New Computational Methods to Predict Cancer Resistance Mutations

and Design D-Peptide Therapeutics

by

Nathan Guerin

Department of Computer Science
Duke University

Date:________________________
Approved:

____
Bruce R. Donald, Supervisor

____
Alberto Bartesaghi

____
Raluca Gordân

____
Teresa Kaserer

Dissertation submitted in partial fulfillment of
the requirements for the degree of Doctor
of Philosophy in the Department of
Computer Science in the Graduate School
of Duke University

2023

ABSTRACT

New Computational Methods to Predict Cancer Resistance Mutations

and Design D-Peptide Therapeutics

by

Nathan Guerin

Department of Computer Science
Duke University

Date:__________________________
Approved:

____
Bruce R. Donald, Supervisor

____
Alberto Bartesaghi

____
Raluca Gordân

____
Teresa Kaserer

An abstract of a dissertation submitted in partial
fulfillment of the requirements for the degree
of Doctor of Philosophy in the Department of
Computer Science in the Graduate School of
Duke University

2023

Abstract

Improving disease treatment relies on advancements in our understanding of disease etiology and evolution. Rational drug design seeks to exploit this understanding to improve human health through targeted molecular interventions. In this dissertation, we present computational methods that 1) predict disease evolution in the form of resistance mutations; and 2) generate *de novo* D-peptide therapeutics. First, we introduce the Resistor algorithm. Resistor uses Pareto optimization with multistate design and cancer-specific mutational probabilities to rank resistance mutations based on their ability to ablate binding to an inhibitor, retain native function, and occur in a specific cancer type. We apply Resistor to 8 inhibitors targeting the EGFR, BRAF, and ERK2 proteins, and provide experimental validation of Resistor-predicted resistance mutations. Second, we introduce DexDesign, a novel algorithm for computationally designing *de novo* D-peptide inhibitors. DexDesign leverages three novel techniques that are broadly applicable to computational protein design: the Minimum Flexible Set, K*-based Mutational Scan, and Inverse Alanine Scan. We apply these techniques and DexDesign to generate novel D-peptide inhibitors of two biomedically important PDZ domain targets: CALP and MAST2. Notably, the peptides we generated are predicted to bind their targets tighter than their targets' endogenous ligands, validating the peptides' potential as lead therapeutic candidates. We provide implementations of Resistor and DexDesign in the free and open source computational protein design software OSPREY.

Dedication

To Emily and Siegfried

Contents

List of Tables

List of Figures

Acknowledgements

First and foremost, I would like to thank my advisor, Bruce R. Donald, for being an unparalleled mentor throughout my time at Duke. Under Bruce's tutelage, I've honed my ability to think critically about science, identify fertile research topics, and interact positively with the larger scientific community. His guidance and support for all of his graduate students, myself included, has contributed much to our scientific, professional, and personal growth throughout our time in his lab. Thank you, Bruce.

I would also like to thank my committee members: Alberto Bartesaghi, Raluca Gordân, and Teresa Kaserer for their generosity in discussing my research, and educating me on theirs, throughout my graduate career. They have provided excellent feedback, which has improved the quality of the research efforts I've been part of. Teresa has been not only a fantastic committee member, but also a dedicated collaborator with whom I've coauthored four manuscripts. In addition, her coordination with experimentalists at her university provided invaluable empirical feedback that helped us to iteratively refine our computational methods.

All the members of the Donald Lab deserve a big thanks, too. Thank you, Kelly, for modeling rigorous research discipline and educating me about biochemistry. Thank you, Graham, for the edifying conversations, advice, and thought-provoking questions. Thank you, Catherine, for your generous willingness to share your biochemistry knowledge. Thank you, Jeff, for setting a high bar in the engineering of OSPREY towards which we all strive. Thank you, Henry, for being a dedicated collaborator and providing detailed feedback on my writing. Thank you, Sari, for always being willing to answer questions. And thank you Hong, Jaden, and Caleb, for being fun people with whom to hang out in the lab.

None of our applied computational research would have been possible without the support of the Duke Computer Science support staff who, without exception, have managed to keep our servers up-and-running despite our best efforts otherwise. Joe Shamblin is not only the most talented IT administrator I've met, he's also the most approachable and always willing to answer questions. I can't count the number of times he has provided me with assistance over the past six years. Thank you.

Lastly, I am grateful for the funding the National Institutes of Health and Duke have provided me throughout my graduate career.

1 Introduction

Developing a new therapeutic takes an enormous amount of time and capital. One recent estimate[1] has put the average cost of bringing a drug to market in 2022 at USD $2.3 billion. Clinical trial cycles require on average 7 years to complete, ranging from a low of about 4 years in infectious disease to about 12 years in oncology. Meanwhile, the internal rate of return (a proxy for R&D productivity; *viz.* the cost to develop a drug versus the expected sales of a drug once launched) has declined from 7% in 2014 to 1.2% in 2022. Such a decline is understandable considering the rising cost and time required to bring a drug to market. There exists no silver bullet that can reverse the trend, but there are many knobs pharmaceutical companies can turn to increase R&D efficiency. One such area includes improving the computational modeling of proteins and peptides.

There is a rich history of applying computation and algorithms to biological phenomena. Entire fields of research, such as genomics, transcriptomics, metabolomics, and others, are predicated on the ability of computers to run algorithms efficiently to assist humans in the analysis and interpretation of experimental data. For example, the foundational construction of the human genome in 2001 was enabled by advances in whole-genome assembly algorithms[2]. Algorithms have since been developed for diverse biological applications such as predicting repair outcomes after CRISPR-Cas9 DNA cleavage[3], optimizing mRNA, proteins, and peptides for stability and reduced immunogenicity[4–9], docking small molecules and peptides to proteins[10–12], among others. Notably, algorithmic breakthroughs in machine learning have recently provided an extraordinary solution to the 50-year-old grand challenge of predicting protein structure from primary sequence in the form of AlphaFold[13,14]. In short, algorithms that provide for new or more accurate modeling capabilities have been one of the primary drivers of advances in biomedical fields over the past two decades. New and improved algorithms that have been applied to computational structure-based protein design (CSPD) are no exception.

1.1 Computational structure-based protein design

CSPD, in contrast to purely sequence-based protein design, incorporates experimental or theoretical models of a protein structure as its primary input. The theoretical basis for the incorporation of protein structures in algorithms is an outgrowth of the central dogma of molecular biology: that information flows forward from DNA to RNA to proteins, that a protein's amino acid sequence determines its structure, and structure determines function[15]. While the notion that a protein adopts a single, static structure is a simplification[16], it is the case that a protein's most probable conformation at any point in time is that conformation which minimizes free energy[15]. The conformation that minimizes free energy is called the *Global Minimum Energy Conformation* (GMEC).

The oversimplification noted above is better explained by statistical thermodynamics. Statistical thermodynamics teaches us that the GMEC is important, but that proteins adopt a (potentially infinite) number of conformations, and the probability of a protein being found in any one conformation c is proportional to that conformation's energy:

$$p(c) \propto \exp\left(-\frac{E(c)}{RT}\right), \qquad \text{(Equation 1)}$$

where the energy of conformation c is denoted $E(c)$, R is the ideal gas constant, and T is the absolute temperature. Since the sum of all conformations' probabilities must equal 1, the probability of $p(c)$ is:

$$p(c) = \frac{1}{Q}\exp\left(-\frac{E(c)}{RT}\right), \qquad \text{(Equation 2)}$$

where Q, the canonical partition function and normalization factor, sums over all c in the set of conformations, C:

$$Q = \sum_{c \in C}\exp\left(-\frac{E(c)}{RT}\right). \qquad \text{(Equation 3)}$$

In addition to being a normalization factor for computing the probability of a conformation, the partition function can be used to calculate additional properties of the molecular system, such as entropy and enthalpy[16]. CSPD algorithms that incorporate the notion of entropy, via the computation or approximation of partition functions, have been successfully applied to diverse protein design tasks, such as enzyme design[17–21], peptide design[22,23], design of inhibitors of protein-protein interactions[24], design of non-classical antifolates[25,26], prediction of resistance mutations[27–31], protein optimization for immunogenicity and stability[32], and design of broadly neutralizing antibodies[33,34]. The computation of provable approximations to the partition function is foundational to many of the protein design algorithms included in the protein design software developed in our lab, OSPREY[35].

1.2 OSPREY

OSPREY (Open-Source Protein REdesign for You) is a free and open-source CSPD software suite developed by our lab at Duke University[35]. OSPREY includes many

CSPD algorithms applicable to predicting protein resistance mutations[27–31], designing enzymes[17–21], inhibitors[22–24], and broadly neutralizing antibodies[33,34], predicting GMECS[36] and low-energy conformational ensembles[17,21,24,37,38] for applications such as improving or ablating a target protein's binding to a protein[24], peptide[23,25], or small molecule[28,30,31]. The predictions its algorithms made have been validated *in vitro*[19,21,24,30] and *in vivo*[33,34], both in retrospective[24,28,35] and prospective[24–26,30] scenarios. Key theoretical components that have contributed to OSPREY's algorithmic accuracy include 1) provable methods, 2) conformational ensembles, 3) continuous motions.

1.2.1 Provable algorithms

Provable methods in CSPD algorithms stand in contrast to heuristic methods that sample a protein's conformational space and lack guarantees as to the accuracy of their predicted solutions. For example, if the protein design task is to find the GMEC of a thermodynamic ensemble, a provable algorithm is guaranteed to find the GMEC (with respect to the input model, *viz.* the inputs to the algorithm: the protein structure, conformational flexibility, energy function, sidechain rotational isomers (rotamers), etc.) whereas a heuristic, sampling-based algorithm is not. The natural question follows: why one would use an algorithm that lacked guarantees to the accuracy of its answer when provable methods exist? In practice, many protein designers accept an answer that is good enough, provided they receive it fast enough.

Yet in our view this methodological choice brings with it several shortcomings that compound when applied to protein design. For one, all CSPD algorithms make necessary simplifications to its structural, thermodynamic, and energetic models to make the CSPD problem formulation tractable. Such simplifications include pairwisedecomposable energy functions, the use of discrete or continuous rotamers, implicit solvation, and limiting the regions of a protein that are flexible. Heuristic algorithms convolve these modeling simplifications with algorithmic inaccuracy. This makes it impossible to know when a solution turns out to be incorrect whether the cause was from inaccurate models or the algorithm itself. Perniciously, this precludes the protein designer from being able to identify and improve model shortcomings. These challenges increase as the protein design problem becomes larger, e.g., for the GMEC example, Simoncini et al. have shown[39] that the probability of a widely used sampling-based algorithm correctly identifying the GMEC quickly approaches 0.

Provable algorithms can guarantee the accuracy of their answers, but there is no free lunch. Finding the GMEC of a protein is NP-hard[40,41], and computing low-energy thermodynamic ensembles and their associated partition functions is #P-hard[42,43]. Nevertheless, techniques such as dead-end elimination[44], A* search[45], and branch and bound algorithms[37,38] implemented in OSPREY provide the protein designer with solutions and guarantees on accuracy that in practice run in reasonable amounts of time (Appendix A.4 provides examples of empirical OSPREY runtimes).

1.2.2 Conformational ensembles and the K* algorithm

Here, for the reader, we review the K* algorithm in the OSPREY protein design software suite[35], which we have presented and analyzed in Lilien et al. (2005)[21], Georgiev et al. (2008),[17] Donald (2011),[46] Gainza et al. (2012),[47] Hallen et al. (2018),[35] Ojewole et al. (2018) [38], and Jou et al. (2020)[37]. In brief, the K* algorithm computes a provably good ε-approximation to the binding affinity constant, K_a. For a proof that K* approximates K_a see Appendix A of Lilien et al[21]. K* does so by calculating an ε-accurate partition function for three structures: the bound protein:ligand complex (denoted PL), the unbound protein (denoted P), and the unbound ligand (denoted L). Let X be an arbitrary state, $X \in \{\, P, L, PL \,\}$. The partition function is the sum of the Boltzmann-weighted energies for all the conformations in the thermodynamic ensemble of X. Let s denote an arbitrary amino acid sequence, then the partition function of s in state X (which we donate as $q_x(s)$) is:

$$q_x(s) = \sum_{c \in Q_x(s)} \exp\left(-\frac{E(c)}{RT}\right), \qquad \text{(Equation 4)}$$

where $Q_x(s)$ is the entire conformational ensemble of sequence s in state X, and c is a single conformation from that ensemble. $E(c)$ is the energy of conformation c. R is the ideal gas constant and T is the temperature in absolute Kelvin.

The K* score for a sequence s approximates K_a:

$$K^*(s) = \frac{q_{PL}(s)}{q_P(s)q_L(s)} \ . \qquad \text{(Equation 5)}$$

K* uses minimized continuous rotamers when computing $E(c)$, as described by the iMinDEE algorithm[47]. It uses the A* algorithm[45,48] to search over $Q_X(s)$ and streams a gap-free list of conformations in order of the lower-bound of a conformation's minimized energy[47]. It then minimizes the conformation to calculate $E(c)$, and generates an ε-approximation of the partition function $q_X(s)$ and the ensemble-complete K* value. This approximation is known as the K* score.

1.2.3 Continuous motions

One common CSPD modeling simplification, used in both heuristic and provable methods, is the use of discrete backbone and sidechain conformations. The backbone is often set to be rigid, and the sidechains are restricted to a set of experimentally derived low-energy rotamers[49]. Georgiev et al. showed[17] that constraining conformational search to discrete rotamers can elide favorable conformations that would be accessible if the sidechains were allowed to flex slightly from their modal rotameric conformations.

The iMinDEE algorithm[17] in OSPREY systematizes continuous rotameric conformational search by allowing the sidechain to minimize in a continuous space within a voxel centered on the modal rotamer. This energy-minimized conformation is then used to compute minimization-aware bounds for the dead-end elimination pruning. The DEEPer algorithm[50] further loosened the discrete backbone paradigm to handle more extensive backbone flexibility and backbone ensembles. Additionally, recent versions of the K* family of algorithms implemented in OSPREY allow protein designers to specify that a ligand can translate and rotate.

1.3 Outline of dissertation

This dissertation describes novel approaches for computational structure-based protein design that provide new insights and capabilities to the drug-design process. Specifically, this dissertation presents novel algorithms for two important tasks in drug-design: 1) the prospective prediction of a protein target's ability to develop resistance to a drug via escape mutations, and 2) the design of *de-novo* D-peptide inhibitors.

Chapter 2 presents Resistor—a novel algorithm for predicting resistance mutations that may arise in a drug target. Resistor uses structure-based multistate design to predict target mutations that ablate binding to an inhibitor while retaining native function with an endogenous ligand. It then uses Pareto optimization to combine these structure-based predictions with sequence-based cancer-specific mutational probabilities to rank prospective resistance mutations. By providing a ranked list of resistance mutations, a medicinal chemist could either proactively modify a drug candidate to make it less prone to resistance, or, in the case that a drug is already in use, anticipate resistance and begin developing the next generation drug that overcomes the resistance. Appendix A provides additional details on how we applied Resistor to predict resistance mutations in EGFR, BRAF, and ERK2. Appendix C is a step-by-step instruction manual for using the OSPREY[35] implementation of the Resistor algorithm.

Chapter 3 develops DexDesign, a novel computational protocol for designing *de novo* D-peptides. The use of L-peptides as therapeutics has a number of advantages, including standard protocols for synthesis, good efficacy, high potency, and selectivity[51,52]. But L-peptide therapeutics also have a number of drawbacks, including poor stability, oral bioavailibility, membrane permeability, and retention[51]. The incorporation of D-amino acids into peptides can obviate some of these drawbacks. For example, D-peptides can increase peptide stability by decreasing the substrate recognition by proteolytic enzymes[53–56].

One goal of DexDesign is to provide the medicinal chemist with a drug design algorithm that exploits the advantages of peptide therapeutics while minimizing the downsides. DexDesign does this by using the MASTER protein substructure search algorithm[57] to identify D-peptides with backbones similar to existing L-peptide binders, and then uses OSPREY’s K* algorithm with a new D-sidechain library to optimize the

D-peptide’s binding to its target. We first describe the protocol, and then demonstrate applying DexDesign to two PDZ domain targets of biomedical importance: CALP and MAST2. Appendix B extends the discussion of the DexDesign algorithm and contains additional predictions and analyses of the complete set of DexDesign-generated *de novo* D-peptide inhibitors.

1.4 Publications

A large component of the content in this dissertation is adapted from published, peer reviewed original research. For example, the material presented in Chapter 2, Appendix A, and Appendix C are adapted from the previously published manuscripts:

1. Guerin, N., Kaserer, T. & Donald, B. R. Resistor: An Algorithm for Predicting Resistance Mutations Using Pareto Optimization over Multistate Protein Design and Mutational Signatures. in (ed. Pe’er, I.) **13278,** 387–389 (Springer International Publishing, 2022).
   (This is the initial, abbreviated version I presented at the RECOMB 2022 conference. The following three publications built upon and extended different aspects of the ideas we originally presented at RECOMB).

2. Guerin, N., Feichtner, A., Stefan, E., Kaserer, T. & Donald, B. R. Resistor: an algorithm for predicting resistance mutations via Pareto optimization over multistate protein design and mutational signatures. *Cell Systems* **13,** 830-843.e3 (2022).

3. Guerin, N., Kaserer, T. & Donald, B. R. RESISTOR: A new OSPREY module to predict resistance mutations. *Journal of Computational Biology* (2022).

4. Guerin, N., Kaserer, T. & Donald, B. R. Protocol for predicting drug-resistant protein mutations to an ERK2 inhibitor using RESISTOR. *STAR Protocols* **4,** 102170 (2023).

5. Kugler, V., Lieb, A., Guerin, N., Donald, B. R., Stefan, E. & Kaserer, T. Disruptor: Computational identification of oncogenic mutants disrupting protein-protein and protein-DNA interactions. *Commun Biol* **6,** 1–6 (2023).

Other content, such as that presented in Chapter 3 and Appendix B, showcase original research which has not yet been published:

Guerin, N., Childs, H., Zhou, P., and Donald, B.R. DexDesign: A new OSPREY-based algorithm for designing de novo D-peptide inhibitors. Submitted to journal, under review.

2 Resistor: A novel algorithm for predicting resistance mutations from structures and sequences

In 2019 Teresa Kaserer contacted our lab to ask for advice on OSPREY. We had recently read her publication in Cell Chemical Reports[28] in which she used OSPREY to predict resistance mutations in cancer systems. We had ideas about extending the protocol she laid out in that paper by using multi-objective optimization, and she had recommendations on which cancer systems to investigate. We decided to collaborate on a new algorithm, Resistor, and apply it to predicting resistance mutations in EGFR, BRAF, and ERK2. We ended up presenting Resistor at the RECOMB 2022 conference[58] and publishing three additional articles that extended Resistor: one describing the algorithm with retrospective and prospective experimental validation[30]; one on the software module in OSPREY that implements Resistor[59]; and one describing a step-by-step protocol for using Resistor[31]. This chapter is adapted from those publications.

Bruce R. Donald, Teresa Kaserer, and I worked closely together on this research. The conceptualization of the algorithm, choice of cancer and drug targets to apply it to, and analysis of the computational predictions and experimental data was a joint effort. The drafting and editing of the manuscripts were also highly collaborative. Fortuitously, Teresa introduced two experimentalists from the University of Innsbruck to our group: Stefan Eduard and Andreas Feichtner. Stefan and Andreas used their KinCon[60–62] cell-based kinase reporter assay to test and validate Resistor's predictions for our Cell Systems publication[30]. In addition to the collaborative effort described above, I implemented Resistor in the open source OSPREY software package and used it to generate all of the computational predictions we present in this chapter and Appendix A.

2.1 Introduction

Acquired resistance to therapeutics is a pressing public health challenge that affects maladies from bacterial and viral infections to cancer[63–67]. There are several different ways cancer cells acquire resistance to treatments, including drug inactivation, drug efflux, DNA damage repair, cell death inhibition, and escape mutations, among others[64]. The accurate, prospective prediction of resistance mutations could allow for the design of drugs that are less susceptible to resistance. Although it is unlikely that medicinal chemists will be able to address all the resistance-conferring mechanisms in cancer cells, progress can be made by the incorporation of increasingly accurate models of the above contributing factors to acquired resistance, leading to the development of more durable therapeutics. To that end, several structure-based computational techniques for therapeutic design and resistance prediction have been proposed.

One such technique is based on the substrate-envelope hypothesis. In short, the substrate-envelope hypothesis states that drugs designed to have the same interactions as the endogenous substrate in the active site will be unlikely to lose efficacy because any mutation that ablates binding to the drug would also ablate binding to the endogenous substrate[68]. C. Schiffer and B. Tidor's labs developed the substrate-envelope hypothesis for targeting drug-resistant HIV strains[68–71]. Their design technique has been successfully applied to develop compounds with reduced susceptibility to drug-resistant HIV proteases[71].

Another computational technique is to use an ensemble-based positive and negative designs[72,73]. There are two specific ways that point mutations can confer resistance to therapeutics: they can decrease binding affinity to the therapeutic or they can increase binding to the endogenous ligand[29,72]. Protein design with the goal of decreasing binding is known as *negative design* and increasing binding is known as *positive design*. As a concrete example, consider the case of a drug that inhibits the tyrosine kinase activity of the epidermal growth factor receptor (EGFR) to treat lung adenocarcinoma. Here, an active site mutation could sterically prevent the inhibitor from entering the active site[74]. On the other hand, a different mutation might have no effect on an enzyme's interactions with the drug but instead increase affinity to its native ligands, resulting in the increased phosphorylization of downstream substrates[75,76]. Because these two distinct pathways to therapeutic resistance exist, it is necessary to predict resistance mutations using both positive and negative design. In other words, predicting resistance can be reduced to predicting a ratio of the change in K_a upon mutation of the protein:endogenous ligand and protein:drug complexes.

K_a is an equilibrium constant measuring the binding and unbinding of a ligand to a receptor. It is defined as:

$$K_a = \frac{K_{\text{on}}}{K_{\text{off}}} = \frac{[RL]}{[R][L]}, \qquad \text{(Equation 6)}$$

where k_{on} and k_{off} are the on- and off-rate constants, and [RL], $[R]$, and $[L]$ the equilibrium concentrations of, respectively, the receptor-ligand complex, unbound receptor, and unbound ligand. K_a is the reciprocal of the disassociation constant K_d. K* is an algorithm implemented in the OSPREY computational protein design software that provably approximates K_a[17,35]. It is defined as the quotient of the bound (complex) to unbound (apo protein and apo ligand) partition functions of a protein:ligand system. See section 1.2.2 for further details on the K* algorithm.

Our lab developed a provable, ensemble-based method using positive and negative design to computationally predict and experimentally validate resistance mutations in protein targets[72]. We then applied this methodology to prospectively predict resistance mutations in dihydrofolate reductase when *Staphylococcus aureus* was treated with a novel antifolate[29], which we later confirmed *in vivo*[29,77], demonstrating the utility of correctly predicting escape mutations during the drug discovery process.

From these previous works, it is clear that multiple criteria must be combined to decide whether a mutation confers resistance. Often it is the human designers themselves who must choose arbitrary weights for different criteria. Yet, multi-objective, or Pareto, optimization techniques would allow designers to combine multiple criteria without choosing arbitrary decision thresholds. Pareto optimization for protein design has been employed by Chris Bailey-Kellogg, Karl Griswold, and co-workers[6–9,78,79]. One such example is PEPFR (protein engineering Pareto frontier), which enumerates the entire Pareto frontier for a set of different criteria such as stability versus diversity, affinity versus specificity, and activity versus immunogenicity[80]. Algorithmically, PEPFR combined divide-and-conquer with dynamic or integer programming to achieve an algorithm where the number of divide-and-conquer "divide" steps required for the search over design space is linear only in the number of Pareto optimal designs. Being dependent on multiple criteria, a multi-objective optimization method that ranks solutions, such as Pareto optimization, is particularly suitable for resistance predictions.

Instead of merely finding a single solution optimizing a linear combination of functions, Pareto optimization finds all consistent solutions optimizing multiple objectives such that no solution can be improved for one objective without making another objective worse. Specifically, let Λ be the set of possible solutions to the multi-objective optimization problem, and let $\lambda \in \Lambda$. Let $\mathcal{F}$ be a set of objective functions and $f \in \mathcal{F}$, where $f: \Lambda \rightarrow \mathrm{R}$ is one objective function. A particular solution λ is said to *dominate* another solution λ' when

$$f(\lambda) \leq f(\lambda') \text{ for all } f \in \mathcal{F}\text{, and} \qquad \text{(Equation 7)}$$

$$g(\lambda) \leq \mathrm{g}(\lambda') \text{ for at least one } \mathrm{g} \in \mathcal{F}. \qquad \text{(Equation 8)}$$

A solution λ is *Pareto optimal* if it is not dominated. Resistor combines ensemble-based positive and negative design, cancer-specific mutational signature probabilities, and hotspots to identify not only the Pareto frontier but also the Pareto ranks of all candidate sequences.

The inclusion of mutational signature probabilities in Pareto optimization is possible because distinct mutational processes are operating in different types of cancers[81,82]. Specifically, these mutational processes drive the type and frequency of DNA base substitutions. Alexandrov et al. postulated each signature to be associated with a biological process (such as APOBEC activity) or a causative agent (such as tobacco use), although not all associations are definitively known. What is certain is that particular signatures tend to appear in particular types of cancer. For example, 12 single-base substitution signatures, 2 double-base substitution signatures, and 7 indel signatures were found in a large set of melanoma samples, with many of those signatures associated with ultraviolet light exposure[82]. Building on the work of Alexandrov et al., Kaserer and Blagg combined the multiple signatures found in each cancer type to generate overall single-base substitution probabilities[28]. Resistor uses these probabilities to compute the overall probability that mutation events will occur in a gene independent of changes to protein fitness. This amino acid mutational probability is one of the axes we optimize over.

The most computationally complex part of provable, ensemble-based multistate design entails computing the K* scores of the different design states. This is largely because for biological accuracy it is necessary to use K* with continuous side-chain flexibility[47,83]. Though OSPREY has highly optimized GPU routines for continuous flexibility[35], energy minimization over a combinatorial number of sequences in a continuous space is, in practice, computationally expensive. Having a method to reduce the number of sequences evaluated would greatly decrease the computational cost. COMETS is an empirically sublinear algorithm that provably returns the optimum of an arbitrary combination of multiple sequence states[36]. Resistor uses COMETS to prune sequences whose predicted binding with the drug improves and binding with the endogenous ligand deteriorates. Although COMETS does not compute the full partition function, it provides a useful method to efficiently prune a combinatorial sequence space, for example, when investigating resistant protein targets with more than one resistance mutation. By virtue of pruning using COMETS, Resistor inherits the empirical sublinearity characteristics of the COMETS sequence search, rendering Resistor sublinear in the size of the sequence space.

The tyrosine kinase EGFR and serine/threonine-protein kinase BRAF are two oncogenes associated with, respectively, lung adenocarcinoma and melanoma. Both kinases are conformationally flexible, but two conformations are particularly determinative to their kinase activity—the "active" and "inactive" conformations. Oncogenic mutations to EGFR include L858R and deletions in exon 19, both of which constitutively activate EGFR[84,85]. Likewise, V600E is the most prevalent constitutively activating mutation in BRAF[86]. Numerous drugs have been developed to treat the EGFR L858R and BRAF V600E mutations. The first-generation inhibitors erlotinib and gefitinib competitively inhibit ATP binding in EGFR's active site, whereas binding by the third-generation osimertinib is irreversible[87–89]. For BRAF, the therapeutics dabrafenib, vemurafenib, and encorafenib were designed to target the V600E mutation and are in clinical use, and PLX8394 is in clinical trials[90–93]. The use of Resistor to predict resistance mutations to these drugs would provide strong validation of the efficacy of this approach.

By presenting Resistor, this chapter makes the following contributions:

1. A multi-objective optimization algorithm that combines four axes of resistance-causing criteria to rank candidate mutations.
2. The use of COMETS as a provable and empirically sublinear pruning algorithm that removes a combinatorial number of candidate sequences before expensive ensemble evaluation.
3. A validation of Resistor that correctly predicted eight clinically significant resistance mutations in EGFR, providing explanatory ensemble-bound structural models for acquired resistance.
4. Prospective predictions with explanatory structural models and experimental validation of resistance mutations for four drugs targeting BRAF mutations in melanoma.
5. Newly modeled structures of EGFR and BRAF bound to their endogenous ligands and inhibitors in cases where no experimental structures exist.

6. An implementation of Resistor in our laboratory’s free and open-source computational protein design software OSPREY[35].

