141–154, Jan. 2010.

[281] S. Costantini, G. Colonna, and A. M. Facchiano, “Amino acid propensities for secondary structures are influenced by the protein structural class,” *Biochem. Biophys. Res. Commun.*, vol. 342, no. 2, pp. 441–451, Apr. 2006.

[282] N. A. Khazanov and H. A. Carlson, “Exploring the Composition of Protein-Ligand Binding Sites on a Large Scale,” *PLoS Comput. Biol.*, vol. 9, no. 11, Nov. 2013.

[283] T. J. A. Developer, M. P. F. User, and D. W. User, *epitools: Epidemiology Tools*. 2012.

[284] W. S. Alnosaier, *Kenward-Roger Approximate F Test for Fixed Effects in Mixed Linear Models*. Oregon State University, 2007.

[285] A. Ahmed, R. D. Smith, J. J. Clark, J. B. Dunbar, and H. A. Carlson, “Recent improvements to Binding MOAD: a resource for protein–ligand binding affinities and structures,” *Nucleic Acids Res.*, vol. 43, no. D1, pp. D465–D469, Jan. 2015.

[286] H. Berman, K. Henrick, and H. Nakamura, “Announcing the worldwide Protein Data Bank,” *Nat. Struct. Mol. Biol.*, vol. 10, no. 12, pp. 980–980, Dec. 2003.

[287] The Cancer Genome Atlas Research Network, J. N. Weinstein, E. A. Collisson, G. B. Mills, K. R. M. Shaw, B. A. Ozenberger, K. Ellrott, I. Shmulevich, C. Sander, and J. M. Stuart, “The Cancer Genome Atlas Pan-Cancer analysis project,” *Nat. Genet.*, vol. 45, no. 10, pp. 1113–1120, Oct. 2013.

[288] J. Zhang, J. Baran, A. Cros, J. M. Guberman, S. Haider, J. Hsu, Y. Liang, E. Rivkin, J. Wang, B. Whitty, M. Wong-Erasmus, L. Yao, and A. Kasprzyk, “International Cancer Genome Consortium Data Portal—a one-stop shop for cancer genomics data,” *Database J. Biol. Databases Curation*, vol. 2011, Sep. 2011.

[289] E. C. Butcher, E. L. Berg, and E. J. Kunkel, “Systems biology in drug discovery,” *Nat. Biotechnol.*, vol. 22, no. 10, pp. 1253–1259, Oct. 2004.

[290] A. L. Hopkins, “Network pharmacology: the next paradigm in drug discovery,” *Nat. Chem. Biol.*, vol. 4, no. 11, pp. 682–690, Nov. 2008.

[291] J. Pei, N. Yin, X. Ma, and L. Lai, “Systems Biology Brings New Dimensions for Structure-Based Drug Design,” *J. Am. Chem. Soc.*, vol. 136, no. 33, pp. 11556–11565, Aug. 2014.

[292] Y. Cao, T. Jiang, and T. Girke, “A maximum common substructure-based algorithm for searching and predicting drug-like compounds,” *Bioinformatics*, vol. 24, no. 13, pp. i366–i374, Jul. 2008.

[293] J. J. McGregor, “Backtrack search algorithms and the maximal common subgraph problem,” *Softw. Pract. Exp.*, vol. 12, no. 1, pp. 23–34, Jan. 1982.

[294] L. P. Cordella, P. Foggia, C. Sansone, F. Tortorella, and M. Vento, “Graph matching: a fast algorithm and its evaluation,” in *Fourteenth International Conference on Pattern Recognition, 1998. Proceedings*, 1998, vol. 2, pp. 1582–1584 vol.2.

[295] R. Guha and J. H. Van Drie, “Structure−Activity Landscape Index: Identifying and Quantifying Activity Cliffs,” *J. Chem. Inf. Model.*, vol. 48, no. 3, pp. 646–658, Mar. 2008.

[296] L. Peltason and J. Bajorath, “SAR Index: Quantifying the Nature of Structure−Activity Relationships,” *J. Med. Chem.*, vol. 50, no. 23, pp. 5571–5578, Nov. 2007.

[297] J. L. Dahlin, J. W. M. Nissink, J. M. Strasser, S. Francis, L. Higgins, H. Zhou, Z. Zhang, and M. A. Walters, “PAINS in the Assay: Chemical Mechanisms of Assay Interference and Promiscuous Enzymatic Inhibition Observed during a Sulfhydryl-Scavenging HTS,” *J. Med. Chem.*, vol. 58, no. 5, pp. 2091–2113, Mar. 2015.