2.2 Results

2.2.1 Overview of Resistor

The Pareto optimization in Resistor optimizes four axes: structure-based positive design, structure-based negative design, sequence-based mutational probabilities, and the count of resistance-causing mutations at a given amino acid location. Briefly, we chose these four criteria because they identify mutations that (1) increase affinity to the endogenous ligand in such a way that it outcompetes the inhibitor, (2) decrease the efficacy of the drug by reducing its binding (leading to the same effect), (3) are predicted to occur based on the DNA sequence and excludes those that are unlikely to arise, and (4) are located at residue positions where many mutations are predicted to confer resistance, thus identifying a position of relative importance. We believe these criteria to be the minimal requirements a cancer clone must fulfill to confer resistance, and we have had success predicting retrospective and prospective resistance mutations in a previous study using these four criteria[28].

In Kaserer and Blagg's earlier study[28], they prioritized potential resistance mutants by first applying four sequence- and structure-based filtering steps and then pruning the remaining predicted resistance mutations by (1) choosing the three residue locations with the highest hotspot cardinality (Section 2.2.4) and (2) ranking the individual amino acids within the hotspots by their mutational probability[28]. In other words, they ranked resistance candidates by two criteria: their hotspot cardinality and mutational probability.

With Resistor, hotspot cardinality instead becomes one of the Pareto objectives. Their earlier work used the positive and negative design K* scores as a binary resistance filter[28]. Here, we use them first as a filter and then as two additional Pareto optimization objectives. This allows Resistor to use thermodynamic predictions not only in a binary, qualitative manner (i.e., whether the ratio of K* positive and negative designs indicates resistance) but also in a quantitative manner (i.e., the magnitude of the affinity-driven resistance). Finally, Resistor also transforms mutational probability from the final ranking criteria to one of the four Pareto objectives. In summary, Resistor's Pareto optimization objective function simultaneously maximizes the ΔK_a of the positive design (the protein bound to the endogenous ligand), minimizes the ΔK_a of the negative designs (the protein bound to the drug), maximizes the mutational probability, and maximizes the count of resistance-causing mutations per amino acid. Figure 1 shows an overview of how these axes are implemented in our algorithm. It should be mentioned that, as a generalizable method, additional resistance-causing criteria could be trivially added to Resistor for further refinement.

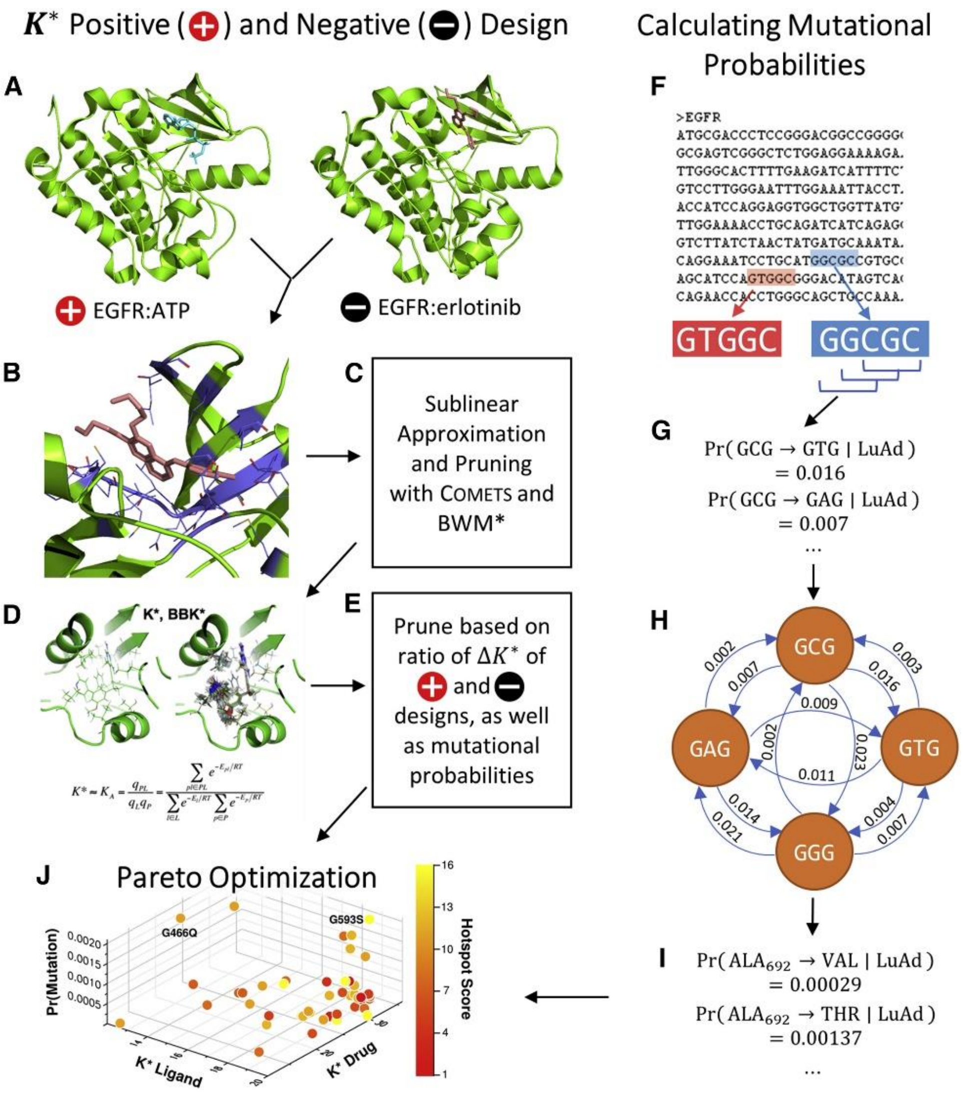

K^* Positive (+) and Negative (−) Design
A
+ EGFR:ATP
− EGFR:erlotinib
B
C
Sublinear Approximation and Pruning with COMETS and BWM*
D
K*, BBK*
$K^* \approx K_A = \frac{q_{PL}}{q_L q_P} = \frac{\sum_{pl \in PL} e^{-E_{pl}/RT}}{\sum_{l \in L} e^{-E_l/RT} \sum_{p \in P} e^{-E_p/RT}}$
E
Prune based on ratio of ΔK^* of + and − designs, as well as mutational probabilities
J
Pareto Optimization
Pr(Mutation)
0.0020
0.0015
0.0010
0.0005
G466Q
G593S
K* Ligand
14
16
18
20
K* Drug
20
Hotspot Score
16
13
10
7
4
1
Calculating Mutational Probabilities
F
>EGFR
ATGCGACCCTCCGGGACGGCCGGGG
GCGAGTCGGGCTCTGGAGGAAAAGA
TTGGGCACTTTTGAAGATCATTTTC
GTCCTTGGGAATTTGGAAATTACCT
ACCATCCAGGAGGTGGCTGGTTATG
TTGGAAAACCTGCAGATCATCAGAG
GTCTTATCTAACTATGATGCAAATA
CAGGAAATCCTGCATGGCGCCGTGC
AGCATCCAGTGGCGGGACATAGTCA
CAGAACCACCTGGGCAGCTGCCAAA
GTGGC
GGCGC
G
Pr(GCG → GTG | LuAd)
= 0.016
Pr(GCG → GAG | LuAd)
= 0.007
...
H
GCG
GAG
GTG
GGG
0.002
0.007
0.003
0.016
0.009
0.002
0.023
0.011
0.014
0.021
0.004
0.007
I
Pr(ALA_{692} → VAL | LuAd)
= 0.00029
Pr(ALA_{692} → THR | LuAd)
= 0.00137
...

Figure 1: An example Resistor workflow with EGFR. Resistor finds the Pareto frontier from OSPREY positive and negative designs, mutational probabilities, and resistance hotspots. **(A)** Two structures are required as input to OSPREY to compute positive and negative design K* scores. The structure for positive design is EGFR (green) bound to its endogenous ligand ATP (blue), for the negative design EGFR is bound to the drug erlotinib (pink). The goal of positive (resp. negative) design is to improve (resp. ablate) binding affinity. A mutation is resistant when its ratio of positive to negative K* scores increase. **(B)** All residues within 5 Å (purple) of the drug are allowed to mutate to any other amino acid. **(C)** COMETS is used as an efficient, sublinear algorithm to quickly prune infeasible mutations. BWM* is used with a fixed branch width to compute a polynomial-time approximation to the K* score. **(D)** Candidate mutations that pass the COMETS pruning step have their positive and negative K* scores computed in OSPREY. We recommend using the BBK* with MARK* algorithm as it is the fastest for computing K* scores. **(E)** Candidate resistance mutations are pruned when their ratio of positive to negative K* scores indicates a mutation does not cause resistance or if the target amino acid requires a mutation in all three DNA bases. **(F and G) (F)** Resistor computes mutational probabilities using a protein's coding DNA along with cancer-specific trinucleotide mutational probabilities for lung adenocarcinoma (abbreviated as LuAd), sliding a window **(G)** over 5′- and 3′-flanked codons. **(H and I) (H)** Resistor employs a recursive graph algorithm to compute the probability that a particular amino acid will mutate to another amino acid **(I)**. **(J)** Finally, Resistor uses Pareto optimization on the positive and negative K* scores, the mutational probabilities, and hotspot counts to predict resistance mutants.

2.2.2 Structure-based positive and negative design

We use the K* algorithm[21,47] in OSPREY to predict an ε-accurate approximation to the binding affinity (K_a) in four states: (1) the wild-type structure bound to the endogenous ligand, (2) the wild-type structure bound to the therapeutic, (3) the mutated structure bound to the endogenous ligand, and (4) the mutated structure bound to the therapeutic. This ε-accurate approximation is called the K* score[17,35]. In order to calculate the score of a protein:ligand complex, it is necessary to have a structural model of the atomic coordinates. Experimentally determined complexes have been solved for EGFR bound to an analog of its endogenous ligand (PDB ID 2itx), erlotinib (PDB ID 1m17), gefitinib (PDB ID 4wkq), and osimertinib (PDB ID 4zau) [94–97]. Similarly, we used the crystal structure for BRAF bound to dabrafenib (PDB ID 4xv2) and vemurafenib (PDB ID 3og7)[98,99]. Experimentally determined complexes of BRAF bound to encorafenib, PLX-8394, and an ATP analog in an active conformation do not exist, so we instead modeled the ligands into BRAF in its activated conformation (for additional details on model selection and preparation see Appendix A.1). We used these predicted complex structures for our resistance predictions.

We added functionality to OSPREY that simplifies the process of performing computational mutational scans. A *mutational scan* refers to the process of computing the K* score of every possible amino acid mutation within a radius of a ligand. Resistor uses this functionality to create the initial set of candidate mutant sequences by selecting and computing the K* scores for each amino acid within a 5 Å radius of the drug or the endogenous ligand. This generated a search space of 2471 sequences. We then set all residues with side chains within 3 Å of the mutating residue to be continuously flexible for the Resistor K* designs. Each sequence has an associated conformation space size dependent on the total number of mutable and flexible residues, which one can use as a heuristic to estimate the difficulty of computing a complex's partition function. The average conformation space size of each sequence was $\sim 5.9 \times 10^{10}$ conformations, thus computing the partition functions is only possible using OSPREY's pruning and provable ε-approximation algorithms[35,37,47]. Empirical runtimes of the positive- and negative-K* designs are shown in Appendix A.4. The change in the K* score upon mutation for the endogenous ligand (positive design) and drug (negative design) becomes two of the four axes of optimization. These two axes also form the basis of a pruning step (as described in Section 2.2.5).

2.2.3 Computing the probability of amino acid mutations

The genomic component of Resistor exploits mutational signatures derived from whole genome and whole exome sequencing of cancers[81,82]. A mutational signature is a distribution representing the probability that one nucleotide will mutate to another nucleotide in a given codon context and particular cancer type[81]. The different signatures are a result of diverse mutational processes[81], and different cancer types are associated with one or more mutational signatures. And although a cancer type is associated with a set of signatures, not every associated signature is found in all tumor samples of a particular cancer type. We use these empirical mutational signature data to calculate the probability that an amino acid's codon mutates to another amino acid.

Specifically, let C be the set of cancers and S the set of mutational signatures, with $c \in C$ and $s \in S$. We denote the set of signatures operative in a particular cancer c as S_c, and the proportion of tumor samples in c exhibiting signature s as W_{cs}. Let D be the set of amino acid-encoding codons and A the set of amino acids, with $d \in D$ and $a \in A$. We denote the set of codons encoding amino acid a as D_a. Last, we denote Z as the set of ways that d can mutate to any $d' \in D_a$ within two single mutational events. Then to calculate the probability that codon d mutates to amino acid a we compute:

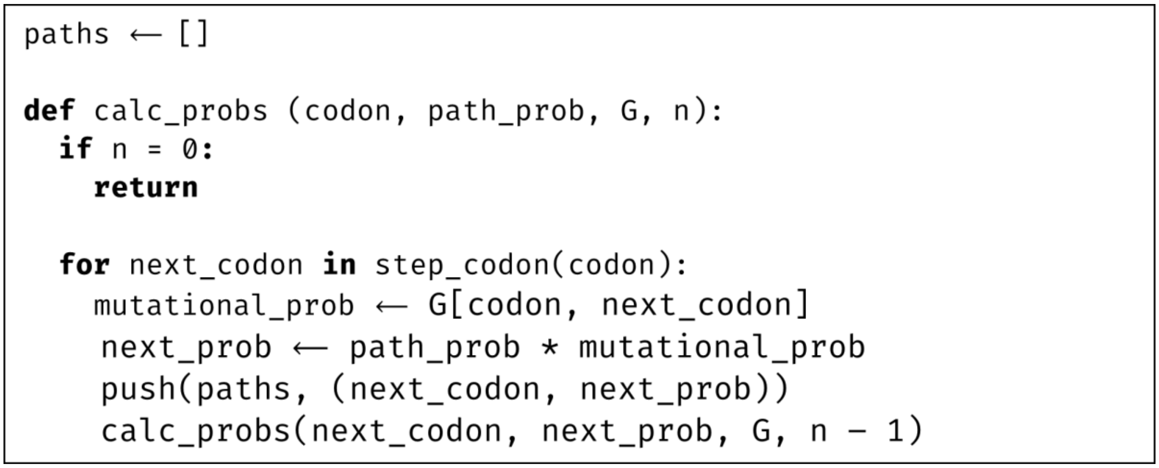
$$P(d \rightarrow D_a \mid C) = s \in S_c P(d \rightarrow D_a \mid s) W_{cs} \qquad \text{(Equation 9)}$$

$$= \sum_{s \in S_c} \sum_{d' \in D_a} P(d \rightarrow d' \mid s) W_{cs} \qquad \text{(Equation 10)}$$

$$= \sum_{s \in S_c} \sum_{z \in Z} P(z \mid s) W_{cs} \, . \qquad \text{(Equation 11)}$$

We determine Z for all amino acids and compute the values $P(z \mid s)$ using a recursive graph algorithm. For this, we construct a directed graph $G(v, e)$ for each mutational signature where the vertices v are codons and edges e connect codons that differ by their center base. The weight assigned to each edge e is the probability of one codon mutating to another codon, as provided by Alexandrov et al[81]. The input codon d must contain two flanking bases to lookup the probability of the first or last base of the codon mutating. Inputs to the algorithm are G, d, the path probability (p), and the max number of mutational steps (n). The algorithm enumerates all possible single point mutations in d in a function called `step_codon`. It looks up the probability of mutating from the current codon to the next codon using G and recursively calls itself n times. When the terminating condition is met the algorithm returns the set of codons it reached in $\leq n$ steps and their probabilities. See Figure 2 for pseudocode of the algorithm.

As evident in Figure 2, the recursive algorithm traverses the graph and find all codons that can be reached within *n* single-base mutations, where *n* is an input parameter. The algorithm then translates the target codons into amino acids and, as a final step, sums the different probabilities on each path to an amino acid into a single amino acid mutational probability (see Figure 1 F–I). One can either (1) precompute a cancer-specific codon-to-codon lookup table consisting of every $5'$- and $3'$-flanked codon to its corresponding amino acid mutational probabilities or (2) read in a sequence's cDNA and compute the mutational probabilities on the fly. The benefit of (1) is it only needs to be done once per cancer type and can be used on an arbitrary number of sequences. On the other hand, when assigning mutational probabilities to proteins that have strictly fewer than 4^5 amino acids, it is faster to compute the amino-acid-specific mutational signature on the fly. In both cases, the algorithm is strictly polynomial and bounded by $O(kn^9)$, where k is the number of codons with flanking base pairs (upper-bounded by 4^5) and n is the number of mutational steps allowed, which in the case of Resistor is 2. An implementation of this algorithm is included in the free and open-source OSPREY repository on GitHub[35].



```
paths ← []

def calc_probs (codon, path_prob, G, n):
  if n = 0:
    return

  for next_codon in step_codon(codon):
    mutational_prob ← G[codon, next_codon]
    next_prob ← path_prob * mutational_prob
    push(paths, (next_codon, next_prob))
    calc_probs(next_codon, next_prob, G, n − 1)
```

Figure 2: An algorithm for calculating mutational probabilities. calc_probs computes the complete set of paths that can be reached within $\boldsymbol{n}$ mutational steps from codon. The parameter path_prob is the probability of reaching the current codon via a particular path. After calc_probs is executed, the codons reached by all paths and their associated probabilities are in the paths variable. The codons in this variable are then grouped and summed by the amino acid they encode (omitted below). The initial invocation of calc_probs initializes path_prob to 1. The step_codon function produces all 9 variants of a codon with a single mutated base.

2.2.4 Identifying mutational hotspots

After calculating the positive and negative change in affinity ΔK_a and determining the mutational probability of each amino acid, Resistor prunes the set of candidate mutations (see section 2.2.5). Post pruning, it counts the number of mutations at each amino acid location. This count is necessary to determine whether a residue location is likely to become a "mutational hotspot," namely a residue location where many mutations are predicted to confer resistance. Correctly identifying mutational hotspots is vital because they indicate that a drug is dependent on the wild-type identity of the amino acid at that location, and it is likely that many mutations away from that amino acid will cause resistance. Consequently, the fourth axis used in Resistor’s Pareto optimization is the count of predicted resistance-conferring mutations per residue location, termed "hotspot cardinality."

2.2.5 Reducing the positive prediction space

Prior to carrying out the multi-objective optimization to identify predicted resistance mutations, we prune the set of candidates. First, we introduce a cutoff based on the ratio of K^* scores of positive and negative designs, an adaption from Kaserer and Blagg[28]. We determine the average of the K^* scores for the drug and endogenous ligand across all of the wild-type designs for the same protein. The cutoff c is:

$$c = \frac{c_0\, K_{\bar{L}}^*}{K_{\bar{D}}^*}\,, \qquad \text{(Equation 12)}$$

where c_0 is a user-specified constant, $K_{\bar{L}}^*$ is the average of the K^* scores for the wild-type protein bound to the endogenous ligand, and $K_{\bar{D}}^*$ is the average of the K^* scores for the wild-type protein bound to the drug. We recommend in practice to set c_0 to be greater than the range $(K_{max}^* - K_{min}^*)$ of wild-type K^* scores—we set it to 100 for the tyrosine kinase inhibitor (TKI) predictions. (In the future, c_0 could be learned from running Resistor on a resistance mutation dataset for homologous systems and examining the K^* scores). A mutation m is predicted to be *resistant* when:

$$\frac{K_L^*(m)}{K_D^*(m)} > c\,, \qquad \text{(Equation 13)}$$

Where $K_L^*(m)$ is the K^* score of the endogenous ligand bound to the mutant, and $K_D^*(m)$ is the K^* score of the drug bound to the mutant.

We also prune mutations predicted to completely ablate endogenous ligand binding, i.e., the predicted K^* score of the protein and endogenous ligand is 0, because such a mutation renders a critical protein non-functional. This is particularly detrimental to a cancer cell, which relies heavily on the activity of a protein. We lastly prune the predicted resistance mutation candidates by removing all mutations that cannot arise within two DNA base substitutions. Whether an amino acid can be reached within two DNA base substitutions is determined by the algorithm described in Section 2.2.3, and if it cannot, then that particular mutation is assigned a mutational probability of 0 and pruned.

2.2.6 Resistor identifies 8 known resistance mutations in EGFR

We evaluated a total of 1,257 sequences across the three TKIs for EGFR. Among these sequences, the average conformation space size for computing a complex's partition function was $\sim 1.3 \times 10^7$. After we ran the Resistor algorithm on these sequences, a total of 108 mutants were predicted as resistance-conferring candidates for all three inhibitors combined from a purely thermodynamic and probabilistic basis, i.e., these mutations were required to lower the affinity of the drug in relation to the endogenous ligand (K* positive and negative design, Figure 1 A–D) and could be formed in patients by less than three-base pair exchanges (calculating mutational probabilities, Figure 1 F–I). To further prioritize mutations and identify those that are most likely to be clinically relevant, we then computed the Pareto frontier over the four axes for each drug (Figure 1 J). Out of these 108 candidates, Resistor correctly prioritized eight clinically significant resistance mutants, with 7 of the 8 in the Pareto frontier of the corresponding inhibitor and the remaining mutant in the 2nd Pareto rank (see Table 1). The full set of predictions are in Table 4 – Table 6 on pages 128-131. A detailed description of the result for each inhibitor is included in the sections below.

Table 1: Resistor correctly identified 8 resistance mutations in EGFR to erlotinib, gefitinib, and osimertinib. For osimertinib, G796R, G796S, G796D, and G796C were on the Resistor-identified Pareto frontier. L792H was in the 2nd Pareto rank. For erlotinib, both T790M and G796D were on the Pareto frontier. For gefitinib, T790M was also on the Pareto frontier. Previous studies have documented all of these resistance mutations as occurring in the clinic[100–107]. [a] Indicates that Resistor predicted the mechanism of resistance to be decreased binding of the drug to the mutant. Note that these predicted mechanisms are only attributed here if the predicted change in the $\log_{10} \Delta K^*$ score ≥ 0.5. [b] Indicates that Resistor predicted the mechanism of resistance to be improved binding of the endogenous ligand to the mutant.

Osimertinib	Erlotinib	Gefitinib
L792H[a]	T790M[a,b]	T790M[a,b]
G796R[a,b]	G796D[a]	-
G796S[a]	-	-
G796D[a]	-	-
G796C[a]	-	-

2.2.7 EGFR treated with erlotinib and gefitinib

Of the 462 sequences evaluated for the TKI erlotinib, Resistor identified 50 as candidate resistance mutations. Pareto ranking placed 19 sequences on the frontier, 13 sequences in the second rank, and 11, 6, and 1 sequences in the third, fourth, and fifth ranks, respectively. Resistor correctly identified two clinically significant mutations, T790M and G796D, as being on the Pareto frontier[100,101]. This is concordant with empirical data showing that T790M is, by far, the most prevalent resistance mutation that occurs in lung adenocarcinoma treated with erlotinib[108]. Similarly, for gefitinib, Resistor evaluated 438 sequences and identified 22 as candidate resistance mutants. The most relevant clinical mutant, T790M, is found on the Pareto frontier.

2.2.8 EGFR and osimertinib

Resistor evaluated 357 OSPREY-predicted structures of EGFR bound with osimertinib and EGFR bound with its endogenous ligand. Of those, 36 were predicted as resistance candidates. Pareto optimization placed 16 sequences on the frontier, 2 sequences in rank 2, 8 sequences in rank 3, 1 sequence in rank 4, and 5 sequences in rank

5. Resistor correctly identified five clinically significant resistance mutations to osimertinib: L792H, G796R, G796S, G796D, and G796C[102–107], and while L792H was in the 2nd Pareto rank, all of the other correctly predicted resistance mutations are on the Pareto frontier.

Two osimertinib resistance mutations stand out: L792H and G796D (see Figure 3). Both of these mutants have appeared in the clinic[102–104,107]. OSPREY generated an ensemble of the bound positive and negative complexes upon mutation, providing an explanatory model for how resistance occurs. In both cases, the mutant side chains are much bulkier than the wild-type side chain (Figure 3A and 5D) and thus are predicted to clash with the original osimertinib binding pose (Figure 3B and 5E). Consequently, in both cases, the ligand is predicted to translate and rotate to create additional space for the mutant side chains (Figure 3C and 5F). We hypothesize that this movement weakens the other molecular interactions osimertinib makes in the EGFR active site.

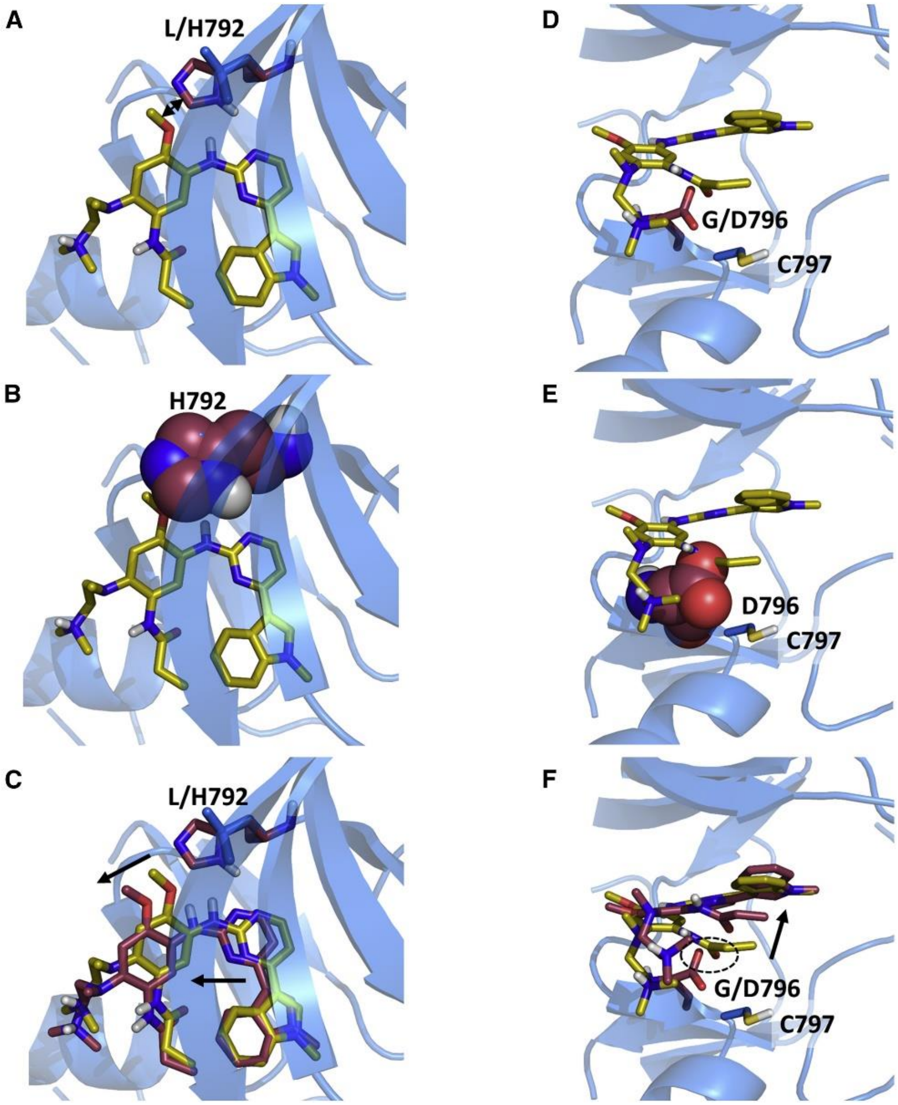


Figure 3: Structural models predicted by OSPREY agree with experimental data and explain mechanisms of osimertinib resistance to EGFR mutations L792H and G796D. Structural models predicted by OSPREY of EGFR wild-type (blue) and resistance mutations (red) bound to osimertinib (yellow sticks). The histidine (A) and glutamate (D) side chains (red sticks) in the EGFR L792H **(A)** and G796D **(D)** mutations are bulkier than the wild-type leucine **(A)** and glycine **(C)** residues (blue sticks). They clash with osimertinib in its original binding pose as highlighted by the sphere representation in **(B)** and **(E)**. **(C** and **F)** To allow for the accommodation of osimertinib in the modeled EGFR mutant structures (red sticks), the inhibitor's position within the binding pocket moves from the experimentally determined binding pose (yellow sticks). Movements are indicated by black arrows. **(F)** In case of the G796D mutation, the carboxylate moiety of D796 is predicted to be in close proximity to the osimertinib amide oxygen (highlighted with the dashed circle), thus leading to electrostatic repulsion. This mutation site is adjacent to C797, which reacts with the allyl-moiety of osimertinib to form a covalent bond in the wild type. Due to the steric and electrostatic properties of the G796D mutant, the allyl group is located further away from C797 in the model, thus preventing covalent bond formation. The movement of the allyl group is indicated by the black arrow.

In the case of G796D, there are additional factors that contribute to acquired resistance. First, the mutation to aspartate introduces a negative charge, which probably leads to electrostatic repulsion with the carbonyl oxygen of the osimertinib amide (Figure 3F, highlighted with a dashed oval). In addition, the exit vector of the hydrogen bound to the amide nitrogen does not allow a hydrogen bond with the aspartate. Second, the allyl group of osimertinib must be in close proximity to C797 for covalent bond formation. In fact, C797 is so important to osimertinib's efficacy that mutations at residue 797 confer resistance[109,110]. Even if osimertinib still binds to G796D, the allyl group would have to move away from C797 (Figure 3F, highlighted with a black arrow). This would prevent covalent bond formation and thus reduce the efficacy of osimertinib considerably. Lastly, it is likely that the mutation away from glycine reduces the conformational flexibility of the loop, incurring an entropic penalty while also plausibly making it more difficult to properly align osimertinib and C797.

2.2.9 Resistor predicts previously unreported resistance mutations in BRAF and provides structural models

In addition to retrospective validation by comparison with existing clinical data for EGFR, we used Resistor to predict how mutations in the BRAF active site could confer resistance. Specifically, we used Resistor to predict which of the 1,214 BRAF sequences would be resistant to four kinase inhibitors—vemurafenib, dabrafenib, encorafenib, and PLX8394. On the Pareto frontier for vemurafenib are 13 mutations, for dabrafenib are 16 mutations, for encorafenib are 15 mutations, and for PLX8394 are 15 mutations. The full sets of predictions are included in Table 4 –

Table 10 on pages 128-142. To validate Resistor's predictions, we compared them with two sources of experimental data: a saturation mutagenesis variant effect assay from Wagenaar et al. and a cell-based kinase conformation reporter assay termed KinCon[60,61]. Furthermore, Stefan Eduard and Andreas Feichtner carried out additional KinCon experiments on a number of Resistor predictions to validate Resistor's predictive capabilities.

2.2.10 Retrospective and prospective validation of Resistor predictions using the BRAF-KinCon biosensor reporter

KinCon, developed by Stefan and colleagues, is an in-cell protein-fragment complementation assay (PCA) that provides a readout of the activity conformation change of full-length BRAF upon mutation or exposure to different inhibitors[111]. KinCon's bioluminescence assay functions by appending parts of a luceriferase enzyme to the N- and C-termini of full-length BRAF and observing the amount of bioluminescence, indicating whether BRAF favors an open, catalytically active or a closed, autoinhibited conformation[111] (see Figure 4A). Stefan and colleagues have demonstrated that activation of BRAF either via upstream regulators such as EGFR and GTP activated Ras or via tumorigenic mutations cause BRAF to favor an open conformation[60,61]. The inhibitors bind to BRAF in the ATP binding site and cause BRAF's N and C termini to interact, shifting BRAF back toward a more closed, intermediate state[60,61,111] (see Figure 4A). This implies that for inhibitor binding and BRAF closing to occur, a mutation (or a combination of mutations and/or upstream signaling events) needs first to induce an open conformation. Not all clinically observed BRAF mutations cause opening, even if they activate the MAPK pathway[61,112] (e.g., L472C). In the same vein, not all BRAF resistance mutants show increased kinase activity, in fact, several are classified as kinase impaired[61,112,113]. One prominent mutation that shows both increased kinase activity and induces an open conformation is V600E (Figure 4B). Inhibitor treatment shifts the V600E conformational equilibrium toward a more closed state[60,61]. By contrast, the gatekeeper mutations T529M and T529I do not confer the opening of the kinase conformation and are thus insensitive to inhibitor treatment[60]. However, in combination with V600E, these mutations do confer resistance to BRAF inhibitors to varying degrees. Given that we model a state that is permissive of ligand binding at the outset (i.e., the ligand-bound BRAF complex), our Resistor calculations align very well with the reported KinCon measurements of double mutants (e.g., V600E/T529M and V600E/T529I; see Appendix A.1 for additional information on modeling).

Specifically, the Resistor predictions of resistance concord with the previous KinCon biosensor results for V600E/T529M and V600E/T529I for three of the four inhibitors: vemurafenib, dabrafenib, and PLX8394[60]. In the case of vemurafenib treatment, the proportion of open to closed conformations in the V600E/T529I mutant is not significantly different from the untreated V600E mutant, indicating vemurafenib treatment is not closing the conformational distribution in the double mutant[60]. These data agree with the Resistor calculation of the ratios of the $\log_{10}$ K* scores, which predict that both double mutants are resistant to vemurafenib, with V600E/T529M more resistant. Treatment of BRAF with PLX8394 follows the same pattern as vemurafenib, namely the V600E/T529I mutant’s closed population increases only 1.2-fold compared with the untreated mutant, and the PLX8394-treated V600E/T529M mutant does not noticeably alter the conformational distribution[60]. By contrast, the PLX8394-treated V600E mutant's closed population increases ~ 3-fold compared with the untreated population, indicating V600E sensitivity to PLX8394 (see Figure 4C). Resistor correctly predicted the V600E/T529I and V600E/T529M double mutants are resistant to PLX8394, with the change in the ratio of the $\log_{10}$ K* scores of the two mutants suggesting that V600E/T529M confers greater resistance. In the case of dabrafenib, the treatment of the V600E/T529I mutant closed the conformational distribution (2.4-fold more closed compared with untreated) more than the treatment of the V600E mutation (2-fold more closed compared with untreated), whereas the dabrafenib treatment of the V600E/T529M mutant increased the closed conformational population less effectively than the V600E mutant alone (1.4- versus 2-fold). This again agrees with the Resistor predictions, namely that V600E/T529I remains sensitive to dabrafenib but V600E/T529M is resistant. Resistor predicted that the V600E/T529I and V600E/T529M mutants would be resistant to encorafenib, but the KinCon data indicates that these mutants may actually retain sensitivity to encorafenib, as the inhibitor induces BRAF's closed state.

In addition, all inhibitors except dabrafenib were predicted to be sensitive against the G466V mutation and showed the closing of the kinase conformation[61]. However, in the case of dabrafenib, the response was comparable with vemurafenib, although vemurafenib was classified as sensitive. Previous KinCon experiments have also shown that G466V (and G466R and G466E[113], see below) impaired kinase function consistent with the reduced endogenous ligand binding predicted by Resistor[61].

In addition to the above retrospective validation, we chose a few Resistor-predicted mutations and evaluated them using the KinCon reporter. We selected the mutants G466E, G466R, V471F, L505H, and G593D because they were prioritized by Resistor for at least one of the investigated inhibitors and were reported as patient mutations in either the COSMIC[108] or cBioPortal[114,115] databases, using the curated set of non-redundant studies (see Table 2).

The expression-normalized basal biosensor signal suggests that both G466E and G466R mutants shift the conformation to an opened state, comparable with the highly oncogenic V600E variant and similar to the effect of the common non-small-cell lung cancer mutation G466V[61]. The V471F, L505H, and G593D mutations, by contrast, did not appear to induce a change in the active conformation (Figure 4B). When exposed to BRAF inhibitors (Figure 4C), G466E and G466R mutants showed the highest fold increase of the biosensor signal for all four inhibitors tested. The majority of inhibitors, three out of four, were predicted as sensitive against these mutants. Resistor predicted G466E and G466R to be resistant to dabrafenib, and although Resistor predicted dabrafenib had lower sensitivity compared with encorafenib and PLX8394 (which is consistent with the KinCon results), dabrafenib-treated mutants shifted to a closed conformation at least as much as vemurafenib-treated mutants did. The L505H and G593D KinCon mutants were not affected by any inhibitors, as those mutations do not shift the kinase into an active opened kinase conformation that is required for inhibitor binding. Although vemurafenib and dabrafenib do not appear to affect the V471F mutant, encorafenib and PLX8394 did induce a closing of the kinase, suggesting that the structural properties of the inhibitor determine the binding affinity to this mutant. This is particularly intriguing, given that the V471F mutation was selected because we predicted it would confer resistance to encorafenib and PLX8394. Although the KinCon results suggest that these two compounds still retain binding to the V471F mutant, the mutant itself did not induce a significant opening of the kinase confirmation required for ligand binding. For the latter three mutations (i.e., L505H, G593D, and V471F), it would therefore be required to induce the open conformation some other way, for example, by introducing the V600E mutation similar to T529I and T529M described above, to investigate whether resistance would develop to the inhibitors[60].

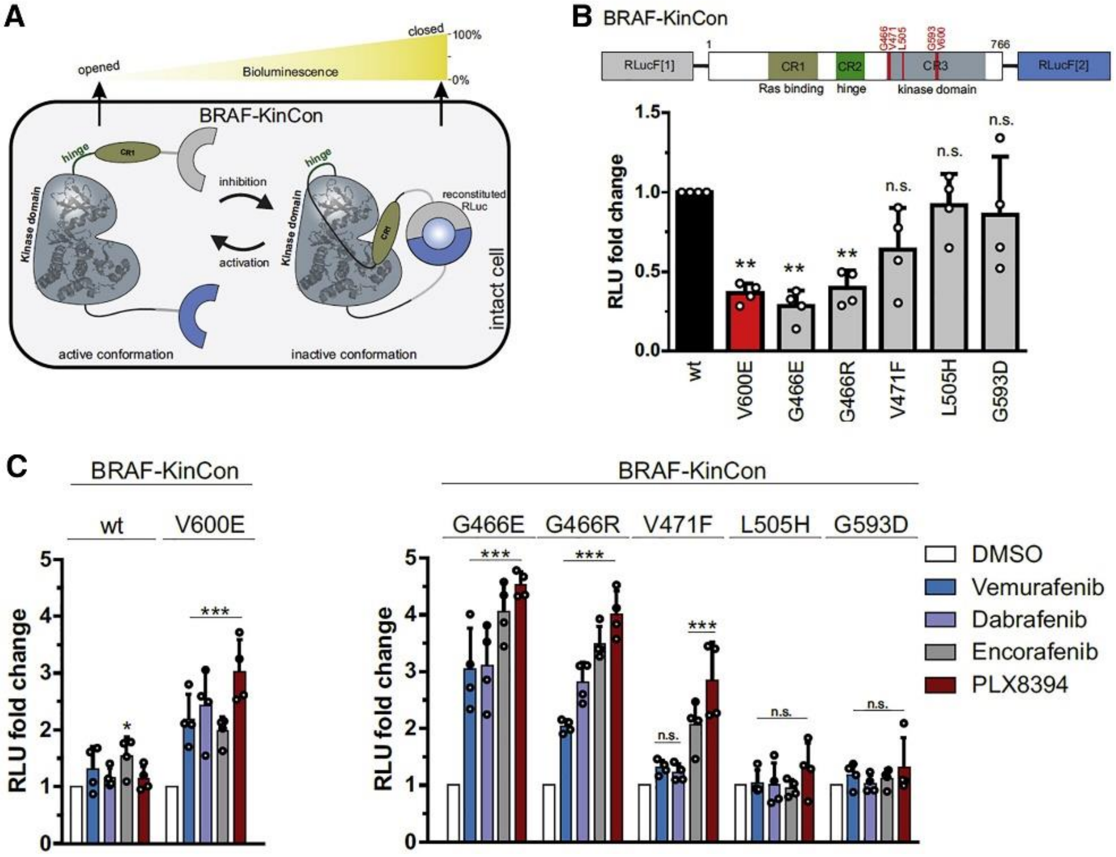


Figure 4: KinCon biosensor results for Resistor-predicted mutants. (A) Schematic depiction of Renilla luciferase (RLuc; F1, fragment 1; F2, fragment 2) PCA-based BRAF kinase conformation (KinCon) reporter system. Conformational rearrangement of the reporter upon (de)activation of the kinase is indicated. Closed kinase conformation induces complementation of Rluc PCA fragments resulting in increased Rluc-emitted bioluminescence signal. **(B)** Domain organization of the BRAF-KinCon reporter (top) and basal bioluminescent signals of the BRAF-wt (black), V600E (red), and Resistor-predicted mutant (gray) KinCon biosensors. Bars represent the mean signals, relative to BRAF-wt, in relative light units (RLUs) with SD of four independent experiments (nodes). Raw bioluminescence signals were normalized on reporter expression levels, determined through western blotting. Asterisk indicates the level of significance versus the wild-type BRAF biosensor. **(C)** BRAF-KinCon biosensor dynamics induced via treatment with respective BRAFi (1 μM for 1 h) prior to bioluminescence measurement. BRAF-wt and V600E KinCon variants serve as the control (left). The Resistor-predicted mutants are shown in a separate bar chart (right). Bars represent the mean signals, relative to the DMSO control, in relative light units (RLUs) with SEM of four independent experiments (nodes). All experiments were performed in HEK293T cells 48 h post transfection. $*p < 0.05$; $**p < 0.01$; $***p < 0.001$; n.s., not significant by t test.

Table 2: Prioritized BRAF mutations selected for experimental testing. We selected these mutants because they were prioritized by Resistor for at least one of the investigated inhibitors and were reported as patient mutations in either the COSMIC or cBioPortal databases. The numbers in the first four columns indicate the Resistor-predicted Pareto rank with melanoma mutational probabilities. The numbers in the last two columns indicate the number of patient samples containing the mutation reported in the respective database (access date 01/12/2022). The absence of a Pareto rank indicates Resistor predicted the mutant would remain sensitive to the drug.

Mutation	Vemurafenib	Dabrafenib	Encorafenib	PLX8394	COSMIC	cBioPortal
G466E	-	1	-	-	49	31
G466R	-	1	-	-	17	3
V471F	-	-	2	3	5	2
L505H	-	-	3	-	8	10
G593D	1	1	1	1	4	0

2.2.11 Retrospective validation of Resistor predictions using BRAF saturation mutagenesis experiments

Wagenaar et al. examined the effects of BRAF inhibitor binding site mutations on inhibitor efficacy[62]. To do so, they carried out targeted saturation mutagenesis on the BRAF vemurafenib binding site in the A375 human melanoma cell line and challenged the mutants with vemurafenib over a 3-week period[62]. They then sequenced the emergent clones and measured the IC_{50} values of a subset of the mutants. Their work demonstrated a correlation between a mutant's deep sequencing enrichment, i.e., the increase in the amount of an amino acid sequence in a sample before and after the addition of an inhibitor, and its IC_{50} value[62]. We, therefore, compared their enrichment data with the Resistor predictions and determined Resistor's vemurafenib resistance prediction specificity to be 91%. There were five Resistor-predicted resistance mutations that had increased enrichment over the 3-week period: T529M already discussed above (enriched 47.96-fold above the V600E baseline, which was the experiment's largest change in enrichment), T529L (enriched 18.57-fold above baseline), T529F (enriched 7.87-fold above baseline), G593I (enriched 4.84-fold above baseline), and L514E (enriched 3.73-

fold above baseline). Furthermore, Wagenaar et al. determined the relative IC_{50} values of T529M, T529L, and G593I that were, respectively, 2.05, 2.16, and 3.19 times larger than the IC_{50} value for vemurafenib applied to the V600E mutant. The IC_{50} values of T529F and L514E were not determined.

To further elucidate the molecular mechanisms conferring resistance to the G593I and L514E mutants, we analyzed the OSPREY-predicted structural models. Although neither mutant requires a movement of vemurafenib (Figure 5A) akin to what was observed in the EGFR and osimertinib structures (Figure 3), the mutations still lead to a loss of favorable interactions and/or the introduction of energetically unfavorable contacts. The residue G593 (Figure 5B) may facilitate structural adaptions required for BRAF to accommodate the vemurafenib propyl sulfonamide moiety in the rear of the ATP binding site and the G593L mutations may thus constrain the flexibility of this loop region. In addition, the leucine side chain may project near the fluoro-substituted central phenyl ring and introduce steric clashes (Figure 5C). The neighboring D594 backbone interacts with the vemurafenib sulfonamide nitrogen (Figure 5B), and this interaction would be weakened in the G593L mutant. Furthermore, residue L514 makes a range of hydrophobic contacts with vemurafenib (Figure 5D), including the central phenyl ring and the propyl chain, which are lost in the L514E mutant (Figure 5E).

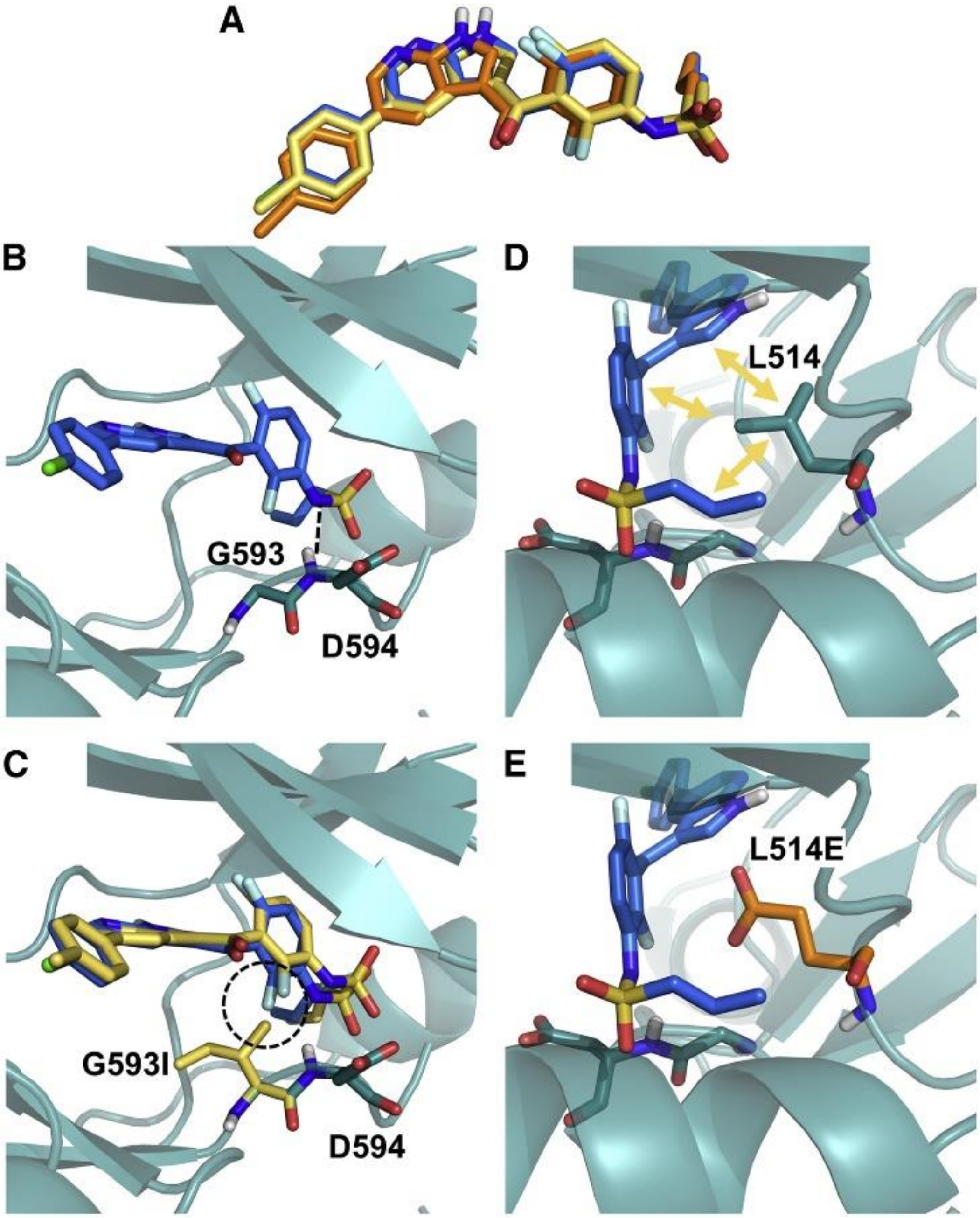


Figure 5: Structural analysis of BRAF mutations G593I and L514E. (A) No major movements were required for vemurafenib to bind to the G593I (yellow) and L514E (orange) mutation in comparison with the wild-type binding pose (blue). **(B)** BRAF G593 is located on the N terminus of the activation loop and may facilitate conformational changes required to accommodate the vemurafenib propyl sulfonamide moiety in the back of the pocket. The backbone of the neighboring D594 residue interacts with the sulfonamide nitrogen of vemurafenib as indicated by black dashed lines. **(C)** Mutation of G593 to L not only restricts the flexibility of the loop but also puts the leucine side chain in too close proximity to the fluoro-substituted phenyl ring (highlighted with the dashed circle). **(D** and **E) (D)** Residue L514 is involved in a variety of hydrophobic contacts with vemurafenib (indicated by yellow arrows), which are lost in the L514E mutant **(E)**.

2.2.12 Complexity

There are several distinct steps in Resistor, each of which has its own complexity. Although there are sublinear K* algorithms, such as BBK*[38] with MARK*[37], these algorithms so far have only been applied to positive and negative design with optimization of specific multiple objectives, such as minimizing/maximizing the bound (respectively unbound) state partition functions and their ratios for computing binding affinity or stability. COMETS[36] provably does multistate design optimizing arbitrary constrained linear combinations of global minimum energy conformation (GMEC) energies, but COMETS does not model the partition functions required for calculating binding affinity. A provable ensemble-based algorithm analogous to COMETS for arbitrary multistate design optimization is yet to be developed. Thus, general multistate K* design remains, unfortunately, a problem linear in the number of sequences and thus exponential in the number of mutable residues.

Computing K* itself, as a ratio of partition functions built from the thermodynamic ensembles of the bound to unbound states, can be expensive[43,116,117]. In order to reduce the number of K* problems to solve, COMETS is employed as a pruning mechanism for all sequences in which there are more than one mutation. Without COMETS, Resistor would need to compute sN K* scores, where s is the number of states and N is the number of sequences. With COMETS, Resistor is able to avoid computing many of these K* scores, as COMETS has been shown in practice to reduce the number of required GMEC calculations by over 99% and to reduce N for continuous designs by 96%, yielding an overall speedup of over 5×10^5-fold[36]. Since in this study we considered only single residue mutations, we omitted the COMETS pruning step, but in any use of Resistor that considers multiple simultaneously mutable residues, we believe COMETS' empirical sublinearity will make the difference between feasible and infeasible searches.

Moreover, by using an approximation containing fixed partition function size and sparse residue interaction graphs, we can use the BWM* algorithm[118] to compute the K* scores in time $O\left(nw^2q^{\frac{3}{2}w} + kn\, log\, q\right)$, where w is the branch width and q the number of rotamers per residue. When we have $w = O(1)$, this is polynomial time. In this study, we found that the ε-approximation algorithms using adaptively sized partition functions, such as BBK* with MARK*, were fast enough (see Figure 15 on page 122). However, for larger problems, the sparse approximations allow us to approximate the necessary K* scores for resistance prediction in time exponential only in the branch width and thus polynomial time for fixed branch widths.

2.3 Discussion

In this work, we report Resistor, a computational algorithm to systematically investigate protein mutations and identify those that have a high likelihood of lowering drug potency in comparison with native substrates. In addition, we analyze the probability that such a mutation is generated in cancer patients and thus likely of clinical importance. Our algorithm applies the power of Pareto optimization to resistance predictions, which provides an objective way of prioritizing the most relevant mutations for experimental testing. In addition, we used computationally predicted input structures of ligand-target complexes whenever experimental data was lacking. This broadens the targets on which Resistor can be used, as we have found that the availability of high-resolution experimental ligand-target structures still can present a major bottleneck in computational protein design.

We have applied Resistor to two case studies, EGFR and BRAF, in a retrospective manner and, in case of BRAF, also included prospective experimental data for validation. In EGFR and BRAF, the algorithm correctly identified resistance mutations. Using the vemurafenib data from Wagenaar et al[62], which is the most comprehensive dataset on BRAF mutations and vemurafenib resistance available, we determined Resistor's vemurafenib resistance prediction specificity and sensitivity to be 91% and 31%, respectively. In a data-rich setting such as proteomics (e.g., Lilien et al., 2003)[119], the sensitivity could be regarded as low. However, the prediction of antineoplastic resistance mutations is a sparse data problem. Comprehensive datasets on drug resistance mutations on specific targets are virtually non-existent. We speculate that the reason for this can be found in the large number of individual mutants that must be generated and tested. For example, in our study, we used Resistor to investigate 462, 438, and 357 individual mutants for erlotinib, gefitinib, and osimertinib, respectively. Although this is computationally feasible, it far exceeds the testing capacities of most experimental groups. Clinical resistance data is even more limited. Furthermore, even for those mutations that have been confirmed to confer clinical resistance in patients, the underlying molecular mechanisms often remain uninvestigated.

Resistor prioritizes escape mutations causing ablation of inhibitor binding and/or tighter substrate binding (the latter as a proxy for K_M). However, mutations affecting the drug target could also mediate resistance via other molecular processes, such as altering the stability of conformational states or affinity of protein-protein interactions[66,120]. One limitation of this study is that we modeled BRAF in its active conformational state. As Röck et al. showed[60], BRAF inhibitors exhibited differences in specificity and efficacy by shifting BRAF’s conformational probability distribution from an open and active to a closed, inactive state. It is plausible that mutations far from the active site could destabilize the closed, inactive state and shift the conformational probability distribution back toward the open, active state. The modeling of the large allosteric destabilization of the inactive conformations has been discussed extensively in our previous work[20,121], but its integration into Resistor is left for future work.

In addition, clinical resistance is caused by several different mechanisms of which the relative importance of escape mutations can vary greatly. In some kinases, such as c-Abl, EGFR, and FLT3, active site escape mutations are the main cause of acquired resistance[122]. In other kinases, such as BRAF, escape mutations are not the main mechanism of acquired resistance[123]. Rather, splice variants, amplification, and mutations in related genes such as N-RAS, MEK1, MEK2, IGF-1R, and AKT comprise the majority of cases of clinical resistance[123]. From this perspective, the specificity of Resistor for BRAF and vemurafenib is remarkable, and the sensitivity is in line with the fraction of resistance mutations whose etiology definitively escaped via active site mutation.

We believe that the remaining gap can be closed in future work by modeling additional conformational flexibility, kinetics, and the protein-protein interactions of additional effectors. Yet, despite these limitations, Resistor is able to prioritize mutations that are demonstrated to confer resistance in patients. Specifically, our results show that detailed and combinatorial thermodynamic computations can form the basis for predicting escape mutations to TKIs. In the future, since some resistance mutations exploit kinetic phenomena, kinetics could be incorporated for a more comprehensive model.

2.4 Conclusions

Resistor contributes to the science of predicting resistance mutations by providing an algorithm to enumerate the entire Pareto frontier of multiple resistance-causing criteria. By categorizing predicted resistance mutations by their Pareto rank, it allows the drug discovery community to prioritize escape mutations on the Pareto frontier. Resistor also provides structural justification for the mechanism of each predicted escape mutation by generating an ensemble of predicted structural models upon mutation. In this study, we have applied Resistor to predict resistance mutations in EGFR and BRAF for several different therapeutics. We demonstrate that Resistor can also be applied to computationally generated input structures, although the accuracy of the results may be somewhat diminished compared with experimentally determined structures of target-ligand complexes. However, computationally derived models can still provide useful insights, especially when considering that the availability of experimental structures appears as a major bottleneck. Although Resistor as described herein optimizes over 4 objectives, as a general method, any number of diverse objectives could be added. Resistor can be applied not only to cancer therapeutics but also to antimicrobial or antiviral drug design. It is our hope that the drug discovery community can use Resistor to design drugs that are less prone to resistance.

3 DexDesign: A new OSPREY-based algorithm for designing *de novo* D-peptide inhibitors

In this chapter I present DexDesign, a novel OSPREY-based algorithm for computationally designing de novo D-peptide inhibitors. The work in this chapter was carried out in collaboration with Henry Childs, Pei Zhou, and Bruce R. Donald. We have written a manuscript which is currently under review. This chapter is based on that manuscript:

> Guerin, N., Childs, H., Zhou, P., and Donald, B.R. DexDesign: A new OSPREY-based algorithm for designing de novo D-peptide inhibitors. Submitted to journal, under review.

Like the development of the Resistor algorithm (described in Chapter 2), the research presented in this chapter was carried out in a highly collaborative fashion. Pei and Bruce initially conceived of the idea of using MASTER[57], geometric reflections, and the K* algorithm to generate de novo D-peptide binders. I explored the feasibility of implementing a D-amino acid conformation library in OSPREY and selected the initial set of PDZ-domains interactions to aim to disrupt, CFTR:CALP and PTEN:MAST2, with *de novo* D-peptide inhibitors. After confirming both theoretically and empirically the equivariance of OSPREY's energy function to geometric transformations like reflection, translation, and rotation, I extended Jeff Martin's recent conformation space work[124] and added D-amino acid molecular preparation and design capabilities to OSPREY. In autumn 2022 we were fortunate to be able to recruit chemistry PhD student (and Donald Lab member) Henry to the project to help with running the DexDesign predictions.

Over the year that we worked on DexDesign, we refined and improved the algorithm (Section 3.2.1), formulated new and generally useful computational protein and peptides design techniques (Section 3.2.3.1), an developed a number of different methods to structurally and computationally assess the DexDesign-generated D-peptide inhibitors (Sections 3.3.1 and 3.4.1). Notably, by the end of this highly iterative process, we had used DexDesign to generate a set of 30 *de novo* D-peptides inhibitors, each predicted to bind its PDZ domain target (CALP or MAST2) tighter than the target's endogenous ligand (CFTR or PTEN, respectively)—an important prerequisite of an effective therapeutic.

3.1 Background and Introduction

Since the 1921 discovery[125] of the peptide hormone insulin to treat diabetes, many peptides and peptide-derived therapeutics have come into clinical use, with more than 30 achieving final regulatory approval just since the year 2000[126]. The use of peptides as therapeutics has a number of advantages, including standard protocols for synthesis, good efficacy, high potency, and selectivity[51,52]. On the other hand, peptide therapeutics have a number of drawbacks, including poor stability, oral bioavailibility, membrane permeability, and retention[51]. The substitution of D- for L-amino acids in peptides is one strategy medicinal chemists have used to address these shortcomings.

In Section 3.1.1, we describe the benefits of incorporating D-amino acids into therapeutic peptides. Section 3.1.2 provides background on PDZ domains in general and two PDZ domains in particular that researchers have investigated targeting with L-peptides for biomedical purposes. Section 3.1.3 describes previous computational protein redesign software and algorithms for designing proteins and peptides incorporating non-canonical and D-amino acids. Section 3.1.4 concludes with a summary of DexDesign, a new algorithm we developed and incorporated into the protein design software OSPREY.

With the necessary background covered, the rest of this chapter focuses on an application of the DexDesign algorithm to generate *de novo* D-peptide inhibitors of two biomedically important PDZ domains targets: CAL and MAST2. We then evaluate each computationally generated peptide using multiple structural criteria, including predicted binding affinity and whether a D-peptide mimics binding interactions previously shown to be important to L-peptide binding to PDZ domains.

3.1.1 Benefits of including D-amino acids in peptides

The inclusion of D-amino acids can increase peptide stability by decreasing the substrate recognition by proteolytic enzymes[53]. For example, Chen et al. improved both stability and binding affinity of a bicyclic peptide inhibitor of the cancer-related protease urokinase-type plasminogen activator by substituting a single D-serine for a glycine[54,55]. Haugaard-Kedström et al. observed that the simple substitution of D-amino acids in two positions of their *de novo* PDZ domain inhibitor greatly improved metabolic stability by increasing their peptide's half-life 24-fold[56]. More ambitious uses of D-amino acids have also been performed. Liu et al. constructed an entirely D-peptide inhibitor of the MDM2 oncoprotein using mirror image phage display, an experimental technique used to discover D-peptide drug candidates, that inhibited growth of glioblastoma both in cell culture and nude mouse xenograph models[127]. Nevertheless, the challenge of preparing an enantiomeric protein target for mirror image phage display remains a drug-discovery bottleneck[128].

3.1.2 PDZ domains

With over 270 unique occurrences in more than 150 human proteins, PDZ domains constitute the largest family of peptide-recognition domains in the human genome[129]. A typical PDZ domain has 80-100 amino acids folded into five core β-strands (β1-β5) and two α-helices (α1 and α2)[129–131]. They facilitate a variety of cellular functions, such as modulating polarization, signaling, and trafficking pathways, through interaction with short linear motifs (SLiMs) located at the C-terminus of their ligands[129,132–134]. Usually SLiMs bind into a groove of the PDZ domain between α2 and β2, extending the β2/β3 sheet[132]. Modulating the interaction between a SLiM and its PDZ binding partner is a strategy that both viruses and therapeutics aim to exploit[129,135] and has been explored by previous computational design techniques[23,136–143].

3.1.2.1 CFTR-associated ligand

Cystic fibrosis can cause serious pulmonary and respiratory problems in the lungs by causing the development of a thick mucus that promotes bacterial infection and inflammation. It is caused by many mutations in the cystic fibrosis transmembrane conductance regulator (CFTR), such as ΔF508, which causes destabilized, misfolded CFTR[23,144]. The CFTR-associated ligand (CAL) binds CFTR via CAL's PDZ domain (CALP), which shepherds CFTR through rapid degradation via a lysosomal pathway[23]. A number of research groups have developed peptide stabilizers that bind to CALP, preventing CFTR lysosomal trafficking and degradation. Our lab used computational peptide design to develop a hexamer that bound 170-fold more tightly to CALP than

CALP bound the CFTR C-terminus, rescuing CFTR activity in monolayers of polarized human upper airway epithelial cells that contain the ΔF508 deletion in CFTR—80% of cystic fibrosis patients are homozygous for this mutation[22,23]. Cyclic peptides targeting CALP have also been developed—Dougherty et al. developed a highly selective stabilizing cyclic peptide that binds CALP with a K_D of 6 nM[144]. Competitive peptide inhibitors have also been developed with the goal of developing new methods of managing neurological disease.

3.1.2.2 MAST2

During viral infection, the rabies virus exploits SLiM/PDZ-domain interactions to further its propogation[145,146]. In a neuron, phosphatase and tensin homolog deleted on chromosome 10 (PTEN)'s SLiM interacts with the PDZ domain of microtubule-associated serine-threonine kinase 2 (MAST2) to regulate pathways inhibiting neuronal survival, regrowth, and regeneration[147,148]. The rabies virus glycoprotein's C-terminal residues interact with MAST2's PDZ domain, disrupting the ability of MAST2 and PTEN to form a complex and inhibit neurite outgrowth and apoptosis[145,148,149]. Recognizing the therapeutic potential of promoting neurite outgrowth in the treatment of neurodegenerative disease, Khan et al. developed three peptides that mimic and improve upon the rabies virus glycoproteins's interaction with MAST2's PDZ domain, stimulating neurite outgrowth in proportion to the affinity the peptide bound MAST2[148].

In Section 3.3, we present 30 *de novo* D-peptide inhibitors targeting CALP and MAST2.

3.1.3 Computational tools and algorithms for designing D-peptides

OSPREY is a free and open-source software program containing a suite of computational protein design algorithms developed in our lab[35]. OSPREY has been used to, among other things, predict resistance mutations ablating efficacy of antibiotics used to treat methicillin-resistant *Staphylococcus Aureus*[29,72] and small molecule inhibitors used to treat melanoma, lung, stomach and colorectal cancers[28,30], design and structurally characterize peptide inhibitors of CALP for treating CFTR[22,23], and improve broadly neutralizing antibodies against HIV-1[34,150]. OSPREY has been used to computationally redesign proteins with canonical and non-canonical amino acids[21,23,24,151], as well as optimize protein:small molecule interactions[28,30,31], but as-of-yet has not had the capability to design D-peptides. Given the promising biomedical potential of D-peptides[127,152–156], having the ability to apply OSPREY's ensemble-based, provable protein design algorithms in pursuit of D-peptide design could greatly decrease the required quantity of expensive, time-intensive experiments.

There are a few previous computational techniques for D-peptide design (also reviewed in Donald 2011, Chapter 9)[46]. One of the earliest was by Elkin et al., which used the Multiple Copy Simultaneous Search[157] method to predict candidate D-peptide inhibitors of hepatitis delta antigen dimerization[158]. Recent versions of Rosetta have included functionality to incorporate non-canonical and D-amino acids[159,160]. Philip Kim's group has developed a computational D-peptide design technique based on creating a mirror image of the PDB, identifying hotspot interactions, and searching the D-PDB for similar configurations of hotspot residues[161,162]. They applied this technique to develop two D-peptide inhibitors to the SARS-CoV-2 spike protein receptor binding domain and the human angiotensin-converting enzyme 2 (ACE2) that mimic the ACE2 α1-binding helix[163,164]. Overall, the number of algorithms the protein designer has available for D-peptide design is notably sparser than for L-design, and the development of additional computational protocols for this important task is warranted.

3.1.4 DexDesign

In this chapter we present a new computational protocol, DexDesign, for designing *de novo* D-peptides in OSPREY. DexDesign constructs D-peptide scaffolds by mirroring the structure of a L-protein:peptide complex into D-space, then uses the geometric search algorithms in MASTER[57] to search hundreds of thousands of L-protein structures for substructures with backbones similar to the D-peptide. It then uses the iMinDEE/K* algorithm[17,47] in OSPREY to redesign a scaffold D-peptide's sidechains to optimize target binding (see Figure 6). Given the biomedical importance of modulating CALP and MAST2 PDZ domain interactions, coupled with the advantages of D-peptide therapeutics, we use DexDesign to predict D-peptides inhibitors of these two protein targets.

In summary, this chapter makes the following contributions:

1. A new computational protocol, DexDesign, for designing *de novo* D-peptide binders,
2. Three novel design techniques leveraging continuous flexibility and the computation of ε-accurate ratios of partition functions over molecular ensembles: the Minimum Flexible Set, Inverse Alanine Scan, and K*-based Mutational Scan,

3. Application of DexDesign to predict D-peptide binders to the PDZ domains of CALP and MAST2,
4. Multi-criterion computational validation and structural analyses of the DexDesign-generated peptides,
5. OSPREY-generated structural ensembles of the D-peptide:PDZ domain complexes, and
6. An open source implementation of DexDesign in the computational protein redesign software OSPREY.

3.2 Methods

3.2.1 Algorithm and Computational Protocol

DexDesign generates *de novo* D-peptides by combining MASTER's molecular structure search[57] with provable computational protein redesign algorithms in OSPREY[35,46], mediated via energy-equivariant geometric transformations (EEGT). EEGTs, such as translation, rotation, or reflection, are geometric transformations of a molecular structure that do not affect the energy of that structure. Each EEGT corresponds to a symmetry in the energy field[165]. For example, an energy function will compute the same energy of protein structure *s* and *s* reflected over the Cartesian *x*-*y* plane. The MASTER algorithm searches a database of protein structures for a user-specified query structure and is guaranteed to find all protein substructures in the database with a backbone RMSD below a cutoff threshold[57]. The K* algorithm in the OSPREY software suite[35] searches for amino acid substitutions that maximize a design objective, such as binding affinity or specificity[17,21,37,38]. It does this by exploiting molecular ensembles to compute a provably accurate ε-approximation to the binding constant, K_a[17,21] (see Section 1.2.2 on the K* algorithm). In essence, DexDesign invokes MASTER search as a subroutine to suggest D-peptide scaffolds with backbone conformations similar to their L-peptide counterpart, then invokes K* as a subroutine to optimize amino acid sequences and side chain conformations on those scaffolds.

After preparing a database (DB) of L-protein structures, a protein designer initiates DexDesign by identifying a protein target *(t)* of interest, for which there exists a structure of a protein or peptide *(p)* bound to *t*. In MASTER terminology, the structure of this bound complex will become our query, q_{tp}. Below, we define terms used in the DexDesign algorithm:

1. Let s_n be a protein structure with *n* residues. We define substructure $s_{i,j}$ of s, where $1 \leq i < j \leq n$, to be a structure of residues *i* through *j* of *s*.
2. Let $r(s, a)$ be a function that reflects all atoms in protein structure s across a plane a. Without loss of generality, we let a be the *x*-*y* plane and define $r(s) = r(s, a)$ henceforth. We note that r is an involution, i.e., $r\big(r(s)\big) = s$.
3. Let $M(DB, s, c)$ be the MASTER subroutine. M returns a set of substructures from DB with backbone RMSD, when optimally aligned with protein substructure *s*, less than *c* Å.
4. Let $O(p, t)$ be the OSPREY K* subroutine. O redesigns peptide *p* towards increased binding affinity with protein target *t* by searching over mutated and continuously minimized amino acid sidechains, and returns a set of mutant sequences (and structural molecular ensembles) derived from *p* that have improved binding with *t*.

The DexDesign algorithm is described in Figure 6 and Figure 7.

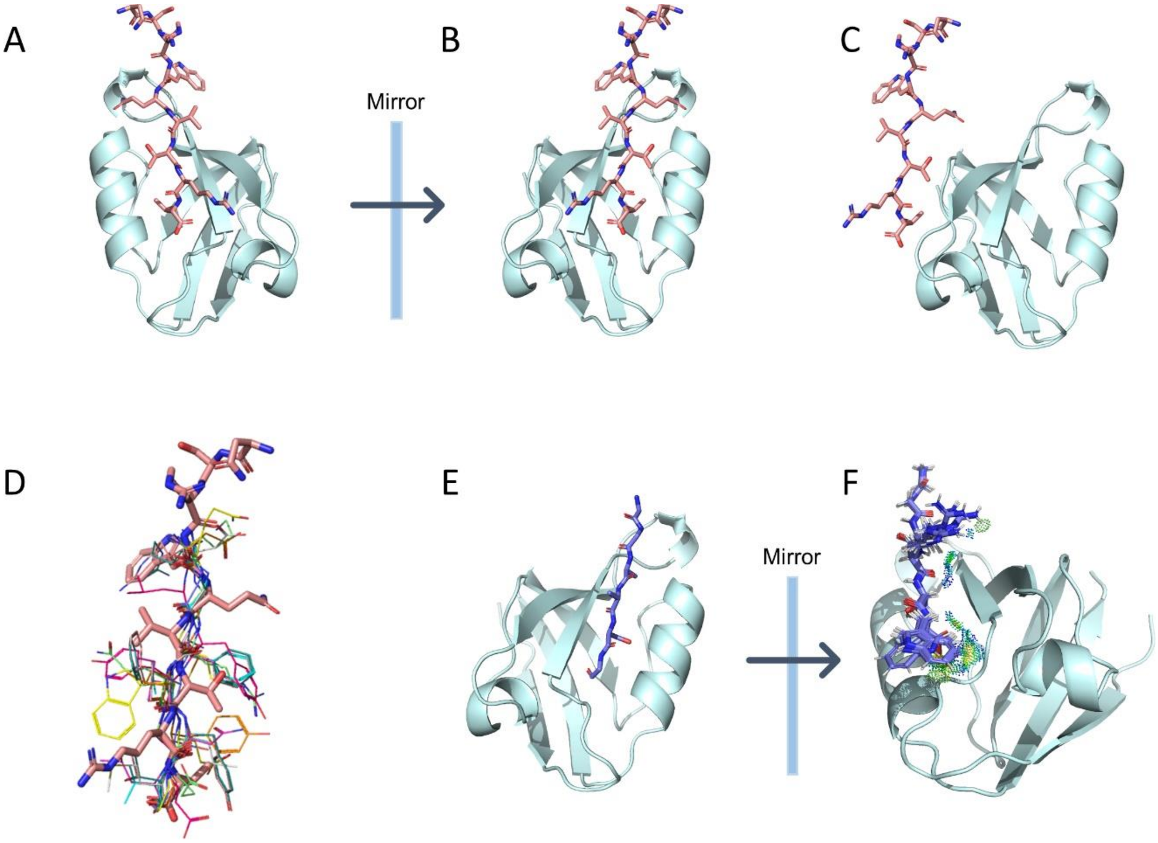
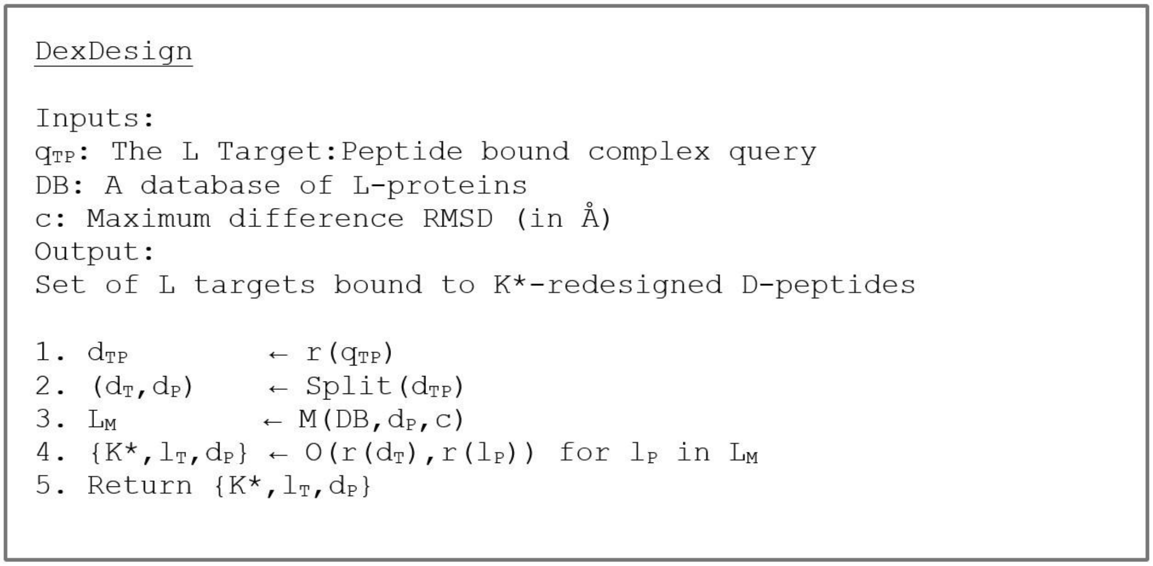


Figure 6: Example of the DexDesign protocol applied to CALP. (A) The potent inhibitor kCAL01 (in pink) bound to CALP (in cyan) (PDB ID 6ov7)[22] is used as a starting point for a DexDesign search for a D-peptide inhibitor of CALP. Both kCAL01 and CALP are composed of solely L-amino acids. **(B)** The input structure is reflected to produce a mirror-image of the kCAL01:CALP complex, which flips the chirality of all amino acids to D. **(C)** The complex is split into its constituent peptide and protein components. **(D)** Residues P^0 to P^{-5}, which are the residues located within CALP's binding pocket, are used as the query structure to conduct a MASTER[57] search of a large database of L-protein structure to find substructures with similar backbones (as determined by backbone RMSD) to the D-version of kCAL01. 10 representative matches of L-peptide segments (in multicolored wire representation) are overlaid on the pink stick D-version of kCAL01. **(E)** Each L-peptide match is aligned to the D-version of kCAL01 in the D-kCAL01:CALP complex structure and D-kCAL01 is removed. Shown (in purple sticks) is an L-peptide substructure GGAASG (residues 168-173) that MASTER identified in Mycobacterium tuberculosis Rv0098 (PDB ID 2pfc)[166]. This L-peptide forms the basis for OSPREY redesign. **(F)** The L-peptide:D-CALP complex is reflected once again to form a D-peptide:L-CALP complex. The K* algorithm[21,47] in OSPREY[35] is then invoked to conduct a search over D-peptide sequences and continuous sidechain conformations to optimize the D-peptide for binding. K* identified two mutations at positions P^0 and P^{-2} predicted to improve binding of the peptide with a normalized ΔΔG of -1.4 kcal/mol, improving K_D by 9-fold (see Appendix B.1 for information on the normalization procedure). Position P^0 is mutated from Gly to Trp, and P^{-2} is mutated from Ala to Arg. Shown is an OSPREY-predicted low energy ensemble of the D-peptide GGARSW with MolProbity probe dots[167–169] showing goodness-of-fit the OSPREY-predicted mutated D-sidechains make with CALP.



```
DexDesign

Inputs:
$q_{TP}$: The L Target:Peptide bound complex query
DB: A database of L-proteins
c: Maximum difference RMSD (in Å)
Output:
Set of L targets bound to K*-redesigned D-peptides

1. $d_{TP}$          ← $r(q_{TP})$
2. $(d_T, d_P)$      ← Split$(d_{TP})$
3. $L_M$             ← $M(DB, d_P, c)$
4. $\{K^*, l_T, d_P\}$ ← $O(r(d_T), r(l_P))$ for $l_P$ in $L_M$
5. Return $\{K^*, l_T, d_P\}$
```

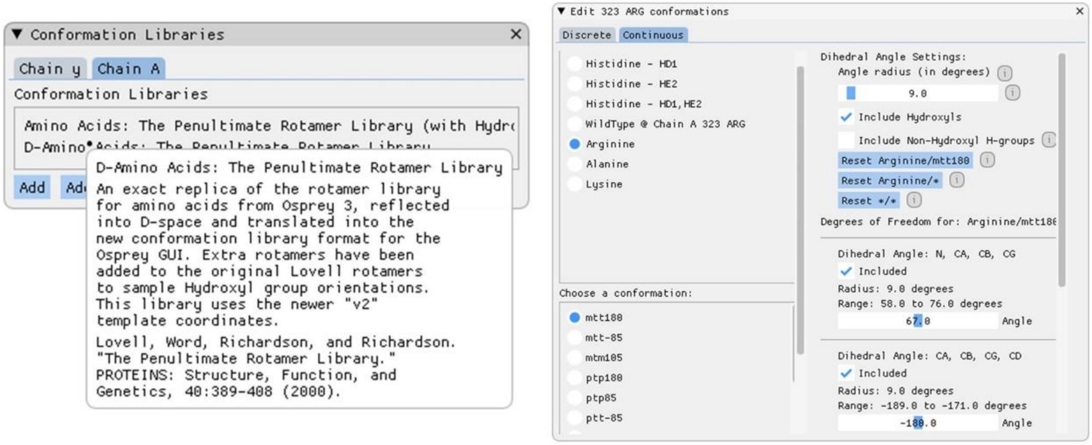
Figure 7: The DexDesign Algorithm. It takes as input a query structure (q) of a bound target (t) and peptide (p), a database (DB) of L-protein structures, and a cutoff (c). Line 1 reflects the L-protein complex into D-space. Line 2 splits the target and peptide into two structures, the D-target (d_T) and the D-peptide (d_P). Line 3 calls MASTER[57] (M) to search the L-protein database for all substructures with a backbone RMSD to d_P less than c. The set of results is saved in L_M. Line 4 reflects each MASTER peptide (l_P) and D-target (d_T), to make a D-version of l_P bound to the original L-target. OSPREY[35] K* redesign[21,47] is then run on each target:peptide complex (l_T + d_P), resulting in a set of K* scores (K*), along with an OSPREY-predicted structural ensemble of the D-peptide (d_P) and L-target (l_T) complexes, with the sequence and continuously minimized sidechains of d_P optimized to bind l_T. The K* scores and computed structural ensembles are returned on Line 5.

3.2.2 New features in OSPREY

3.2.2.1 New feature: customize existing or add new conformation libraries

In previous works[25,26,28,30,170], a typical OSPREY-based computational protein redesign entailed 1) selecting a starting molecular structure, 2) adding hydrogens, 3) specifying the design algorithm and its input parameters, 4) running the algorithm, and 5) analyzing the results. To enable Step 4, OSPREY included a default library of amino acid atom connectivity templates from Amber[171] and rotamers from Lovell et al[49]. These templates and rotamers then became starting points for continuous minimization within a voxel during the sequence and computational search[17,172]. Embedding the templates within the algorithm provided protein designers with simple defaults for the majority of protein redesign problems. And while some works[19,150,151] have expanded these defaults in certain cases, such as in our use of non-canonical amino acids to design CALP inhibitors[151], the necessity of providing a simple, general approach that enabled protein designers to experiment with diverse and novel biochemical building blocks remained. The implementation of a general, in contrast to application-specific, approach to modeling templates and flexibility enables designers to design proteins with chemistries that the creators of the protein design software didn't even anticipate!

To meet this need, we have simplified the process of specifying rotamers, voxel-based continuous minimization, new molecular fragment templates, or even entire conformation libraries in OSPREY. OSPREY continues to provide intelligent defaults, but they are moved from deep within the software and are now exposed to the designer, allowing the designer to modify them as needed in a simple graphical user interface (see Figure 8, right). This seemingly simple change has profound implications for OSPREY. When the complete conformation space specification (i.e., the design parameters such as the mutable residues, the flexible residues, etc. See Section 3.2.3.2 for further definition) is a user-modifiable input to the algorithm, new classes of design capabilities, such as design with D-amino acids via DexDesign, are unlocked.



Conformation Libraries

Amino Acids: The Penultimate Rotamer Library (with Hydr

D-Amino Acids: The Penultimate Rotamer Library

D-Amino Acids: The Penultimate Rotamer Library

An exact replica of the rotamer library for amino acids from Osprey 3, reflected into D-space and translated into the new conformation library format for the Osprey GUI. Extra rotamers have been added to the original Lovell rotamers to sample Hydroxyl group orientations. This library uses the newer "v2" template coordinates.

Lovell, Word, Richardson, and Richardson. "The Penultimate Rotamer Library." PROTEINS: Structure, Function, and Genetics, 40:389-408 (2000).

○ Histidine - HD1
○ Histidine - HE2
○ Histidine - HD1,HE2
○ WildType @ Chain A 323 ARG
● Arginine
○ Alanine
○ Lysine

Choose a conformation:

● mtt180
○ mtt-85
○ mtm105
○ ptp180
○ ptp85
○ ptt-85

Dihedral Angle Settings:
Angle radius (in degrees) ⓘ
9.0 ⓘ
☑ Include Hydroxyls
□ Include Non-Hydroxyl H-groups ⓘ
Reset Arginine/mtt180 ⓘ
Reset Arginine/* ⓘ
Reset */* ⓘ
Degrees of Freedom for: Arginine/mtt18

Dihedral Angle: N, CA, CB, CG
☑ Included
Radius: 9.0 degrees
Range: 58.0 to 76.0 degrees
67.0 Angle

Dihedral Angle: CA, CB, CG, CD
☑ Included
Radius: 9.0 degrees
Range: -189.0 to -171.0 degrees
-180.0 Angle

Figure 8: Screenshots of new OSPREY protein design specification options. OSPREY[35] now allows protein designers to add their own conformation libraries and easily control rotamer selections and allowed movements. Left: DexDesign includes, in addition to the standard L-conformation library, a D-conformation library that is the mirror image of the L-library. A conformation library describes the standard connectivity templates, rotamers, and allowed movements, all of which can be further customized by the protein designer. A protein designer can specify multiple, distinct conformation libraries per chain. Right: New detailed control over side chain conformational flexibility. Each of the conformation library rotamers (e.g., tptm, pttm, etc.) can be included or excluded. The angle of voxel in which OSPREY continuously minimizes a rotamer[47] can now be set, and each dihedral angle can be included or excluded from the continuous minimization. OSPREY provides the protein designer with complete control of the definition of the conformation space. This control enables the designer to explore new types of conformation spaces, such as the D-peptide space.

3.2.2.2 New feature: D-protein/peptide design

The molecular interaction forces between two molecules are invariant over a reflection of those two molecules. Put another way, if $K_D(x, y)$ is the dissociation constant for protein *x* binding protein *y*, then $K_D(x, y) = K_D\big(r(x), r(y)\big)$. The OSPREY energy function, as described in detail in previous works[35,47,124,173], mimics this physics precisely be being (exactly) energy equivariant with respect to reflection, allowing us to add the ability to design D-proteins and peptides in OSPREY. We accomplished this by reflecting OSPREY's default L-conformation library into D-space. A protein designer can now use the functionality described in Section 3.2.2.1 to specify a D- or L-conformation library on a per-protein basis (see Figure 8, left). DexDesign requires this capability because designing a D-peptide targeting an L-protein requires the use of both D- and L-conformation libraries.

3.2.3 Applying DexDesign to CALP and MAST2

To use DexDesign to predict *de novo* D-peptide inhibitors to CALP and MAST2, we started with structures of their bound complexes: kCAL01 bound to CALP (PDB ID 6ov7)[22] and PTEN bound to MAST2 (PDB ID 2kyl)[174]. We created a database of high-resolution L-protein structures by mining the RCSB PDB[175] for crystallographically determined structures with a resolution better than 2.5 Å, omitting DNA, RNA, and small molecules. This resulted in a database containing 119,160 structures (see Figure 16 on page 151 for further description of the composition of the database). Using the DexDesign algorithm in Figure 7, we first reflected each molecular structure to D-space and split the peptide and target PDZ domain into two separate structures, d_p and d_t, respectively. We then used the MASTER algorithm[57] to query the database for L-protein substructures with backbones similar to d_p. MASTER returns a set of candidate L-peptides (L_M), each of which ($l_p \in L_M$) we superimposed over d_p in the d_p:d_t complex and subsequently removed d_p. We then again reflect each bound complex l_p:d_t, resulting in a *de novo* D-peptide candidate $r(l_p)$ bound to the original L-protein target $r(d_t)$ in a complex $r(l_p)$:$r(d_t)$.

Prior to executing Step 4 of the DexDesign algorithm (K* redesign; see Figure 7) we further pruned the set of D-peptide candidates based on two additional criteria. First, we visualized candidate D-peptide:L-protein complex structures in PyMol[176] using Molprobity dots[167,168] in our lab's Protein Design Plugin[169] to evaluate the number and severity of steric clashes, as steric clashes need to be resolved via sequence mutation and additional modeling of continuous side chain flexibility in the K* algorithm. Since clash resolution and peptide improvement via K* redesign utilize the same algorithmic technique and therefore draw from the same pool of limited computational resources, we chose to prioritize D-peptide candidates with fewer clashes so we could allocate more computational resources to improving a D-peptide candidate via K* sequence redesign. Second, we observed instances where MASTER found identical D-peptide candidate sequences with nearly identical structures in multiple distinct PDB files, which we resolved by removing the duplicate results.

Using the above criteria, we selected 8 promising D-peptide candidates to use as starting points for OSPREY K* redesign. We call these selected candidates *D-peptide redesign (DPR) scaffolds*. The complete set of DPR scaffolds is described in Table 11 on page 147. To evaluate and improve upon the DPR scaffolds, we developed three new design techniques.

3.2.3.1 New Design Techniques: Minimum Flexible Set, Inverse Alanine Scanning, and Mutational Scanning

DexDesign's *de novo* peptide design has one important distinction from protein redesign: the model of the starting protein structure used as input for K* redesign is a theoretical model, rather than one determined by experiment. As described in Section 3.1.3, OSPREY's algorithms have been successfully applied to a large and diverse set of biomedical applications. Yet in the most common uses of the K*-family of algorithms, *viz.*, those provably approximating K_a (K*[17,21], BBK*[38], MARK*[37], and EWAK*[24]), the protein designer starts a redesign to achieve a specific redesign goal (e.g., improving or ablating binding of a protein to a ligand) from an experimentally determined structure of the protein:ligand complex. The mere existence of this experimental structure provides a solid foundation upon which further redesign builds, i.e., the given protein and ligand bind *at least in vitro*, and *at least to some degree*. Unfortunately, we do not have this luxury when designing *de novo* peptides to bind a given protein target. For example, while DexDesign uses the backbones of known L-peptide binders as input to the algorithm, the resulting DPR scaffolds are sufficiently different in sequence, side chain conformation, and chirality that the protein designer should assume that their DPR scaffold will not bind its protein target *in vitro* without further optimization via K*. To address this challenge—which is inherent to *de novo* design—we have developed new design techniques that systematically evaluate the quality of DPR scaffolds and rigorously suggest mutations that are predicted to improve the D-peptide's binding affinity to its target PDZ domain.

The following design techniques assume that the protein designer has a fixed amount of time and computational resources at their disposal. To that end, they are formulated to allow designers to rapidly evaluate their DPR scaffolds by restraining the conformation space to the minimal size necessary to computationally evaluate a hypothesis. Conformation space size grows exponentially with the number of flexible residues. Here, restricting the size of a conformation space is an effective technique to obtaining computational predictions quickly.

3.2.3.2 Design Technique 1: Identifying a Minimum Flexible Set

The K* algorithm in OSPREY predicts a provable approximation to K_a by calculating provable bounds on the partition function values of three molecular ensembles: the protein:ligand complex, the apo protein, and the apo ligand[17,21]. To generate these low-energy ensembles, the K* algorithm enumerates a stream of conformations in order of increasing energy, stopping only when it reaches a point when its enumerated conformations are sufficient to calculate a provably good ε-approximation to the partition function value, and thus the K* score. The set of conformations that K* enumerates is determined entirely by its *conformation space*, or the combinatorial set of all conformations that can be generated from given flexibility rules. Examples of such rules include the number of side chain rotamers an amino acid can explore, and the degree of continuous rotational flexibility permitted for a dihedral angle. The conformation space is in turn specified by the protein designer as an input to the K* algorithm (see Figure 8). While specifying an appropriate conformation space has always been an important factor in K*'s ability to find mutations that accomplish a protein design's goal, specifying a sufficiently efficient, but still expressive, conformation space is an essential prerequisite of DPR scaffold redesign.

The MASTER search in Step 3 of the DexDesign algorithm (see Figure 7) returns a set of L-peptides with low backbone RMSD to the D-peptide query. Due to the fact the MASTER search is by backbone-only RMSD, it is often the case that the L-peptide search results have side chains in sterically unfavorable positions that clash with the D-protein target. As discussed in Section 3.2.3, we prune DPR candidate peptides that cause many unfavorable clashes. On the other hand, we keep DPR candidate peptides with only a few small clashes because these clashes can typically be resolved with an appropriately specified conformation space. We call the set of residues that the protein designer must specify to be continuously flexible to resolve these clashes the *Minimum Flexible Set*.

The Minimum Flexible Set is different for each DPR scaffold, since the D-peptide in each DPR scaffold is unique. Given a fixed budget of computational resources, protein designers should prefer DPR scaffolds requiring smaller Minimum Flexible Sets, since such DPR scaffolds allow K* to expend more compute resources on searching for favorable mutations that increase binding to the target protein. See Figure 9 (A) for an example of specifying the Minimum Flexible Set for a CALP-DPR candidate.

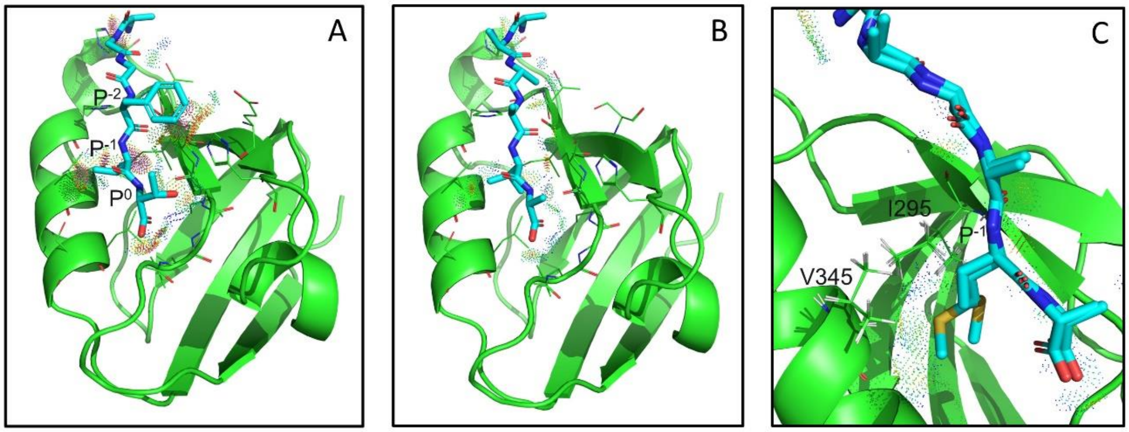


Figure 9: Illustration of the Minimum Flexible Set and Inverse Alanine Scanning design techniques applied to CALP-DPR5. (A) Choosing the Minimum Flexible Set. The CALP-DPR5 scaffold peptide (in cyan) is extracted from a crystal structure of the tobacco necrosis virus (PDB ID 1c8n, residues C61-66, AGGFVT)[177]. When aligned and superimposed over the query peptide kCAL01 and CALP[22] (in green) to create the DPR scaffold, sidechain and backbone clashes are present that the designer must address (red and pink MolProbity dots)[167–169]. The four peptide residues that clash with CALP are located at P^0 (Thr), P^{-1} (Val), P^{-2} (Phe), and P^{-5} (Ala). We specify the *Minimum Flexible Set*, or those residues that must be allowed to undergo continuous minimization (see Gainza et al, 2012)[47] in all designs derived from CALP-DPR5, as the peptide residues located at P^0, P^{-1}, and P^{-2}. The peptide:CALP clash involving the alanine located at P^{-5} can be resolved by allowing OSPREY to translate and rotate the peptide during K* optimization, an option now available to the protein designer in the process of specifying a redesign's conformation space (related to Figure 8). **(B** and **C)** The Inverse Alanine Scanning applied to CALP-DPR5. Complementing the Minimum Flexible Set technique, *Inverse Alanine Scanning* addresses peptide:target clashes by mutating all peptide residues modulo a single amino acid to alanine. In **(B** and **C)**, we focus on position P^{-1} (Val) and use K* to mutate all other peptide residues to alanine, as well as to continuously minimize peptide:target sidechain conformations. **(B)** Inverse Alanine Scanning structural prediction with the source amino type, valine. As expected, the clashes present in CALP-DPR5 **(A)** vanish in the Inverse Alanine Scanning peptide (as indicated by the lack of red and pink MolProbity dots). Furthermore, K* has rotated P^{-1} (Val) to point towards the peptide's C-term, indicating that conformation is preferable. With this rotation, P^{-1} (Val) remains within CALP's hydrophobic pocket. **(C)** An OSPREY-predicted ensemble of the result of the Inverse Alanine Scan mutating position P^{-1} to methionine. CALP V345 and I295 form favorable van der Waals interactions (green and blue MolProbity dots) with multiple conformations of P^{-1} (Met), which is also reflected in the increase in K* score of P^{-1} (Met) compared to P^{-1} (Val). This result indicates that further K* binding affinity optimization can include methionine at P^{-1}, and that V345 and I295 should be allowed to flex continuously in CALP-DPR5 design candidates containing mutations at P^{-1}.

3.2.3.3 Design Technique 2: Inverse Alanine Scanning

Inverse Alanine Scanning is a technique complementary to the Minimum Flexible Set. Whereas the Minimum Flexible Set technique identifies a conformation space that resolves all clashes, Inverse Alanine Scanning allows the designer to investigate single residue mutations on the peptide that may increase binding to the target protein only when the target protein's nearby residues are provided sufficient flexibility in the conformation space. In contrast with Minimal Flexible Set, this technique resolves the problem of clashes between DPR scaffolds and the protein target by mutating all peptide residues, modulo the single residue under investigation, to alanine. We call this *Inverse* Alanine Scanning because this computational technique mutates all the peptide's residues *except* the residue of interest to alanine, the opposite of the canonical alanine scanning experiment. See Figure 9 (B & C) for a picture of this technique.

Notably, Inverse Alanine Scanning not only provides evidence as to which residues in the target protein must be flexible to accommodate certain favorable mutations, but it also provides evidence about which protein residues can safely remain rigid because they do not interact with peptide mutations in their vicinity. In the former case, a protein designer can visually inspect the OSPREY-generated structural ensemble of a favorable mutant to determine the set of residues which flexed and interacted with the mutated sidechains: these residues must remain flexible in the final conformation space. Conversely, when Inverse Alanine Scanning identifies a favorable mutant that, upon visual inspection of the OSPREY-generated structural ensembles, has residues that *do not* flex, then these residues can safely be omitted from the final conformation space. This knowledge is valuable because it allows the designer to specify a smaller conformational space than they otherwise would in the next technique, K*-based Mutational Scanning.

3.2.3.4 Design Technique 3: K*-based Mutational Scanning

After learning which residues must flex from Minimum Flexible Set and obtaining hints as to which mutations might improve binding from Inverse Alanine Scanning, the K*-based Mutational Scanning technique (hereafter *Mutational Scan*) can be used. A *Mutational Scan* uses K* to systematically mutate a residue in the DPR scaffold to all 20 amino acids. We specified a K* design implementing a Mutational Scan for each residue in a DPR peptide and set the conformation space as the union of the set of flexible residues from the Minimum Flexible Set and the Inverse Alanine Scanning steps.

We ran K* Mutational Scans on each of the DPR scaffolds. In many cases we observed mutations that notably improved the K* score. We then used the results of the Mutational Scans to inform the specification of additional K* designs that permitted multiple simultaneous peptide mutations in order to optimize peptide:target binding. We then further refined that set by removing sequences whose increase in K* score was driven primarily by peptide destabilization, as indicated by a large decrease in the unbound peptide's partition function value q_L (see Section 1.2.2 on the K* algorithm). We then sorted the remaining favorable DPR sequences by their K* score. Finally, we analyzed the OSPREY-predicted structures of the top 3 sequences for each DPR scaffold. Our analysis is included below.

3.3 Results

3.3.1 DPR validation criteria

The aim of DexDesign is to predict novel D-peptides inhibitors. For this reason, we validated each of the DPR peptides across multiple criteria relevant to PDZ domain inhibitors. These criteria include:

1. **The DPR's binding affinity**. As an effective inhibitor must interact with its protein target in such a way that it disrupts the target protein's ability to bind its endogenous ligand, to validate the inhibitory potential of the D-peptides we

compared their K* scores to the K* scores of each PDZ-domain's endogenous ligand, as well as some previous L-peptide inhibitors. We use the K* algorithm[17,21] to optimize the D-peptide's sequence to increase binding affinity between the peptide and the endogenous ligand's binding site (the groove between α2 and β2 in the PDZ domain). See Section 1.2.2 for a definition of K* scores and how they are generated by the K* algorithm.

2. **The DPR's ability to replicate biophysical facets common to PDZ domain binding.** Due to their central role in regulating cellular trafficking and signaling pathways[129], much research has been conducted to better understand and characterize PDZ domains[129–134,145,178–180]. This research has identified structural and biophysical elements that are commonly found facilitating canonical PDZ domain interactions[134]. One such element is the presence of a hydrogen bond network formed between the peptide's C-terminal carboxylate and the loop connecting β1 and β2, termed the *carboxylate binding loop (CBL)*[130,134,181]. Another is the presence of β-strand-β-strand interactions between the peptide and β2[133,134]. A third commonality is the presence of a hydrophobic pocket in the α2-β2 groove, which canonically is filled by a hydrophobic residue at position P^0 in the peptide[133], and in some peptides, by the P^{-2} residue[134].

We assessed the following biophysical facets in our validation of D-peptides: 1) the H-bond network formed by the D-peptide carboxylate and the CBL; 2) β-strand interactions between the D-peptide backbone and β2, and; 3) the ability of the D-peptide to fill the hydrophobic pocket. We believe these facets to be sufficient, but not necessary, in the development of novel D-peptide inhibitors (see Section 3.4.1 for additional information on this point).

3. **The presence of novel and favorable interactions in the DPR.** Our validation includes a structural analysis of the OSPREY-predicted low-energy ensemble of the DPR bound to its target PDZ domain. Since DexDesign predicts *de novo* D-peptides, and since empirical structures of D-peptide inhibitors bound to PDZ domains are lacking, it is possible that a D-peptide could bind its target PDZ domain in a mode quite distinct from that of canonical L-peptides. For example, L-peptide residues at positions that point into the PDZ domain's binding groove may, in a D-peptide, point away (and vice-versa, see Figure 17 on page 152 for an example), providing the possibility for some peptide residues to interact with parts of the PDZ domain in ways not formerly possible. To account for the possibility of novel modes of binding (and not, e.g., disregard D-peptides that fail to replicate all the criteria listed in 2), we analyzed the OSPREY-predicted low-energy molecular ensembles of the DPR bound to its target PDZ domain. In these analyses, we highlight the presence (or absence) of notable structural features capable of further validating the quality of the DPRs.

3.3.2 Designed inhibitors targeting CALP

Using our crystal structure of kCAL01 bound to CALP (PDB ID 6ov7)[22], we used DexDesign to generate 5 DPR scaffolds: CALP-DPR[1-5] (see Table 11 on page 147 for further information about the DPR scaffolds). We then applied the design techniques and selection procedures from Section 3.2.3.1 to optimize the DPRs, thereby obtaining a final set of 15 D-peptide CALP inhibitors, CALP-PEP[1-15]. We assessed each of the CALP-

PEPs using the quantitative and structural validation criteria described in Section 3.3.1. We also compared the CALP-PEPs to CALP's endogenous ligand (CFTR) and also to the most binding-efficient L-peptide CALP inhibitor, kCAL01, which our group reported in 2012[23] and solved a crystal structure of in 2019[22]. An overview of each of the CALP-PEP's K* scores, CBL H-bonds, and peptide:β2 backbone H-bonds is shown in Figure 10. Notably, each of the CALP-PEPs is predicted to bind CALP tighter than the CFTR C-terminal SLiM, a critical prerequisite of an effective inhibitor. After normalization (see Appendix B.1 for information on the normalization procedure) and conversion to Gibbs free energy, the top 3 peptides, CALP-PEP9, CALP-PEP4, and CALP-PEP5, when compared to the CFTR C-terminus, have a ΔG of 2.3, 2.3, and 2.1 kcal/mol lower than the CFTR C-terminus (CALP-PEP9: $\Delta G = -6.9$ kcal/mol, CALP-PEP4: $\Delta G = -6.9$ kcal/mol, CALP-PEP5: $\Delta G = -6.7$ kcal/mol, CFTR C-terminus: $\Delta G = -4.6$ kcal/mol), improving K_D over the CFTR C-terminus by 46-, 44-, and 33-fold, respectively. Below, we provide the results and analyze the OSPREY-predicted structural ensembles.

K* redesign of the peptide sequence enabled each of the CALP-PEPs to achieve a tighter binding affinity to CALP when compared to the DPR scaffold from which it was generated. This is indicated by their positive $\log_{10}$ ΔK* score (see Table 12). Notably, the CALP-PEPs ΔK* scores strongly correlate with their K* scores (Spearman correlation = 0.95). We postulate this strong correlation indicates that DexDesign's K* optimization is not merely alleviating clashes in the DPR scaffolds, but that it is also identifying peptide sequences forming novel side chain interactions that increase binding affinity. For example, CALP-PEP9's P^{-2} arginine reaches across β2 and makes favorable contacts with β3's Glu309 (see Figure 11, D). The magnitude of the predicted improvement in binding ranges from ΔΔG = -2.0 kcal/mol for CALP-PEP15 to -3.5 kcal/mol for CALP-PEP9. Next, we validated the CALP-PEPs based on their ability to replicate canonical PDZ domain binding motifs.

The three canonical PDZ domain binding motifs we used as criteria to further validate the CALP-PEPs were: 1) the presence of an H-bond network between the peptide's C-terminal carboxylate and the CBL; 2) the presence of peptide:β2 backbone interactions; and 3) whether the D-peptide filled the hydrophobic pocket typically filled by P^0 in L-peptides. To quantify (1), we counted the number of H-bonds formed between the peptide's C-terminal carboxylate and the CBL, and to quantify (2) we counted the number of H-bonds between the peptide and β2 backbones. We used visual inspection with MolProbity Probe Dots[167,168] in our lab's Protein Design Plugin[169] to perform a binary classification for (3), more specifically, we classified the hydrophobic pocket as *filled* if a D-peptide's sidechain contacted the residues within the pocket. As a point of reference, kCAL01 and the CFTR C-terminal SliM each have 3 H-bonds with the CBL, 3 H-bonds between the peptide and β2 backbone, and fill CALP's hydrophobic pocket their P^0 amino acid (valine for kCAL01, leucine for CFTR C-terminal SliM).

9 of the 15 CALP-PEPs (CALP-PEP[4-12]) had 2 or more H-bonds between their C-terminal carboxylate with the CBL. CALP-PEPs derived from the CALP-DPR1 and CALP-DPR5 scaffolds had 1 (or in the case of CALP-PEP1 and CALP-PEP3, 0) H-bonds. Encouragingly, all the CALP-PEPs formed at least 2 backbone H-bonds with CALP's β2 strand (see Figure 10 and Table 12). CALP-PEP9, which we predict to be the tightest binder to CALP, forms 3 C-terminal carboxylate H-bonds with the CBL and 3 backbone H-bonds with CALP's β2 strand, matching the numbers formed by both the

CFTR C-terminal SliM and kCAL01 (see Figure 11). CALP-PEP9 contains arginines at position P^{-2} and P^{-5}, both of which are predicted to make favorable van der Waals contacts with CALP (see Figure 11, B&D). In addition, CALP-PEP's P^{-2} guanidino group is predicted to form an H-bond with CALP's S294 and a salt bridge with E309, and P^{-5} H-bonds with CALP's E300 and H301 (Figure 11, B&D). While it does not appear that the quantity of CBL and backbone H-bonds drives the predicted strength of binding of CALP-PEPs (see Figure 10), the CBL does play an important role in determining peptide specificity[129], therefore we regard evaluation of D-peptide:PDZ domain H-bonds as necessary components of a larger ensemble of criteria.

In contrast, whether a CALP-PEP fills CALP's hydrophobic pocket is important to designing a tight binder. All the CALP-PEPs saw an increase in their K* scores when they mutated P^{-1} to an amino acid capable of filling the pocket (see Table 12 on page 148). 11 of the 15 CALP-PEPs mutate P^{-1} to histidine, 3 of the 15 to phenylalanine, and CALP-PEP15 is unique with methionine. For example, the mutation to histidine at P^{-1} in CALP-PEP9 fills and favorably interacts with multiple residues in the hydrophobic binding pocket (see Figure 11, C).

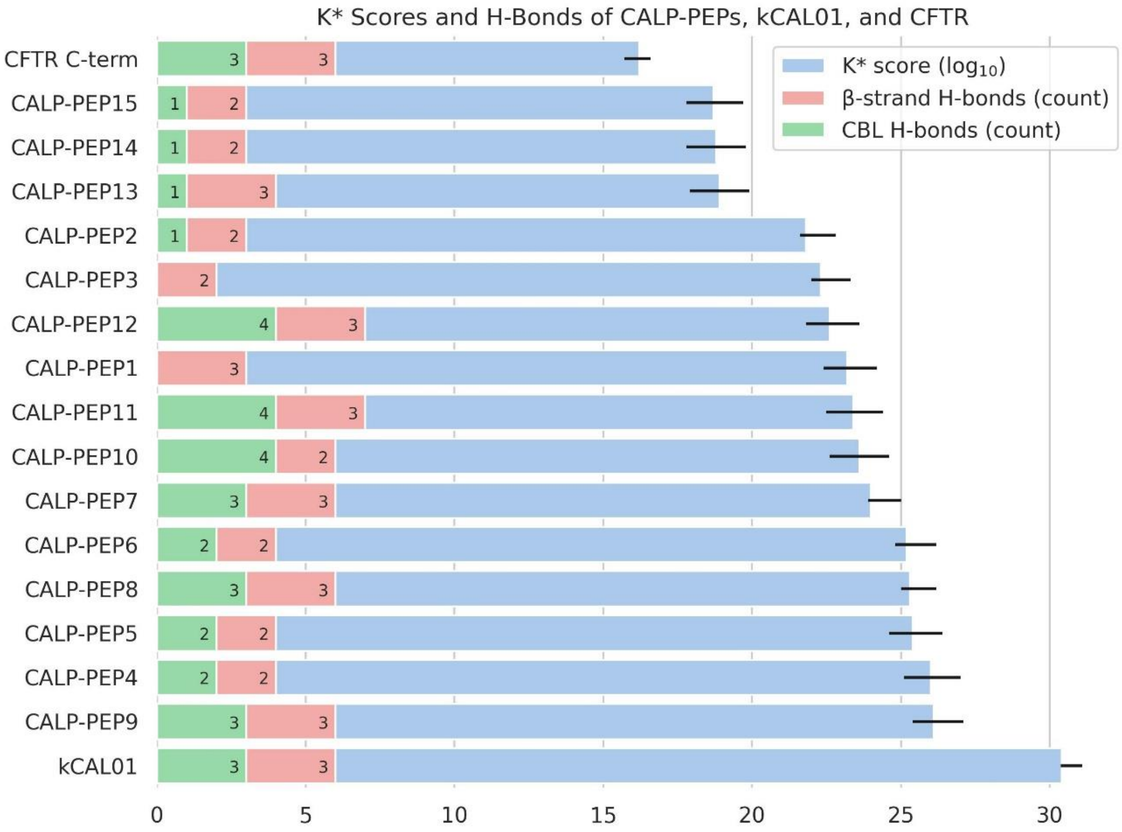


Figure 10: Quantitative and Structural Analysis of the CALP-PEPs, kCAL01, and the CFTR C-terminal SliM. The OSPREY-predicted binding affinity ($\log_{10}$ K* score) of the tightest known peptide binder of CALP (kCAL01, PDB ID 6ov7)[22], the CALP-PEPs, and CALP's endogenous ligand (the CFTR C-terminal SliM, PDB ID 2lob)[182] show (blue bars) that the CALP-PEPs are predicted to bind more tightly, with $\log_{10}$ K* scores ranging from 18.7 (CALP-PEP15) to 26.1 (CALP-PEP9), than the CFTR C-terminal SliM ($\log_{10}$ K* score of 16.2). Conversely, the CALP-PEPs are predicted to bind CALP less tightly than the best known CALP peptide inhibitor, kCAL01[22,23], which OSPREY predicts to have a $\log_{10}$ K* score of 30.4. Since the primary objective of a competitive inhibitor is to outcompete an endogenous ligand in binding to the target protein, the K* scores, *viz.* provably accurate ε-approximations to K_a (see Section 1.2.2), of the 15 *de novo* D-peptide CALP-PEPs exceeding that of CFTR's C-terminal SliM indicates that the CALP-PEPs meet their fundamental design objective. For example, CALP-PEP9 has a ΔΔG of -2.3 kcal/mol, improving K_D 46-fold, compared to the CFTR C-terminus (see Appendix B.1). While not predicted to bind as tightly as kCAL01, D-peptides have therapeutic advantages over L-peptides, including improved metabolic stability (described in Section 3.1.1), that can compensate for not reaching the binding affinity of the strongest CALP peptide inhibitor. The red bars show the number of β-strand H-bonds contributing to the common β2-sheet extension PDZ-binding motif. The green bars show the number of H-bonds between the peptide's C-terminal carboxylate and the CBL. The number of CBL and β-strand H-bonds varies across the CALP-PEPs, but the one predicted to bind tightest, CALP-PEP9, has 3 CBL and 3 β-strand H-bonds, the same number CFTR and kCAL01 have. The K* scores of the CALP-PEPs and empirical structures were determined using the K* algorithm[17,21] in OSPREY. Section 1.2.2 provides a definition of the K* algorithm and K* score. The error bars on the K* scores show the provable upper- and lower-bound of the K* approximation. The number and type of H-bonds between the peptides and CALP were determined using Pymol[176].

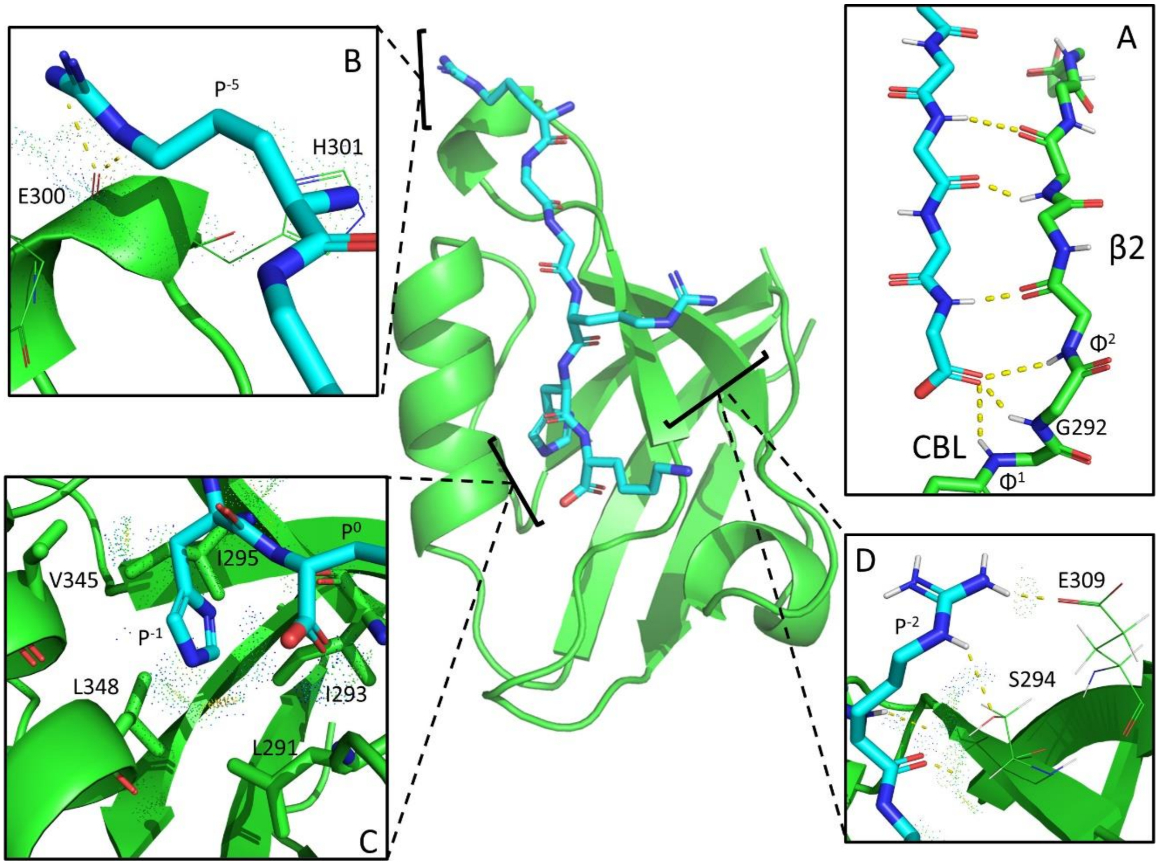


Figure 11: Structural analysis of OSPREY-generated ensemble of CALP-PEP9. Of the 15 CALP-PEPs, CALP-PEP9 (RGGRHK) is predicted to be the tightest binder to CALP with a $\log_{10}$ K* score of 26.1, which approaches the predicted affinity of the most binding-efficient L-peptide inhibitor kCAL01 (previously reported[23] by our lab; with a $\log_{10}$ K* score of 30.4), and vastly exceeds the predicted binding affinity of the CFTR C-terminal SliM ($\log_{10}$ K* score of 16.2). CALP-PEP9 is predicted to improve K_D by 46-fold over the CFTR C-term, with a normalized K_D of 8.9 μM versus 420 μM[183] for the C-terminal SliM (see Appendix B.1). **(A)** CALP-PEP9's P^0 carboxylate forms favorable H-bonds with the carboxylate binding loop (CBL: G290-I293, $G\Phi^1G\Phi^2$) and strand β2, mimicking canonical PDZ binding interactions[134] of L-peptides. **(B)** The amino acid at position P^{-5} in CALP-PEP9 is arginine. P^{-5}'s amino group and sidechain make favorable van der Waals contacts, indicated by blue and green MolProbity dots[167–169], with CALP's E300 and H301. Its guanidino sidechain also forms H-bonds with E300's carboxyl group. **(C)** In canonical L-peptide's, a PDZ-domain's hydrophobic pocket is filled by a hydrophobic amino acid at position P^0.[134] In contrast, all of the CALP DPRs fill the pocket with the amino acid at position P^{-1} (see Figure 17). CALP's hydrophobic pocket, defined as the groove between α2 and β2 and involving V345, I295, I293, L291, and L348, is filled by a histidine in position P^{-1}. **(D)** CALP-PEP9's P^{-2} is arginine, which is predicted to make favorable van der Waals contacts, form an H-bond with β2's S294 and a salt bridge with β3's E309. The K* scores and additional structural validation of all the CALP-PEPs can be found in Table 12.

3.3.3 Designed inhibitors targeting MAST2

Using an NMR structure of PTEN bound to MAST2 (PDB ID 2kyl)[174], we used DexDesign to generate 3 DPR scaffolds: MAST2-DPR[1-3] (see Table 11 on page 147). From these 3 DPR scaffolds, we used the design techniques described in Section 3.2.3.1 to generate 15 peptides, MAST2-PEP[1-15]. Figure 12 shows an overview of the MAST2-PEPs K* scores and how they compare to MAST2's endogenous ligand PTEN. Additional structural information about the MAST2-PEPs, such as which residue fills MAST2's hydrophobic cavity, is in Table 13 on page 149.

PTEN binds MAST2 209-fold tighter than CFTR binds CALP (K_D = 1.9 ± 0.05 µM vs. 420 ± 80 µM)[183,184] and binds MAST2 as tightly as the strongest known inhibitor of CALP, kCAL01 (K_D = 2.3 ± 0.2 µM)[23]. In other words, to design competitive inhibitors of the MAST2:PTEN interaction requires us to design D-peptide inhibitors with a better affinity than the tightest known L-peptide inhibitor of CALP. Despite the challenge inherent in disrupting the MAST2:PTEN interaction, all the MAST2-PEPs are predicted to bind MAST2 with affinities surpassing PTEN (see Figure 12).

The $\log_{10}$ K* scores of the MAST-PEPs range from a low of 29.4 for MAST2-PEP9 to a high of 32.7 for MAST2-PEP4. MAST2-PEP4 is the best DexDesign-generated inhibitor and is predicted to bind MAST2 with a normalized Gibbs free energy ΔG of -8.8 kcal/mol, a -1.1 kcal/mol improvement over MAST2:PTEN, resulting in a 5-fold improvement in K_D. In some cases a 5-fold improvement might be considered small, but we have previously shown[23,34] that differences of this magnitude can have profound effects on biological activity. For example, kCAL01 binds only 6-fold tighter than previous competing peptides, such as iCAL35, that were discovered via high-throughput

SPOT arrays[23]. However, in *ex vivo* assays (see Section 3.1.2.1) iCAL35 had non-significant biological activity[23], whereas the 6-fold tighter binding kCAL01 had significant biological activity. Since the SLiMs modulate a delicate network of competing affinities and specificities[129,130,183], 5-7x improvements in affinity (such as that achieved by CALP-PEP4) can make the difference between failure and true biological activity.

The MAST2-PEPs replicate some of the canonical L-peptide PDZ-binding motifs, such as the residue in position P^0 filling the hydrophobic pocket between the PDZ domain's α2 helix and β2 strand. 9 out of 15 MAST2-PEPs have residue P^0 filling the hydrophobic pocket (see Table 13). This contrasts with the CALP-PEPs, where in all cases the residue at position P^{-1} filled the hydrophobic pocket. In the best predicted inhibitor, MAST2-PEP4, the P^0 leucine fills MAST2's hydrophobic cavity formed by Tyr17, Phe19, Val77, Ile79, and Leu81 (see Figure 13, C & D). In contrast to PTEN's P^0 valine, MAST2-PEP4's P^0 leucine forms favorable interactions with all 5 of the cavity's hydrophobic residues. In addition, a rotation of MAST2-PEP4's C-terminal carboxylate alleviates a steric clash with the carboxylate binding loop present in the MAST2:PTEN complex.

The MAST2-PEPs also exploit novel geometric features of D-peptides not available to their L-counterparts. For example, MAST2-PEP4's P^{-3} glutamate makes favorable van der Waals contacts with MAST2's His73 imidazole side chain (see Figure 13, A). PTEN does not make the analogous interaction, and instead PTEN's P^{-3} isoleucine is oriented towards MAST2's β2 strand, and the residue nearest to His73 is a glutamine at P^{-4}, whose amide fails to make van der Waals contacts with His73 (see Figure 13, B). The creation of novel favorable interactions with MAST2 is common in the designed MAST2-PEPs and compensates for the loss of some of the canonical PDZ-domain binding motifs. We discuss the trade-offs between replicating canonical interactions and finding new modes of binding available to D-peptides in Section 3.4.

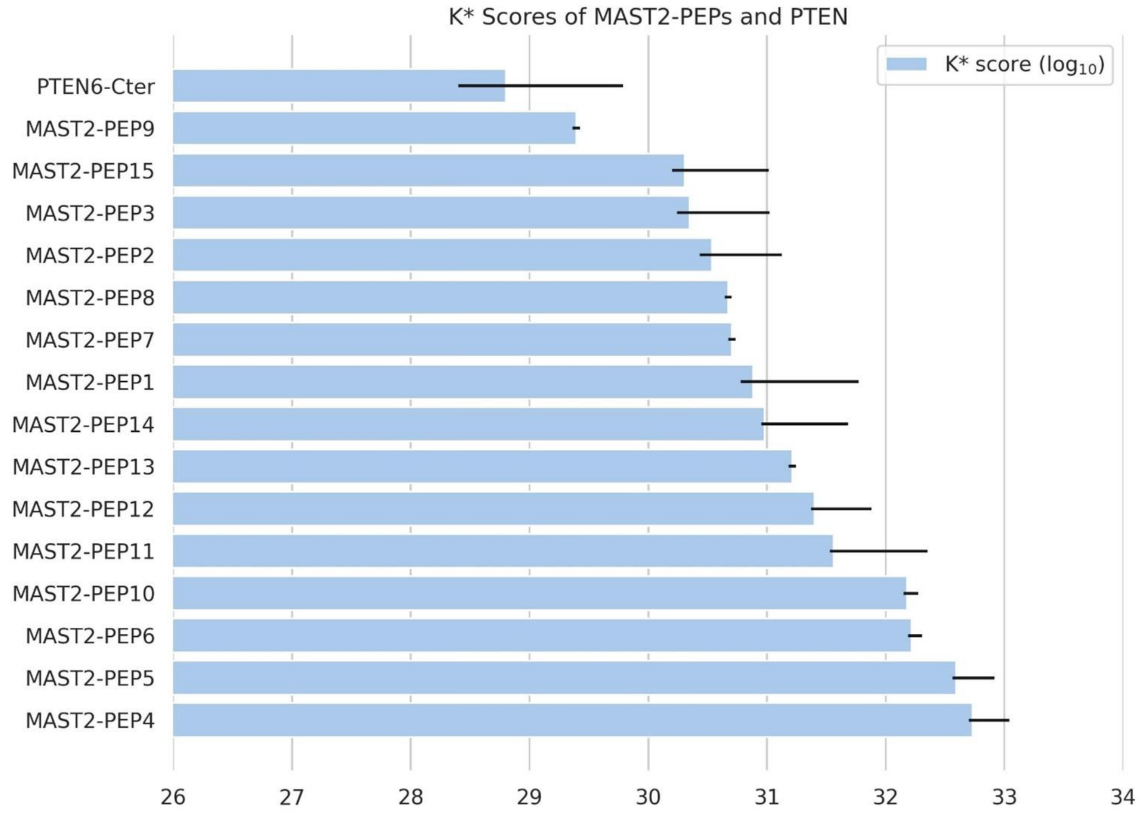


Figure 12: The DexDesign-generated D-peptides are predicted to bind to MAST2 tighter than PTEN. Blue bars show the OSPREY-predicted binding affinity of the MAST-PEPs and PTEN6-Cter (hereafter denoted PTEN6, the 6 C-terminal residues of MAST2's endogenous ligand PTEN). We used the DexDesign algorithm (described in Section 3.2.1) and novel design techniques (described in Section 3.2.3.1) to generate 15 *de novo* D-peptides predicted by the K* algorithm[21,47] to bind MAST2 tighter than PTEN6. Notably, PTEN binds MAST2 209-fold tighter than CFTR binds CALP (K_D = 1.9 ± 0.05 µM vs. 420 ± 80 µM)[183,184], and binds MAST2 as tightly as the strongest known inhibitor of CALP, kCAL01 (K_D = 2.3 ± 0.2 µM)[23], indicating a more challenging target of inhibition. OSPREY predicts PTEN6 to bind MAST2 with a $\log_{10}$ K* score (a provably accurate ε-approximations to K_a, see Section 1.2.2) of 28.8. Despite the more difficult target, all the MAST2-PEPs have higher $\log_{10}$ K* scores than PTEN6, ranging from 29.4 for MAST2-PEP9 to 32.7 for MAST2-PEP4, meaning the MAST-PEPs are predicted to outcompete PTEN6 and inhibit PTEN6:MAST2 binding. The best predicted inhibitor, MAST2-PEP4, has a ΔΔG of -1.1 kcal/mol, improving K_D 5-fold compared to the PTEN6 (see Appendix B.1). The K* scores of the MAST2-PEPs and empirical structures were determined using the K* algorithm[17,21] in OSPREY[35]. Section 1.2.2 provides a definition of the K* algorithm and K* score. The error bars on the K* scores show the provable upper- and lower-bound of the K* approximation.

3.4 Discussion

3.4.1 Replication and Restitution: a framework for evaluating *de novo* peptides

Though the CALP-PEPs and the MAST2-PEPs both target PDZ domains, the interactions they make with their respective targets can be generally categorized into one of two kinds: replication or restitution. *Replication* means to replicate interactions previously observed in L-peptide inhibitors, for example, in the case of the PDZ domains, the β2 strand extension, residue P^0 filling the hydrophobic pocket, and the H-bond network a peptide's terminal carboxylate makes with the CBL. *Restitution*, on the other hand, refers to the process of compensating for typical L-peptide binding motifs by making novel interactions that are now possible to explore due to the change from L- to D-chirality of the peptide. Whereas in some cases we achieve improved binding through replicating the canonical PDZ interactions, in other cases, due to the special geometric properties of D-peptides, we instead observe an increase in binding (restitution) due to novel interactions that we observe in the structures that were not available to L-peptides. This suggest the intriguing possibility that some peptides may be stabilized by replicating native-like interactions from L-peptides, whereas others might be stabilized by forming novel interactions, available only to the ligands with D-configuration of peptides.

In the *de novo* peptides we generated, both the CALP-PEPs and the MAST2-PEPs contained elements both replicating and restituting the binding interactions formed by the endogenous L-peptides from which their backbone conformations are derived (see Figure 6). In general, the CALP-PEPs relied more on a strategy of replication to improve binding affinity, whereas the MAST2-PEPs exploited a strategy of restitution. Take for example the SLiM's canonical C-terminal carboxylate H-bond network formed with the CBL. Whereas 13 of the 15 CALP-PEPs replicated (to varying degrees, and sometime even exceeding the number of) H-bonds formed with the CBL (see Figure 10), the MAST2-PEPs' terminal carboxylate tended to have few, if any, H-bonds with the CBL (see, e.g., Figure 13 C). We postulate that the reason for the MAST-PEPs' use of restitution instead of replication for the CBL H-bond network is that the structure of the PTEN:MAST2 complex (PDB ID 2kyl)[174] indicates the existence of steric clashes between the C-terminal carboxylate and the CBL (see Figure 13 D). When using MolProbity[167] to evaluate the lowest-energy model of the empirical NMR structure, it indicates there is a bad clash (van der Waal radii overlap of 0.518 Å) between PTEN P^0 valine's OXT atom and the HA atom from Lys16 in MAST2's CBL. OSPREY's energy function[35,47,124,173], which uses continuous sidechain minimization in addition to translation and rotation of the peptide to minimize the energy of each conformation evaluated by OSPREY's iMinDEE/K* algorithm[17,47], pushes and rotates the C-terminal carboxylate away from the CBL to alleviate the steric clash (see Figure 13 C), with the trade-off being the loss of replication of some of canonical PDZ CBL H-bond interactions.

The CALP-PEPs and MAST2-PEPs also exploit restitution to form novel favorable interactions with their target proteins. One type of restitution is the creation of favorable new side chain interactions between the peptides and their targets. For example, CALP-PEP9's P^{-2} arginine forms a new H-bond with CALP β2 strand's S294 and a salt bridge with β3's E309 (see Figure 11, D) that are absent in the CALP:kCAL01 structure (PDB ID 6ov7)[22]. Interestingly, CFTR's P^{-1} arginine forms an analogous H-bond with

E59 CALP:CFTR structure (PDB ID 2lob)[182], providing evidence that this interaction, as restituted in CALP-PEP9 (based on the kCAL01 structure lacking this interaction), is both plausible and favorable *in vitro*. The MAST2-PEPs likewise restitute novel favorable interactions. For example, MAST2-PEP4's P^{-3} glutamate makes favorable van der Waals contacts with MAST2’s His73 imidazole side chain (see Figure 13 A). In contrast, PTEN cannot make some favorable interactions available to MAST2-PEP4. In the MAST2:PTEN structure, PTEN's P^{-3} isoleucine is oriented towards MAST2’s β2 strand, and the residue nearest to His73 is P^{-4} is glutamine, whose amide fails to make van der Waals contacts with His73 (see Figure 13 B). In the future, we believe that designed D-peptide libraries of binders and inhibitors can be characterized as falling on a spectrum ranging from replication (1) to restitution (-1) and can be visualized as a per-residue replication-restitution score ranging from 1 to -1. In this way, the functional contributions to binding of *de novo* peptides could be mapped into a vector space which can be visualized or exploited as novel features for machine learning design approaches.

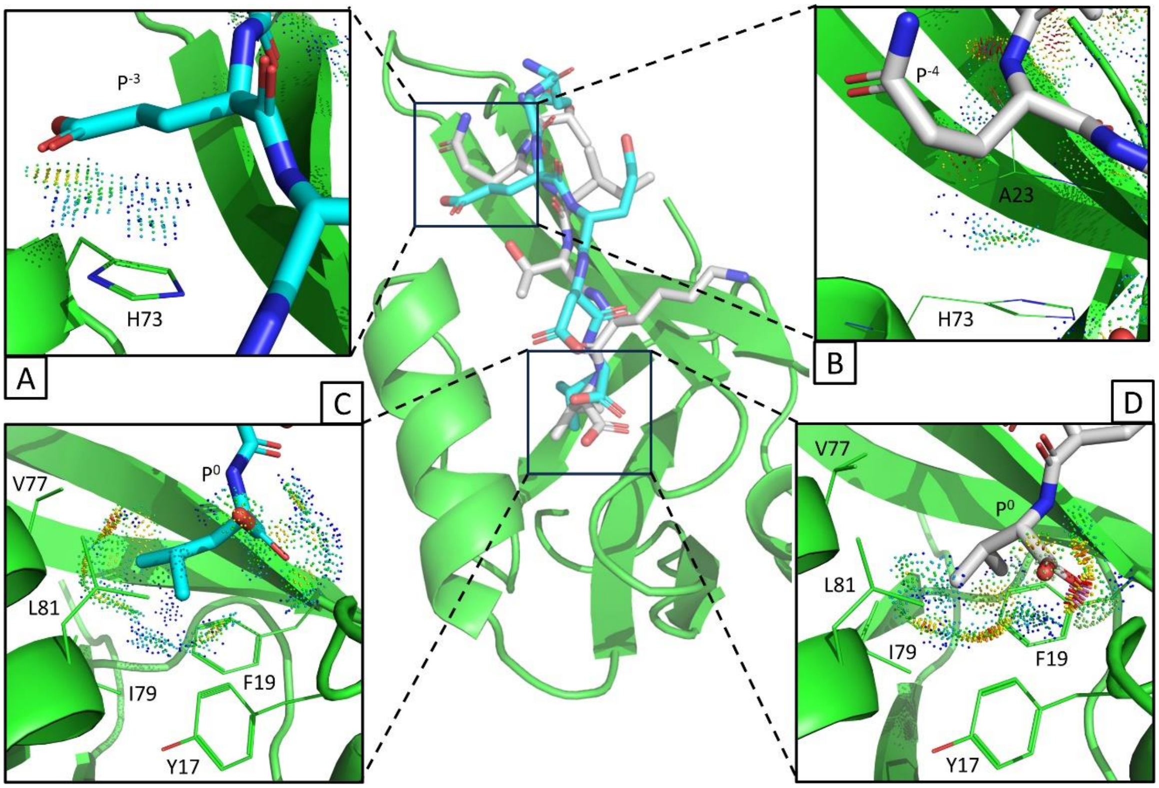


Figure 13: MAST2-PEP4 creates novel favorable interactions with MAST2 not found in PTEN. MAST2-PEP4 (cyan sticks) is the DexDesign-generated *de novo* D-peptide predicted to bind MAST2 (green cartoon and lines) with the tightest affinity. The K* algorithm[21,47] in OSPREY[35] predicts MAST2-PEP4 to bind MAST2 with a $\log_{10}$ K* score (a provably accurate ε-approximations to K_a, see Section 1.2.2) of 32.7, compared to 28.8 for PTEN6 (gray sticks, the 6 C-terminal residues of MAST2's endogenous ligand PTEN). After normalization (see Appendix B.1), the Gibbs free energy change ΔG of the MAST2:MAST2-PEP4 complex is -8.8 kcal/mol, a -1.1 kcal/mol improvement over MAST2:PTEN6, resulting in a 5-fold improvement in K_D. **(center)** The lowest-energy conformation from the OSPREY-predicted conformational ensemble of MAST2 (green) bound to MAST2-PTEN6 (cyan) and the lowest-energy model of PTEN6 (grey) from an empirical solution NMR ensemble of the MAST2:PTEN complex (PDB ID 2kyl)[174]. In comparison to the binding modes of, e.g., the CALP-PEPs to CALP (see Figure 11) which largely *recover* canonical PDZ-domain binding interactions, MAST2-PEP4 *restitutes* binding to MAST2 by exploiting novel geometric features of D-peptides not available to their L-counterparts (see Section 3.4.1). **(A)** MAST2-PEP4's P^{-3} glutamate makes favorable van der Waals contacts with MAST2's His73 imidazole side chain, as indicated by predominantly green and blue MolProbity dots[167–169]. These favorable contacts are absent in the MAST2:PTEN6 complex. **(B)** In contrast to **(A)**, PTEN6 cannot make some favorable interactions available to our D-peptides. For example, in MAST2:PTEN6, PTEN6's P^{-3} isoleucine is oriented towards MAST2's β2 strand (not shown), and the residue nearest to His73 is P^{-4} glutamine, whose amide fails to make van der Waals contacts with His73. **(C)** MAST2-PEP4's P^0 leucine fills MAST2's hydrophobic cavity[184] formed by Tyr17, Phe19, Val77, Ile79, and Leu81. In contrast to PTEN6's P^0 valine **(D)**, MAST2-PEP4's P^0 leucine forms favorable interactions, as indicated by the green and blue MolProbity dots, with all 5 of the cavity's hydrophobic residues. In addition, a rotation of the C-terminal carboxylate alleviates a steric clash (indicated by the red and pink MolProbity dots in **D**) with the carboxylate binding loop present in the MAST2:PTEN6 complex.

3.4.2 Validation of DexDesign scaffold discovery and redesign

To assess DexDesign, we performed an experiment to measure the ability of DexDesign to design a *de novo* D-peptide similar to a D-peptide found in an empirical D-peptide:L-protein complex. In our experiment, we began with the crystal structure of a known D-peptide in complex with an L-protein and applied a global reflection to the D-peptide:L-protein complex. Then, we employed MASTER to search the L-database using the resulting, now flipped, L-peptide as the query. The returned L-structures were aligned with the reflected complex and reflected once again to produce D-peptide:L-protein scaffolds ordered by increasing backbone alignment RMSD. The first, and therefore lowest, RMSD backbone alignment was then selected for redesign using OSPREY. We then measured the similarities of the redesigned D-peptide to the empirical D-peptide. While DexDesign is intended for construction of novel *de-novo* D-peptides, the capability to generate a DPR similar to the D-form empirical structure should validate our redesign protocol.

We selected a crystal structure of the D-amino acid containing peptide GyGlanvdessG in complex with streptavidin (PDB ID 5n8j)[185]. Streptavidin is a homotetrameric protein that binds the vitamin biotin with high affinity[186], and is therefore commonly used in Western blotting and immunoassays[187]. A monomer of streptavidin forms a β barrel, with ligands oriented towards the interior of the barrel. Similar to CBL interactions with CALP and MAST2, streptavidin forms favorable hydrogens bonds with ligands via a flexible binding loop[188]. Analogously, streptavidin exhibits hydrophobic contributions through inward-facing tryptophan residues of the β barrel[189]. Therefore, a high-affinity ligand should establish hydrogen bonds with the binding loop while orienting hydrophobic residues towards β barrel tryptophans. We selected this system due to its comparable D-peptide size and similar chemistry to PDZ domains.

We sourced the lowest backbone RMSD (0.48 Å) of inverted D-amino acid GyGlanvdessG from chain A residues 608 to 616 of ST0929 (PDB ID 3hje)[190], a glycol transferase. After application of Minimum Flexible Set, Inverse Alanine Scanning, and K*-based Mutational Scan to this scaffold (see Section 3.2.3.1), we determined the optimal binder, hereafter denoted as DPRV, to have a $\log_{10}$ K* score of 32.8 with the sequence WWMIGDWND. This differed slightly from GyGlanvdessG (GLANVDESS), which has a $\log_{10}$ K* score of 32.2. The sequence similarity between the two peptides is 21.43%, a degree of native sequence recovery comparable to reported recovery in popular protein design programs such as Rosetta for L-proteins[191]. This is especially true for NMR structures, such as we used in our MAST2 study (see Section 3.3.3). With DPRV, we report that DexDesign generates a D-ligand with chemistry unique to the DPR scaffold.

While DexDesign exhibits comparable performance to state-of-the-art methods[191], native sequence recovery on a short (9 residue) peptide may be a poor indicator of ligand binding. For example, a 40% sequence recovery equates to 3.6 residues for our redesigned peptide. This is a small number of residues, and likely fails to capture the geometric and chemical features that drive high affinity. To investigate the similarities of GyGlanvdessG and DPRV, we also report the backbone alignment RMSD of DPRV to GyGlanvdessG:streptavidin (0.48 Å), and the geometric, chemical, and biophysical properties of our designed peptide that enable binding (see Figure 14). Finally, we report the $\log_{10}$ K* scores computed over molecular ensembles as validation of binding competency (above and see Table 14 on page 150).

We also performed a control experiment wherein we mutated the ST0929 scaffold sequence (RYEEGLFNN) directly to the sequence of the D-peptide GyGlanvdessG, without using any OSPREY-based techniques such as K*-based Mutational Scan and Inverse Alanine Scanning. The purpose of this experiment was to investigate the predicted binding of the GyGlanvdessG sequence on the ST0929 scaffold backbone. This control experiment produced a $\log_{10}$ K* score of only 26.6, a difference of -5.7 from GyGlanvdessG. Interestingly, a 100% wildtype sequence recovery mutant yields lower predicted binding despite the selection of a DPR scaffold with the lowest backbone RMSD. Therefore, we conclude that the GyGlanvdessG sequence was not recovered by the full DexDesign protocol because these novel OSPREY-based design techniques would not permit optimal binding on the lowest backbone RMSD scaffold. Instead, the DexDesign techniques (as outlined in Section 3.2.3.1) resulted in a novel D-peptide. These results highlight the sensitivity of the peptide design to the starting scaffold; even similar-appearing scaffolds have different degrees of designability due to geometric differences between backbones. Overall, our experiment highlights the utility of DexDesign for generation of novel peptides, as opposed to sequence recovery of known binders.

The difference between amino acid composition of the D-amino acid containing peptide GyGlanvdessG and the redesigned peptide is likely due to subtle differences in scaffold geometry. As shown in Figure 14 and Figure 18, the backbones of D-amino acid GyGlanvdessG, DPRV, and the ST0929-sourced peptide scaffold mutated to the endogenous ligand (control) vary at residues important for establishing hydrogen bonds. For example, GyGlanvdessG's Glu7 residue makes hydrogen bonds with residues Asn23 and Ser27 of streptavidin (Figure 18 A). DPRV's Trp7, which is shifted 1.9 Å away from streptavidin Ser27 in comparison to GyGlanvdessG, does not form either of these hydrogen bonds. However, DRPV's Asp9 facilitates hydrogen bond formation with residues Ser45 and Ser52 on streptavidin (Figure 14 B). These residues belong to the flexible binding loop, where favorable contacts are crucial for high-affinity binding[186]. The hydrogen bond formed with flexible loop residue Ser52 is unique to DRPV and is not present in the ST0929 control experiment or GyGlanvdessG. Therefore, a peptide that replicates some characteristics of a known ligand, while restituting novel interactions, may be a much more competent binder.

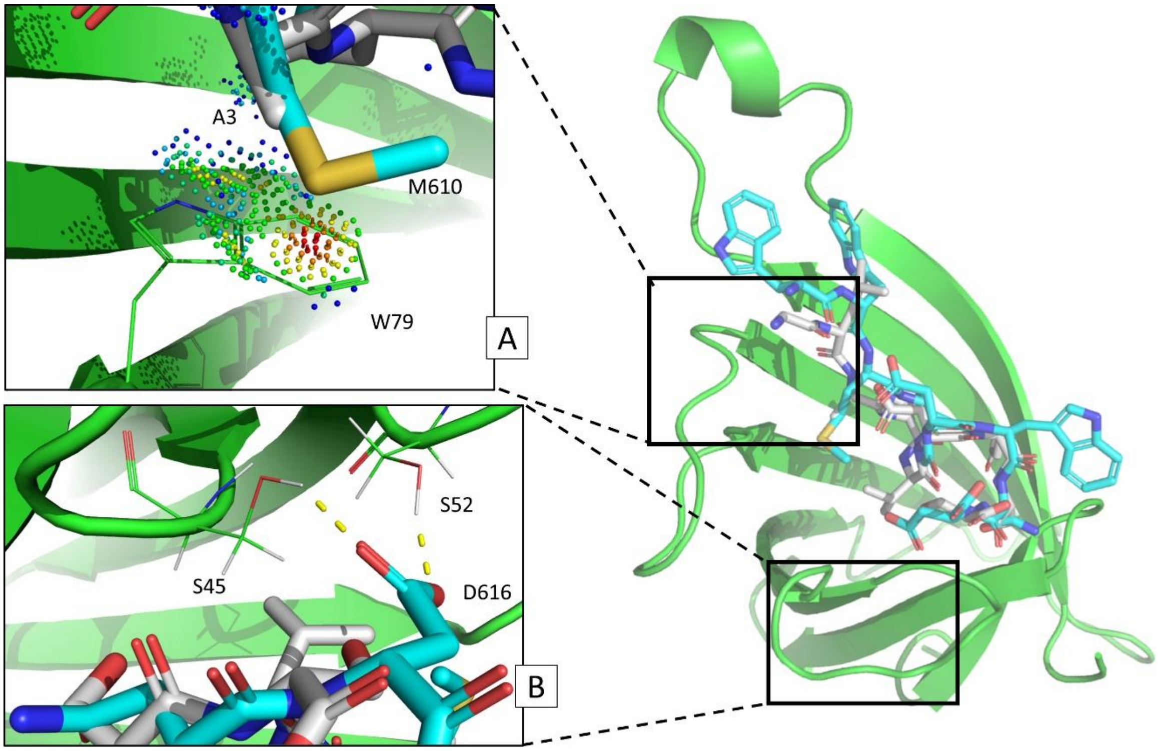


Figure 14: Geometric, chemical, and physical properties of DPRV that drive binding to streptavidin. (A) Similar to GyGlanvdessG Ala3, DRPV Met610 displays favorable hydrophobic and van der Waals contacts with streptavidin Trp79. Streptavidin (green cartoon and lines) displays hydrophobic contributions through inward-facing tryptophan residues of the β barrel, which have been reported as important for ligand binding[189]. Favorable van der Waals interactions are shown as green and blue dots between DRPV (cyan) Met610 and streptavidin Trp79. GyGlanvdessG (grey) Ala3 also shares similar hydrophobic and van der Waals contacts with streptavidin Trp79. **(B)** Differences in C-terminal orientation and interactions between D-peptide GyGlanvdessG:streptavidin and DPRV:streptavidin. Unlike GyGlanvdessG, DRPV's aspartic acid C-terminus makes hydrogen bonds with both streptavidin residues Ser45 and Ser52. However, GyGlanvdessG's C-terminal Ser9 fails to form a hydrogen bond with streptavidin Ser52. Residues number Ser45-Ser52 on streptavidin are known to be important for establishing binding of biotin[186], so these contacts are likely evidence of high-affinity binding of DPRV.

3.5 Conclusions

In this chapter, we presented a new algorithm, DexDesign, to computationally design *de novo* D-peptides. In addition, we have presented three new computational protein design techniques, the Minimum Flexible Set, Inverse Alanine Scan, and K*-based Mutational Scan, that are generally applicable to both D- and L-peptide design. The process of developing DexDesign required us to add new capabilities to the OSPREY[35] protein redesign software, including the ability to add arbitrary conformation libraries. This enables exciting new opportunities for the types of chemistries OSPREY can model and the conformations its algorithms can search and optimize. With DexDesign we have added a D-amino acid conformation library to OSPREY by reflecting the L-version of OSPREY's standard protein conformation library. Future designs and algorithms that model non-proteinogenic molecular building blocks, such as non-canonical amino acids or small molecule rotamers, are now substantially easier to implement. We envision providing additional generally useful standard conformation libraries within OSPREY itself in the future, and protein designers with specialized use cases can create their own conformation libraries and import them in their design specification in a trivial and code-free process.

We have used DexDesign to generate and optimize 30 *de novo* D-peptide inhibitors for two biomedically important PDZ targets: CALP and MAST2. We used provable approximations of binding affinity (see Section 1.2.2) and analyzed the OSPREY-predicted low-energy ensembles of the bound D-peptide:target structures to assess the quality of the novel peptides. We employed a novel restitution-replication framework for analyzing the basis upon which our DexDesign-generated D-peptides improved binding compared to their targets' endogenous ligands. There are many other peptide-recognizing PDZ domain targets for which one could use DexDesign to design *de novo* D-peptide inhibitors. Furthermore, DexDesign is not restricted to PDZ-domains, it could be applied to design novel antineoplastic, antifungal, or antibiotic D-peptide therapeutics. It is a general algorithm applicable to any target for which there exists structural models of a peptide:target complex. The structural models can be determined experimentally or computationally predicted using machine learning-based algorithms such as AlphaFold[13,14], although the accuracy of the results may be somewhat diminished compared to experimentally determined structures of ligand-target complexes.

There are many opportunities for application or extension of DexDesign in future work. One possibility could be to develop and incorporate additional computational methods of assessing the DexDesigned peptides. Another would be to experimentally validate DexDesigned peptides *in vitro*. Beyond CALP and MAST2, there are many other exciting and biomedically relevant protein targets for which DexDesign could be an enabling algorithm for the development of novel D-peptide therapeutics. Thus, DexDesign provides an important tool to the drug discovery community interested in therapeutic design.

4 Conclusion

In this thesis, I have focused on two biomedically relevant topics: predicting drug resistance and *de novo* D-peptide design. Drug resistance is a pernicious problem. One of the most underappreciated aspects of living in a wealthy country in the 21st century is easy access to basic medications like antibiotics when suffering from, for example, a urinary tract infection. Yet one needs to travel back in time only about a hundred years to encounter a world where a urinary tract infection could be deadly. At present we might take the efficacy of antibiotics for granted, but it does not take a particularly strong imagination to envision a future in which widespread antimicrobial resistance renders our current slate of antibiotics ineffective. Such a future may be closer than we think: antimicrobial resistance is *currently* a global public health threat contributing to over 5 million deaths a year[63].

It is not just bacteria that evolve to beat the drugs meant to kill them; fungi[192] and viruses[67,193,194] do, too. Even our own body's cells can develop resistance to drugs we may need to take to keep us alive[64–66]. Acquired resistance to antineoplastics is unfortunately a common problem for many cancer patients. If we were able to predict how cancers, bacteria, fungi, and viruses will develop resistance to a drug, we might be able to redesign the drug to be less prone to resistance.

In Chapter 2 and Appendix C, we presented Resistor, a novel algorithm that combines structure and sequence data to predict resistance mutations. Resistor computes the Pareto frontier of four resistance-causing criteria: the change in binding affinity (ΔK_a) of the (1) drug and (2) endogenous ligand upon a protein's mutation; (3) the probability a mutation will occur based on empirically derived mutational signatures; and (4) the cardinality of mutations comprising a hotspot. To validate Resistor, we applied it to kinase inhibitors targeting EGFR and BRAF in lung adenocarcinoma and melanoma. Resistor correctly identified eight clinically significant EGFR resistance mutations, including the "gatekeeper" T790M mutation to erlotinib and gefitinib and five known resistance mutations to osimertinib. Furthermore, Resistor predictions are consistent with sensitivity data on BRAF inhibitors from both retrospective and prospective experiments using the KinCon biosensor technology[60–62].

One can think of the Resistor as an algorithmic reduction of an extremely difficult biological problem, namely the act of predicting evolution—the evolutionary mechanisms by which a particular cancer cell population, in order to survive, develops resistance to therapeutics—to a standard application of protein design algorithms and multi-objective optimization. Given the largely positive results Resistor achieved with retrospective and prospective experimental validation (Sections 2.2.10 and 2.2.11), this reduction appears to be appropriate, at least in the case of resistance via escape mutation. Together with our earlier work on resistance in the infectious disease space[29,72], our reduction of resistance caused by escape mutation to a provable combination of negative and positive protein design represents the first systematic attempt to computationally predict resistance mutations at scale based on geometric and physical reasoning, and represents a major step forward towards showing that it is possible to predict, using principled and provable algorithms, certain types of evolution on the computer.

Resistance prediction in oncology brings with it unique challenges. In cancer, the exact same protein with the exact same mutations may have a different resistance profile depending on the tissue in which it is located (e.g., liver versus lung) or a different type of cancer (e.g., lung adenocarcinoma versus colorectal cancer). These are factors we did not have to consider in our previous work predicting antimicrobial resistance[29,72]. To account for this, Resistor follows Kaserer and Blagg's example[28] and incorporates mutational signatures[81,82] into the reduction paradigm outlined above. Mutational signatures are “echoes” of mutational processes occurring within the cell[81] for which we lack biophysical and structural information. In the future we would like to obtain the relevant structural and mechanistic information summarized by the mutational signatures to be able to model the mutational process as molecules and forces, though until then, due to the special nature of cancers (as compared to bacteria and viruses), Resistor will continue to use the signatures as proxies for the underlying mutational process.

As mentioned above, our lab previously used positive and negative K* design to predict resistance mutations to propargyl-linked antifolates in dihydrofolate reductase from *Staphylococcus aureus*[29], which we later confirmed *in vivo*[29,77]. Our collaborator Teresa (who is also an author of Resistor) subsequently extended this work by incorporating mutational signatures[81,82] and hotspot scores into a sequential computational workflow to predict prospective resistance mutations to small molecule inhibitors of KIT, EGFR, Abl, and ALK[28].

Resistor improves upon these previous methods by using multi-objective optimization to provide a view of the entire resistance landscape. Resistor incorporates the same four criteria as Teresa's earlier work[28]. In contrast to the previous protocol (which analyzed potential resistance mutations in the top three hotspots, ranked by reIP)[28] Resistor uses Pareto optimization over the four axes to calculate a Pareto rank for each potential resistance mutation. Ranks provide a mechanism by which a medicinal chemist can prioritize investigation without filtering out potentially important resistance mutations that may not be in one of the top three hotspots. This is a critical distinction—for example, the most common resistance mutation in EGFR encountered in the clinic is T790M, and if we had only looked at the top three hotspots we would have missed this important resistance mutation as it's in the 5th-ranked hotspot (see Table 4). In contrast, Resistor correctly placed it on the Pareto frontier (see Table 1).

In addition, we developed a new graph-based algorithm (Figure 2) for calculating and assigning cancer-specific mutational probabilities to each potential resistance mutation. We also developed new YAML OSPREY design file specifications that enable code-free use of Resistor and OSPREY for users who prefer not to write code (see Figure 22 and Figure 23 on page 175 for an example). And to facilitate dissemination and scientific reproducibility, we have provided detailed step-by-step instructions for the entire protocol, in which we demonstrate using Resistor to predict ERK2 resistance mutations to the inhibitor SCH772984 (Appendix C).

In Chapter 3, we introduced DexDesign, a novel algorithm for computationally designing *de novo* D-peptide inhibitors. DexDesign leverages three novel techniques that are broadly applicable to computational protein design: the Minimum Flexible Set, K*-based Mutational Scan, and Inverse Alanine Scan. In Section 3.3 we applied these techniques and DexDesign to generate novel D-peptide inhibitors of two biomedically important PDZ domain targets: CALP and MAST2. Section 3.4.1 introduced a new framework for analyzing *de novo* peptides—evaluation along a replication/restitution axis—which we applied to the DexDesigned D-peptides. Notably, the peptides we generated are predicted to bind their targets tighter than their targets' endogenous ligands, validating the peptides' potential as lead therapeutic candidates.

I wrote a lot of open source software to enable the research described in the previous paragraph. For example, I wrote programs that generated D-peptide:L-protein complexes from MASTER's[57] search results. As we would often obtain thousands or tens of thousands of matches of varying quality, it was infeasible for us to analyze the results manually. So, I wrote a program that processed and computed statistics over all the results in one go. We were interested in statistics such as solvent accessible surface area and the full backbone RMSD to the query structure.

To add the ability to design with D-amino acids in OSPREY, I added and modified many OSPREY components. For example, many of the steps necessary for preparing a molecule for design, such as bond identification, protonation, minimization, and others, rely on Amber[171] programs which, through trial-and-error, I discovered not to be invariant to chirality. I solved problems in this class by reflecting the D-structure into L-space, invoked the relevant Amber program with the L-peptide as its argument, and then again reflected the peptide back into D-space for further molecular preparation in OSPREY. To support minimization of D-proteins and peptides in OSPREY, I added the ff14SB forcefield[195], which Professor Carlos Simmerling of Stony Brook (the forcefield's author) confirmed to me is invariant to chirality. I created a D-conformation library, which contains definitions of molecular fragments pertaining to all amino acid types and the Richardsons' rotamer library[49], by reflecting OSPREY's standard L-conformation library. Lastly, I added the ability to set a per-chain chirality to the OSPREY user interface, so that when a user sets a residue's mutations and rotamers they are set with templates of the correct chirality.

Finally, below I share some reflections I have made during the process of carrying out the research described above. In a sense, these are lessons I wish I had known earlier in my graduate career, as many of them were learned the hard way—by expending time and energy on pursuits that were ultimately unfruitful. It is my hope that sharing these reflections will enable future computer scientists or computational biologists to learn from my mistakes and avoid making the same ones.

4.1 Successful Computational Protein Design Requires Knowledge of Algorithms, Biochemistry, and Engineering

Imagine, for a moment, a world where thanks to infinite resources computers never crash; combinatorial, #P-hard algorithms completed instantaneously; proteins were trivially modeled in solvent with unlimited flexibility; and energy functions instantaneously and accurately calculated the integral of a protein's energy over time. In this world, the job of a protein designer is easy: they would choose the best provably accurate algorithm to model a protein *in silico* (as computational complexity is irrelevant), model each residue backbone, sidechain, and individual atoms as infinitely flexible and bathed in explicit solvent (as this has no additional cost), and integrate a continuous Boltzmann distribution over every possible sequence. In this imaginary world, the protein designer would not need to understand algorithmic complexity, to make trade-offs in choosing the type and requisite accuracy (and thus cost) of different biophysical models, or to use simplified energy functions for computational tractability. In essence, there would be no hard choices.

Back in our world, computers have limited resources and sometimes crash, #P-hard algorithms often take a long time to complete, and computational costs increase with improved modeling accuracy of the biophysics and energetics. In our reality, the job of a protein designer is nothing if not making hard choices between competing modeling priorities. One typical choice a protein designer must make is between a provable and stochastic protein design algorithm (described in Section 1.2.1). The designer knows that provable algorithms are more reliable but might wonder if their predictions will be ready by an experimental collaborator's deadline. On the other hand, they are confident the stochastic algorithm will complete in time but wonder how the collaborator might react if none of its predictions match their experimental data. It is in making these types of choices that a designer's knowledge of algorithms, biochemistry, and engineering provides a useful framework with which to provide reliable answers in the required timeframe.

I learned the utility of understanding all three pillars of this framework through experience. As a computer scientist, I had a firm grasp on the algorithms but was less fluent in biochemistry. For these reasons, my initial approach to computational protein design was to implement a generalized protein design protocol that could be used to redesign any system. For example, one such protocol was to run the K* algorithm (see Section 1.2.2) over sequences I generated by mutating all pairs of residues within an enzyme's active site while setting all residues within 4 Å of a mutable residue to be continuously flexible. While this often resulted in a large conformation space and consequently design runs which would not finish, I was relieved of making difficult choices about the design, choices that to make correctly I would have had to better understand the biochemistry and interaction mechanisms of the redesign target. Like a typical computer scientist, I developed a system that did the hard work for me.

While these protocols generated an enormous number of computational predictions, I soon discovered (by sharing them with my biochemist collaborator) that few were of practical use. For example, I suggested mutating the antimicrobial peptide thanatin's β-hairpin loop or disulfide bridge, both of which I later learned are destabilizing mutations[196]. At times, when asked for predictions that interested my collaborators, the best answer I could give was that the designs were enqueued for execution, waiting behind hundreds of other designs to run first. In other words, my protocol-based approach to redesign prioritized predictions that those who best understood the redesign target considered unimportant!

This experience taught me a few valuable lessons:

1. There is no substitute for biochemical knowledge about a redesign target. Grokking a target allows the designer to prioritize sequence modifications at locations already known to be important. In addition, knowledge of which residues are unimportant allows you to investigate them later (or skip them).
2. In a collaboration, it is useful to generate and share predictions quickly, even if they are imperfect. I consider such preliminary predictions "drafts". There are two main benefits of sharing draft predictions:
    a. In highly technical fields it is common for miscommunications to occur among people with different expertise. Sharing draft predictions early and

often reduces the probability the designer will waste time in pursuit of a misinterpreted design goal.

   b. Collaborators will likely have useful ideas for how to improve your designs, enabling the generation of fruitful results quicker.

3. Always assess the OSPREY-generated low-energy molecular ensembles upon completion of a computational prediction. The designer should be able to formulate a hypothesis with structural justification to explain numeric predictions.

There is also an important aspect of engineering in protein design, in the sense that the protein designer needs to know which metaphorical knobs to turn achieve a desired result. For example, an engineer recognizes that the choice we presented at the beginning of this section about choosing between a provable and slow or stochastic and fast design algorithm presented a false dichotomy. Provable algorithms execute quickly by, e.g., reducing the size of the sequence and conformation space, loosening the bounds of the prediction, or using a faster, yet still provable, algorithm.

Once the immediate need for predictions subsides, the protein designer might consider what kind of theoretical advances, such as by finding tractable but sufficiently accurate subproblems in the vein of BWM*[118], could attack provable algorithms' lower bounds. While achieving algorithmic and modeling breakthroughs is not guaranteed, and any hypothesis that asserts a modeling simplification provides sufficient accuracy would need to be validated against experimental data, such breakthroughs are extremely valuable. They might arise through observations that, e.g., bounds on subtrees of conformation space are more useful than bounds on an individual conformation (as in the case of MARK*)[37], or that for certain types of designs only a certain energy window is useful (as in FRIES/EWAK*)[24]. Inspiration might also be found from biochemical insights gained through collaborations.

4.2 Machine Learning's Impact on Protein Design

It is now clear that the era of machine learning (ML)-based approaches to protein structure determination[197,198], structure prediction[13,14,199–201], and design[201–209] has arrived, heralded by the shockingly good performance of DeepMind AlphaFold's protein structure prediction at CASP13[210]. Confirming that AlphaFold's performance was not a one-off fluke, DeepMind two years later released AlphaFold2[13,14], which was able to predict the 3D-structure of a protein from sequence alone to near experimental accuracy[211]. Not to be left behind, companies not traditionally involved in biomedical research, like Meta and Salesforce, leveraged their strengths in AI and large language models to develop applications for single-sequence protein structure prediction[200] and the generation of protein sequences with predictable functions[201]. The leaps in performance afforded by ML-based methods have left some wondering about the uncertain role physics-based methods for protein structure determination and design will play in the future[212].

My hypothesis is that ML-based protein design methods will continue to improve in capabilities and accuracy, therefore it will behoove the computer scientists and protein designers of tomorrow to understand these methods and be able to apply them to tasks which they were trained for. Yet, at least in the foreseeable future, there are certain protein design tasks for which physics-based methods will likely retain the upper hand. To perform well, ML models need large amounts of data on which to train. In cases where available training data is sparse, such as empirical models of protein complexes, even state-of-the-art ML models such as AlphaFold-Multimer perform noticeably poorer than their single-chain versions[213]. Whereas our lab has previously shown[24,35] that the K* algorithm in OSPREY can predict the effect of variants on protein binding with a Spearman (ranking) correlation coefficient of > 0.75 (and also provides the PAStE algorithm[214] designed for this very task), a recent study by Pak et al.[215] examining whether AlphaFold could predict the effect a single point mutation has on a protein's stability ($\Delta\Delta G$) found weak to no correlation. This is unsurprising, as the AlphaFold FAQ itself disclaims the ability to predict the effect of mutations[216]. Pak et al. conclude that, due to the relative scarcity of experimental $\Delta\Delta G$ values, it is unlikely a ML-based $\Delta\Delta G$ predictor will be able to obtain a predictive accuracy that outperforms a much simpler, template-based approach[215,217].

Another design task for which there exists almost no empirical data (and relevant to Chapter 3) is structures of D-peptides bound to L-proteins. Thus, for the same reasons as described in the previous paragraph, it is unlikely that an ML-based approach will obsolete the physics-based DexDesign algorithm soon. On the other hand, if large amounts of D-peptide:L-protein experimental data is one day made available, I would expect that ML-based approaches could be successfully applied to *de novo* D-peptide binder design.

4.3 Science and Software

The contemporary research process is increasingly reliant on computing (i.e., software) to make inferences and predictions. Yet attention and allocation of funding has not yet caught up to software's growing importance in research[218]. Writing software that is 1) legible, 2) correct, 3) maintainable, and 4) reusable is difficult, and the skills required to achieve these are different than those typically honed for research. In academia, there is scant reward (such as increased funding) for writing good software, and consequently software is often treated as an afterthought, or something to do only to produce results for publications. Yet well-written software pays dividends by accelerating future research and enabling scientific reproducibility.

Well-written software can accelerate future research because it is reusable, allowing its incorporation into future applications. Reusability is often attained by having a sufficiently modular piece of code (i.e., mostly self-contained) at the right level of abstraction (i.e., the provided interface and functions are only those needed to accomplish the code's purpose). The right level of abstraction also has the advantageous quality of permitting future users of the code to not have to understand the details of how a module is implemented, which facilitates adoption and reuse. Reuse allows the future scientist to focus on the novel aspects of their research.

Legibility is important, as software written to be read by others facilitates third-party review and verification. This code review process, much like the scientific peer-review process, makes the resulting code more likely to be correct. Industry learned this lesson long ago, where code peer review is common practice. Factors that contribute to code legibility include descriptive function and variable names, standard code formatting, adequate use of white space, standard use of library functions, and others, as reviewed in Martin (2008)[219]. Legibility also facilitates maintainability by making it easier for others to contribute to the software.

Legibility, correctness, maintainability, and reusability contribute to higher quality of software and, importantly, enable scientific reproducibility. Software that is easy to use will be used, allowing reproduction of previous results, whereas few will use software that is difficult and indecipherable. As researchers, we should want others to validate our results, which means that we need to provide them with software they can use easily.

Unfortunately, there is no shortcut to learning to write good software. Much like learning to write in general, one must read much and write more. One method is to collaborate on a software project with more experienced engineers and scientists who are known to write quality software. It will take time, but it is worth it.

Appendix A: More Details on Applying Resistor to Predict Resistance Mutations in EGFR and BRAF

This appendix includes additional information and data relevant to our presentation of the Resistor algorithm in Chapter 2. It is adapted from the following publication:

> Guerin, N., Feichtner, A., Stefan, E., Kaserer, T. & Donald, B. R. Resistor: an algorithm for predicting resistance mutations via Pareto optimization over multistate protein design and mutational signatures. *Cell Systems* **13,** 830-843.e3 (2022).

A.1 Preparation of empirical and docked structures for K predictions*

The crystal structures used for the EGFR predictions were adopted from Kaserer and Blagg[28]. A full description of the PDB entries used can be found in that article's Table S7, and details on how the structures were prepared for OSPREY predictions is in that article's section "Structure Selection and Preparation."

For BRAF, the crystal structures of vemurafenib (PDB ID 3og7)[99] and dabrafenib (PDB ID 4xv2)[98] in complex with BRAF V600E were selected as input for Resistor. Both structures have been prepared using the default setting of the Protein Preparation Wizard[220] in Maestro[221]. In the case of encorafenib and PLX8394, crystal structures of structurally closely related, but not the identical, molecules were available. Teresa used those experimental complexes to generate encorafenib and PLX8394 models.

Encorafenib was docked into PDB ID 4xv3[98] using the default settings of the induced fit docking procedure in Maestro[221–224]. For validation, the co-crystallized ligand PLX7922 was re-docked. The highest scored docking pose of encorafenib was selected for further investigation. We found that the conserved substructures in encorafenib and PLX7922 aligned very well in this docking pose.

For PLX8394, re-docking of the co-crystallized ligand PLX7904 (PDB ID 4xv1)[98] failed with the induced fit docking procedure but was successful using a rigid docking workflow in GOLD version 5.8.0[225]. The binding site was defined as 6 Å around the ligand and the water molecule HOH905 was set to toggle and spin. The default settings of all other parameters were used.

An experimental structure of the endogenous ligand ADP was available, however, BRAF adopted in inactive conformation in this complex. Apo BRAF in its active conformation (PDB ID 4mne)[226] was thus combined with ANP-bound protein kinase c-src (PDB ID 2src)[227] to generate an active, endogenous ligand-bound BRAF complex. Teresa used this model as a template to build a BRAF:ADP homology model in the Molecular Operating Environment[228] using the default settings. This included refinement steps to resolve potential steric clashes in the rather crude ANP-BRAF input template.

As we note in these preceding paragraphs, in each case the BRAF structure we modeled was in its active conformation. There are some mutations, such as V600E, that are activating mutations and shift BRAF's conformational probability distribution to the active state[60,61]. With use of Resistor for mutational scanning of single point mutations within the active site, we assumed that the mutation is either itself activating or is a secondary mutation following an activating mutation, such as V600E. In our discussion of Resistor predictions of the BRAF double mutants V600E/T529M and V600E/T529I in Section 2.2.10, our assumption was that the V600E mutation is the activating mutation (which the existing drugs are effective against) and T529M/I are the secondary, resistance-causing mutations.

For all complexes, water molecules not involved in mediating interactions between the ligand and the target were deleted and only residues with a 12 Å radius around the ligand were kept in the final input structures.

A.2 Evaluation of ligand affinity

The command line interface of OSPREY was used to generate distinct YAML design files for each residue within 5 Å of a ligand. These YAML design files specify the input structures, the mutable residues, the flexible residues, and connectivity templates for OSPREY. To create the forcefield parameters files for the inhibitors and endogenous ligands, we used the Antechamber program in the AmberTools software package[229]. Then, to calculate the K* scores we used OSPREY with the following command input:

```
> osprey affinity --design YAML-design-file \
    --epsilon 0.63 --frcmod force-field-modification-file \
    --stability-threshold -1
```

where *YAML-design-file* was replaced with the individual YAML design file and *force-field-modification-file* was replaced with the AmberTools-generated file. The YAML design and forcefield modification files used in this study are available in the Harvard Dataverse.

A.3 Luciferase PCA analyses

Our collaborators Stefan Eduard and Andreas Feichtner, for the purpose of testing my computational predictions[30], transiently overexpressed indicated versions of the Rluc-PCA–based KinCon biosensors in 24-well plate formats. Experiments were performed 48 hours post transfection. For the luciferase-PCA measurements, the growth medium was carefully removed and the cells were washed with phosphate-buffered saline (PBS). Cell suspensions were transferred to 96-well plates and subjected to luminescence analysis using the PHERAstar FSX (BMG Labtech). Luciferase luminescence signals were integrated for 10 seconds following addition of the Rluc substrate benzyl-coelenterazine (NanoLight, #301). Cell lysates were prepared post RLU measurements. Expression levels of the biosensor were determined via western blot analysis.

A.4 Empirical Resistor runtimes

The Resistor computation entails three stages: 1) computing the positive and negative K* designs; 2) assigning mutational signature probabilities to each mutation, and; 3) running Pareto optimization over the four axes. Steps 2 and 3 empirically take a negligible amount of time (on the order of seconds). Step 1, however, computes two partition functions for each sequence and can take more time. Figure 15 shows the empirical runtime (in seconds) that it took our computers to run the positive and negative K* designs, where a design mutated a residue to each of the 19 other possible amino acids.

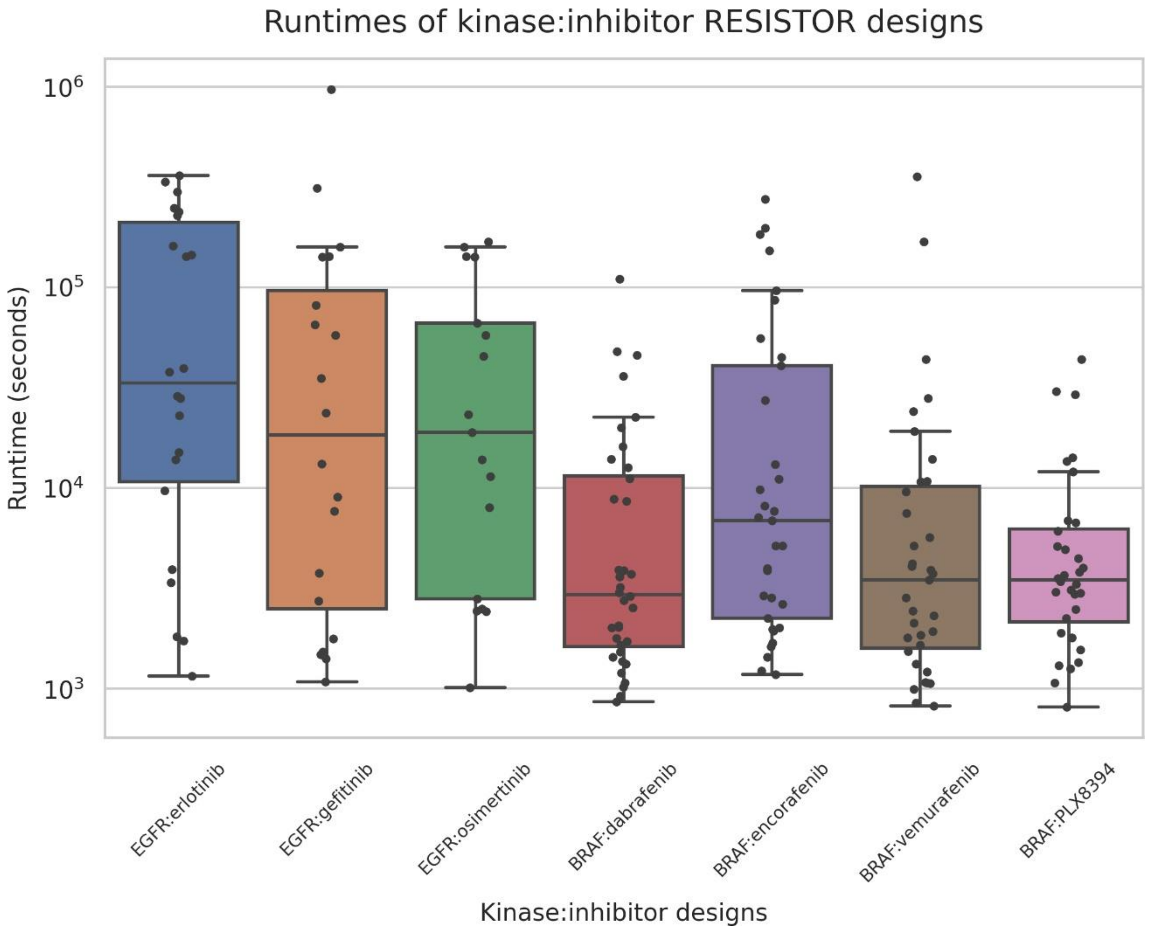


Figure 15: Positive and negative design runtimes. Box-and-whisker plot showing the minimum, maximum, median, first quartile, and third quartile runtimes per inhibitor:kinase pair. The whiskers extend to points that lie within 1.5 times the interquartile range. Each dot represents the number of seconds that Resistor took to compute the positive and negative K* designs for a given mutation location in a kinase:inhibitor complex. In other words, each dot represents the computation of 40 K* scores. The computation times across all the inhibitors range from 813 to 972,465 s, with the average being 40,630 or 1,015 s per sequence. The designs were run on a 24-core, 48-thread Intel Xeon processor with 4 Nvidia Titan V GPUs.

A.5 Quantification and statistical analysis

In Figure 4, the student's T-test was used to evaluate whether the mean of the RLU of a mutant was significantly different from that of the relative DMSO control. The SEM was used with n = 4. Significance was defined to three different p-levels, where $*p < 0.05$, $**p < 0.01$, and $***p < 0.001$.

To compute the specificity and sensitivity values reported in Section 2.3, we used the dataset from Table S1 from from Wagenaar et al[62]. We then reduced this set to those mutants for which Resistor made a prediction (Resistor made predictions for sequences with a mutated amino acid within 5 Å of the inhibitor or endogenous ligand). If Resistor predicted that a mutation caused resistance and Wagenaar et al. indicated that the mutant increased normalized drug enrichment, then that was considered a true positive. If Resistor predicted that a mutation was benign and Wagenaar et al. did not find increased drug enrichment, then that was considered a true negative. The specificity and sensitivity values were computed using their standard formulas.

A.6 Predictive contributions of individual parameters

Resistor optimizes four criteria, K* positive design, K* negative design, mutational probability, and hotspot cardinality. To further investigate the contributions of each criterion, we ran a computational ablation study on EGFR resistance to osimertinib. We first describe the contributions of the structure- and sequence-based criteria to pruning the mutational sequence space. We then describe how we omitted each of the four objectives, one-at-a-time, and recomputed each mutant's Pareto rank using the three remaining objectives. To disambiguate between Resistor with four criteria and the results of this computational ablation study, we denote the latter as *3-RANK*. We compared Resistor and 3-RANK's results to see whether using 3-RANK could improve the predictions.

Note that 3-RANK is, by nature, an imperfect comparison to Resistor because an ablation study should remove each criterion from both the pruning and the ranking steps.

As described above, Resistor consists of two components, *viz.*, a pruning and a Pareto-ranking step. We considered an ablation study where each of the criteria is removed from both the pruning and ranking steps. However, by applying the cut-off c (described in Section 2.2.5), we combine positive and negative design in a nonlinear fashion, and it is therefore not possible to remove them individually. In addition, the residue hotspot cardinality is a result of the pruning and can thus not be ablated from the pruning. To account for these limitations, we describe below first the magnitude of the effect of each of the three pruning criteria (the ratio of positive to negative design, loss of wildtype clonal fitness, and mutational probability) on the pruning step, and then describe how each of the criteria, when ablated solely from the ranking step, affects the ranking step (assuming full Resistor pruning has already occurred).

There were two structure-based pruning criteria, namely the resistance cut-off and loss of wildtype clonal fitness via ablated endogenous ligand binding as described in Section 2.2.5. These two criteria prune the largest proportion of sequences, accounting for the removal of 86.24 ± 3.20% of mutants in the EGFR and BRAF case studies. Removing all mutations with a mutational signature-derived probability of zero removes an additional 3.5 ± 0.87% of mutants. However, we have applied consecutive filtering steps here, and if we look at each parameter in this particular EGFR dataset individually, the cut-off c alone prunes 51.03 ± 5.97% of mutants, ablated endogenous ligand binding prunes 37.20 ± 6.18% of mutants, and mutations with a mutational probability of zero alone prunes 19.21 ± 0.70% of mutants. This shows that the structure-based criteria prune the overwhelming majority of the candidate sequences. In this particular

EGFR:osimertinib case, post pruning there remain 36 out of the original 357 candidate sequences that we then ranked using Pareto optimization.

In Table 1, we list Resistor's correct predictions for EGFR. There are five correct predictions for resistance mutations in EGFR when treated with osimertinib: L792H, G796R, G796D, G796C, and G796S. Resistor predicted L792H to be in the second Pareto rank and the four mutations at position 796 to be on the Pareto frontier. In comparing the Resistor ranks to the 3-RANK results, we found that omitting an objective with 3-RANK never improved the predictive accuracy (see Table 3 for the full results).

Being clinically confirmed resistance mutations, ideally 5 of the mutants should be on the Pareto fronter, i.e., have a Pareto rank of 1. 3-RANK that omits the K* score of the EGFR:wildtype ligand complex changed G796C's Pareto rank from 1 to 2, and increased the ranks to 3 and 4 respectively of the HIE and HIP protonation states of L792H. 3-RANK that omits the K* score of the EGFR:osimertinib complex reduced G796D's Pareto rank from 1 to 2, and reduced the HIP, HIE, and HID 792H protonation states from a Pareto rank of 2 to 5, 4, and 3, respectively. The largest reduction in rank accuracy is when 3-RANK omits mutational probabilities. In this case, L796R, L796D, L796C, and L796S have their Pareto ranks reduced from 1 to 2, 5, 6, and 8. The L792H mutation, in all protonation states, has its Pareto rank reduced from 2 to 9. On the other hand, when 3-RANK omits hotspot cardinality the Pareto ranks of the clinically confirmed resistance mutants remain the same. This is not too surprising, as hotspot cardinality is a count of the number of amino acids at a particular location that are predicted by K* positive and negative design, as well mutational probability, to confer resistance. It is thus dependent on the other three criteria, and in essence boosts locations that are predicted to be critical for drug or endogenous ligand binding. This indicates that in the future it might be possible to omit hotspot cardinality with only a minor drop in predictive accuracy.

In summary, positive design, negative design, and mutational probability all affect the pruning step, with the structural components most aggressively pruning the candidate resistance mutations. In the ranking step, the omission of positive design, negative design, or mutational probability in the Pareto optimization all negatively impact the accuracy of the results. Hotspot cardinality has a smaller effect on the predicted rankings than the other three criteria.

Table 3: Pareto ranks of computational Pareto objective ablation study on EGFR:osimertinib. Every mutation in this table is predicted to confer resistance. “Pos” is the position of the residue. “WT AA” is the wildtype identity of the amino acid. “Mut AA” is the resistance mutation. “Rank” is the Resistor-computed Pareto rank. “w/o (+) Design” is the Pareto rank of the mutation when the K* score of the wildtype ligand (ATP) is omitted from the Pareto optimization. “w/o (-) Design” is the Pareto rank of the mutation when the K* score of the drug (osimertinib) is omitted from the Pareto optimization. “w/o Probs” is the Pareto rank of the mutation when the mutational probability is omitted from the Pareto optimization. “w/o Count” is the Pareto rank of the mutation when the hotspot cardinality is omitted from the Pareto optimization.

Pos	WT AA	Mut AA	Rank	w/o (+) Design	w/o (-) Design	w/o Probs	w/o Count
718	leu	TRP	1	2	6	1	2
718	leu	PHE	1	1	6	2	1
718	leu	HIP	1	1	5	3	1
718	leu	HIE	1	2	3	3	1
718	leu	MET	1	1	1	5	1
719	gly	VAL	1	1	1	7	1
726	val	TRP	1	2	2	1	1
743	ala	ASP	1	1	1	6	1
796	gly	TRP	1	1	5	1	2
796	gly	TYR	1	1	2	1	2
796	gly	PHE	1	1	2	2	1
796	gly	LEU	1	2	1	1	1
796	gly	ARG	1	1	1	2	1
796	gly	ASP	1	1	2	5	1
796	gly	CYS	1	2	1	6	1
796	gly	SER	1	1	1	8	1
718	leu	HID	2	3	4	4	2
718	leu	ARG	2	2	2	5	2
723	phe	ILE	2	5	2	3	2
792	leu	HIP	2	2	5	9	2
792	leu	HIE	2	3	4	9	2
792	leu	HID	2	4	3	9	2
796	gly	GLU	2	2	3	3	3
796	gly	HIE	2	3	2	2	2
796	gly	ASN	2	3	3	5	2
718	leu	LYS	3	4	3	8	3
719	gly	THR	3	5	6	6	4
726	val	ARG	3	5	6	6	5
726	val	LYS	3	5	7	7	4
743	ala	GLU	3	4	7	5	3
796	gly	HID	3	4	3	3	3
796	gly	THR	3	4	4	7	3
844	leu	TRP	3	6	6	7	3
796	gly	HIP	4	5	4	4	4
718	leu	GLY	5	6	6	8	5
743	ala	ARG	5	6	8	5	5

A.7 All Resistor-predicted resistance mutations

Table 4: All Resistor resistance mutation predictions for EGFR with erlotinib. “Pos” is the position of the residue. “WT AA” is the wildtype identity of the amino acid. “Mut AA” is the resistance mutation. “Sig Prob” is the mutational signature probability for the mutation from “WT AA” to ”Mut AA” in lung adenocarcinoma. “ATP WT"” and “ATP Mut” are the K* scores of the endogenous ligand with the wildtype and mutant residues, respectively. “Drug WT” and “Drug Mut” are the K* scores of erlotinib with the wildtype and mutant residues, respectively. “Count” is the number of resistance mutations at the position. “Rank” is the Pareto rank of the mutation. Note: K* scores are in $\log_{10}$ units where possible and 0 where there is predicted to be no binding.

Pos	WT AA	Mut AA	Sig Prob	ATP WT	ATP Mut	Drug WT	Drug Mut	Count	Rank
718	LEU	PHE	0.000247	19.05	17.16	25.26	0	8	1
718	LEU	HIP	0.00042	19.05	18.86	25.26	-62.44	8	1
718	LEU	HIE	0.00042	19.05	18.92	25.26	-61.9	8	1
723	PHE	VAL	0.000316	19.06	19.14	25.2	0	5	1
723	PHE	LEU	0.00827	19.06	19.05	25.2	0	5	1
726	VAL	PHE	0.000509	19.04	19.71	25.24	0	2	1
743	ALA	ASP	0.0109	19.14	13.51	25.22	0	4	1
790	THR	LYS	0.00738	19.14	19.54	25.17	22.01	4	1
790	THR	MET	0.00602	19.14	19.79	25.17	23.89	4	1
791	GLN	PRO	0.0023	19.12	19.22	25.19	0	3	1
791	GLN	LYS	0.0163	19.12	19.1	25.19	0	3	1
796	GLY	TRP	2.06E-05	18.99	19.13	25.42	0	12	1
796	GLY	LEU	4.41E-05	18.99	19.55	25.42	-25.46	12	1
796	GLY	GLU	0.000154	18.99	18.88	25.42	1.16	12	1
796	GLY	PHE	0.000176	18.99	19.48	25.42	4.68	12	1
796	GLY	ARG	0.00286	18.99	19.54	25.42	9.36	12	1
796	GLY	ASP	0.00532	18.99	19.15	25.42	18.29	12	1
796	GLY	CYS	0.00384	18.99	19.28	25.42	21.71	12	1
796	GLY	SER	0.00643	18.99	19.23	25.42	22.24	12	1
718	LEU	GLY	8.49E-06	19.05	18.15	25.26	0	8	2
718	LEU	TRP	1.75E-05	19.05	17.96	25.26	0	8	2
718	LEU	HID	0.00042	19.05	18.9	25.26	-60.92	8	2
718	LEU	ARG	0.00238	19.05	19.41	25.26	22.5	8	2
726	VAL	TRP	4.82E-05	19.04	19.51	25.24	0	2	2
745	LYS	ILE	0.000243	18.98	19.05	25.18	0	5	2
745	LYS	MET	0.00516	18.98	18.97	25.18	0	5	2

790	THR	ARG	0.00139	19.14	19.32	25.17	11.4	4	2
791	GLN	GLY	1.81E-05	19.12	19.06	25.19	0	3	2
796	GLY	TYR	4.34E-05	18.99	19.48	25.42	-13.32	12	2
796	GLY	ASN	5.25E-05	18.99	19.36	25.42	21	12	2
796	GLY	HIE	1.88E-05	18.99	19.55	25.42	23.68	12	2
800	ASP	GLY	0.00153	19.06	19.13	25.21	0	1	2
718	LEU	LYS	0.00027	19.05	19.22	25.26	22.98	8	3
723	PHE	ASP	8.41E-07	19.06	18.98	25.2	0	5	3
745	LYS	HIE	6.76E-05	18.98	18.85	25.18	0	5	3
745	LYS	THR	0.00126	18.98	18.71	25.18	0	5	3
790	THR	ASN	0.000219	19.14	19.16	25.17	21.87	4	3
793	MET	ASN	0.000104	19.04	18.93	25.16	0	1	3
796	GLY	THR	3.45E-05	18.99	19.28	25.42	16.88	12	3
844	LEU	TRP	1.90E-05	18.99	19.02	25.45	-17.34	4	3
844	LEU	HID	0.00042	18.99	18.8	25.45	22.65	4	3
844	LEU	HIE	0.00042	18.99	18.74	25.45	22.63	4	3
854	THR	ASN	0.000262	19.01	19.09	25.43	21.08	1	3
723	PHE	ALA	5.72E-07	19.06	18.98	25.2	0	5	4
723	PHE	GLY	5.92E-07	19.06	18.91	25.2	0	5	4
743	ALA	CYS	7.34E-05	19.14	14.53	25.22	0	4	4
743	ALA	GLU	7.74E-05	19.14	4.17	25.22	0	4	4
745	LYS	HID	6.76E-05	18.98	18.81	25.18	0	5	4
844	LEU	HIP	0.00042	18.99	18.57	25.45	22.42	4	4
743	ALA	ARG	1.73E-05	19.14	6.55	25.22	0	4	5

Table 5: All Resistor resistance mutation predictions for EGFR with gefitinib. "Pos" is the position of the residue. "WT AA" is the wildtype identity of the amino acid. "Mut AA" is the resistance mutation. "Sig Prob" is the mutational signature probability for the mutation from "WT AA" to "Mut AA" in lung adenocarcinoma. "ATP WT'" and "ATP Mut" are the K* scores of the endogenous ligand with the wildtype and mutant residues, respectively. "Drug WT" and "Drug Mut" are the K* scores of gefitinib with the wildtype and mutant residues, respectively. "Count" is the number of resistance mutations at the position. "Rank" is the Pareto rank of the mutation. Note: K* scores are in $\log_{10}$ units where possible and 0 where there is predicted to be no binding.

Pos	WT AA	Mut AA	Sig Prob	ATP WT	ATP Mut	Drug WT	Drug Mut	Count	Rank
718	LEU	PHE	0.000247	19.05	17.16	26.94	-23.15	7	1
718	LEU	TRP	1.75E-05	19.05	17.96	26.94	1.76	7	1
718	LEU	HIP	0.00042	19.05	18.86	26.94	4.5	7	1
718	LEU	HID	0.00042	19.05	18.9	26.94	4.92	7	1
718	LEU	HIE	0.00042	19.05	18.92	26.94	4.99	7	1
718	LEU	ARG	0.00238	19.05	19.41	26.94	23.69	7	1
743	ALA	GLU	7.74E-05	19.14	4.17	26.93	-49.27	2	1
777	LEU	HID	0.000281	19.05	19.04	26.86	0	1	1
790	THR	ARG	0.00139	19.14	19.32	26.95	12.73	4	1
790	THR	LYS	0.00738	19.14	19.54	26.95	23.21	4	1
790	THR	MET	0.00602	19.14	19.79	26.95	25.06	4	1
796	GLY	LEU	4.41E-05	18.99	19.55	26.88	22.57	1	1
844	LEU	TRP	1.90E-05	18.99	19.02	26.97	-28.99	4	1
718	LEU	LYS	0.00027	19.05	19.22	26.94	24.45	7	2
726	VAL	PHE	0.000509	19.04	19.71	26.94	25.26	1	2
743	ALA	ARG	1.73E-05	19.14	6.55	26.93	-4.31	2	2
745	LYS	ILE	0.000243	18.98	19.05	26.88	16.99	1	2
790	THR	ASN	0.000219	19.14	19.16	26.95	23.52	4	2
844	LEU	HIP	0.00042	18.99	18.57	26.97	23.69	4	2
844	LEU	HID	0.00042	18.99	18.8	26.97	23.97	4	2
844	LEU	HIE	0.00042	18.99	18.74	26.97	23.94	4	2
854	THR	ASN	0.000262	19.01	19.09	26.98	20.61	1	2

Table 6: All Resistor resistance mutation predictions for EGFR with osimertinib. “Pos” is the position of the residue. “WT AA” is the wildtype identity of the amino acid. “Mut AA” is the resistance mutation. “Sig Prob” is the mutational signature probability for the mutation from “WT AA” to ”Mut AA” in lung adenocarcinoma. “ATP WT”” and “ATP Mut” are the K* scores of the endogenous ligand with the wildtype and mutant residues, respectively. “Drug WT” and “Drug Mut” are the K* scores of osimertinib with the wildtype and mutant residues, respectively. “Count” is the number of resistance mutations at the position. “Rank” is the Pareto rank of the mutation. Note: K* scores are in log$_{10}$ units where possible and 0 where there is predicted to be no binding.

Pos	WT AA	Mut AA	Sig Prob	ATP WT	ATP Mut	Drug WT	Drug Mut	Count	Rank
718	LEU	TRP	1.75E-05	19.05	17.96	27.51	0	9	1
718	LEU	PHE	0.000247	19.05	17.16	27.51	0	9	1
718	LEU	HIP	0.00042	19.05	18.86	27.51	-38.77	9	1
718	LEU	HIE	0.00042	19.05	18.92	27.51	-38.57	9	1
718	LEU	MET	0.0108	19.05	19.44	27.51	25.52	9	1
719	GLY	VAL	0.017	19.05	14.7	27.5	20.75	2	1
726	VAL	TRP	4.82E-05	19.04	19.51	27.48	0	3	1
743	ALA	ASP	0.0109	19.14	13.51	27.43	17.5	3	1
796	GLY	TRP	2.06E-05	18.99	19.13	27.38	-97.39	14	1
796	GLY	TYR	4.34E-05	18.99	19.48	27.38	-61.59	14	1
796	GLY	PHE	0.000176	18.99	19.48	27.38	-41.78	14	1
796	GLY	LEU	4.41E-05	18.99	19.55	27.38	-3.75	14	1
796	GLY	ARG	0.00286	18.99	19.54	27.38	10.61	14	1
796	GLY	ASP	0.00532	18.99	19.15	27.38	14.96	14	1
796	GLY	CYS	0.00384	18.99	19.28	27.38	21.85	14	1
796	GLY	SER	0.00643	18.99	19.23	27.38	24.76	14	1
718	LEU	HID	0.00042	19.05	18.9	27.51	-37.61	9	2
718	LEU	ARG	0.00238	19.05	19.41	27.51	20.16	9	2
723	PHE	ILE	0.00209	19.06	19.5	27.45	25.68	1	2
792	LEU	HIP	0.00486	19.03	18.88	27.47	24.98	3	2
792	LEU	HIE	0.00486	19.03	18.93	27.47	25.07	3	2
792	LEU	HID	0.00486	19.03	18.98	27.47	25.15	3	2
796	GLY	GLU	0.000154	18.99	18.88	27.38	2.49	14	2
796	GLY	HIE	1.88E-05	18.99	19.55	27.38	11.32	14	2
796	GLY	ASN	5.25E-05	18.99	19.36	27.38	18.38	14	2
718	LEU	LYS	0.00027	19.05	19.22	27.51	24.7	9	3
719	GLY	THR	4.02E-05	19.05	17.76	27.5	20.4	2	3
726	VAL	ARG	2.32E-05	19.04	17.68	27.48	21.62	3	3
726	VAL	LYS	5.39E-05	19.04	17.07	27.48	21.87	3	3

743	ALA	GLU	7.74E-05	19.14	4.17	27.43	0.47	3	3
796	GLY	HID	1.88E-05	18.99	19.49	27.38	11.69	14	3
796	GLY	THR	3.45E-05	18.99	19.28	27.38	22.43	14	3
844	LEU	TRP	1.90E-05	18.99	19.02	27.56	22.12	1	3
796	GLY	HIP	1.88E-05	18.99	19.46	27.38	12.09	14	4
718	LEU	GLY	8.49E-06	19.05	18.15	27.51	24.02	9	5
743	ALA	ARG	1.73E-05	19.14	6.55	27.43	12.58	3	5

Table 7: All Resistor resistance mutation predictions for BRAF with dabrafenib. “Pos” is the position of the residue. “WT AA” is the wildtype identity of the amino acid. “Mut AA” is the resistance mutation. “Sig Prob” is the mutational signature probability for the mutation from “WT AA” to ”Mut AA” in lung adenocarcinoma. “ATP WT’” and “ATP Mut” are the K* scores of the endogenous ligand with the wildtype and mutant residues, respectively. “Drug WT” and “Drug Mut” are the K* scores of dabrafenib with the wildtype and mutant residues, respectively. “Count” is the number of resistance mutations at the position. “Rank” is the Pareto rank of the mutation. Note: K* scores are in log$_{10}$ units where possible and 0 where there is predicted to be no binding.

Pos	WT AA	Mut AA	Sig Prob	ATP WT	ATP Mut	Drug WT	Drug Mut	Count	Rank
466	GLY	ARG	5.84E-02	18.8	10.53	37.16	-167.92	11	1
466	GLY	LYS	2.38E-02	18.8	11.79	37.16	-52.4	11	1
466	GLY	GLU	2.19E-01	18.8	12.89	37.16	21.32	11	1
471	VAL	LEU	4.43E-04	18.65	19.67	37.26	25.1	6	1
508	THR	ARG	2.95E-04	18.59	18.59	37.23	-118.81	4	1
535	SER	PRO	1.30E-03	18.72	18.65	37.26	0	1	1
593	GLY	PHE	1.99E-06	18.66	20.07	37.17	0	16	1
593	GLY	TYR	3.58E-05	18.66	19.86	37.17	0	16	1
593	GLY	ARG	7.80E-04	18.66	16.17	37.17	0	16	1
593	GLY	GLU	2.76E-04	18.66	18.73	37.17	-60.32	16	1
593	GLY	ASN	1.34E-03	18.66	19.16	37.17	-39.82	16	1
593	GLY	ASP	1.63E-02	18.66	18.89	37.17	-29.79	16	1
593	GLY	CYS	1.66E-03	18.66	19.06	37.17	17.46	16	1
593	GLY	VAL	9.35E-04	18.66	19.18	37.17	28.45	16	1
593	GLY	ILE	4.55E-05	18.66	19.8	37.17	30.24	16	1
593	GLY	SER	6.09E-02	18.66	18.83	37.17	34.27	16	1
466	GLY	GLN	7.24E-05	18.8	12.55	37.16	11.37	11	2
466	GLY	ASP	7.51E-04	18.8	17.06	37.16	18.64	11	2
466	GLY	VAL	2.51E-03	18.8	13.44	37.16	29.23	11	2
467	SER	PRO	7.37E-04	18.62	18.85	37.15	30.42	1	2
481	ALA	LYS	2.74E-04	18.58	17.32	36.98	-4.22	8	2
481	ALA	LEU	7.06E-05	18.58	18.68	36.98	9.97	8	2
481	ALA	GLU	1.21E-03	18.58	17.92	36.98	22.11	8	2
505	LEU	ARG	8.61E-04	18.59	18.58	36.85	16.53	5	2
508	THR	LYS	9.16E-04	18.59	18.59	37.23	27.22	4	2
514	LEU	ARG	5.22E-05	18.57	17.18	37.1	21.32	12	2
514	LEU	ILE	1.65E-03	18.57	18.4	37.1	32.55	12	2
529	THR	PHE	3.77E-05	18.58	15.91	36.99	-125.31	11	2
529	THR	MET	1.74E-05	18.58	18.65	36.99	-8.16	11	2

529	THR	ASN	9.96E-04	18.58	18.55	36.99	34.54	11	2
593	GLY	HIE	1.66E-05	18.66	19.58	37.17	0	16	2
593	GLY	THR	2.67E-05	18.66	19.06	37.17	27.33	16	2
464	GLY	GLN	7.24E-05	18.58	2.95	37.09	11.05	1	3
466	GLY	THR	5.47E-05	18.8	14.85	37.16	29.21	11	3
481	ALA	ILE	5.14E-05	18.58	14.2	36.98	21.38	8	3
481	ALA	VAL	1.46E-03	18.58	17.77	36.98	33.45	8	3
505	LEU	SER	2.78E-05	18.59	18.58	36.85	35.02	5	3
514	LEU	PRO	1.21E-03	18.57	18.12	37.1	33.34	12	3
527	ILE	LEU	6.37E-05	18.6	18.61	37.3	33.5	1	3
593	GLY	HIP	1.66E-05	18.66	19.56	37.17	0	16	3
514	LEU	SER	2.27E-04	18.57	18.1	37.1	34.19	12	4
593	GLY	HID	1.66E-05	18.66	19.55	37.17	0	16	4
471	VAL	MET	8.44E-06	18.65	18.96	37.26	27.83	6	5
481	ALA	ARG	1.53E-05	18.58	17.02	36.98	-15.13	8	5
481	ALA	ASP	4.39E-06	18.58	19.21	36.98	31.9	8	5
508	THR	GLU	1.05E-05	18.59	18.59	37.23	35	4	5
514	LEU	PHE	1.53E-05	18.57	18.19	37.1	0	12	5
578	LYS	TYR	5.24E-06	18.55	18.39	37.11	-143.76	1	5
593	GLY	TRP	4.78E-06	18.66	18.83	37.17	0	16	5
593	GLY	LEU	3.87E-07	18.66	19.35	37.17	-85.52	16	5
466	GLY	PRO	1.77E-07	18.8	18.27	37.16	0	11	6
466	GLY	TRP	2.10E-06	18.8	13.1	37.16	0	11	6
466	GLY	LEU	4.82E-06	18.8	9.74	37.16	-39.41	11	6
466	GLY	CYS	3.82E-06	18.8	17.62	37.16	27.45	11	6
469	GLY	PRO	1.77E-07	18.6	18.75	37.1	0	1	6
471	VAL	PRO	5.74E-07	18.65	17.88	37.26	0	6	6
471	VAL	ARG	3.03E-07	18.65	18.99	37.26	23.05	6	6
471	VAL	GLU	9.01E-07	18.65	18.82	37.26	31.61	6	6
481	ALA	GLN	1.80E-06	18.58	18.37	36.98	15.47	8	6
508	THR	GLN	2.51E-06	18.59	18.59	37.23	33.59	4	6
513	ILE	ARG	1.06E-06	18.57	18.57	37.09	-30.17	2	6
513	ILE	TYR	2.73E-06	18.57	18.57	37.09	27.86	2	6
514	LEU	HIP	3.37E-07	18.57	17.68	37.1	25.98	12	6
514	LEU	HID	3.37E-07	18.57	17.72	37.1	26.06	12	6

514	LEU	LYS	1.68E-06	18.57	17.37	37.1	32.25	12	6
514	LEU	MET	2.51E-06	18.57	18.42	37.1	33.68	12	6
528	VAL	ARG	5.57E-07	18.57	18.61	37.01	-72.34	1	6
529	THR	TYR	1.12E-06	18.58	18.58	36.99	-10.9	11	6
529	THR	ARG	4.98E-07	18.58	18.6	36.99	-4.95	11	6
529	THR	LYS	1.74E-06	18.58	18.57	36.99	4.7	11	6
529	THR	LEU	3.10E-06	18.58	18.56	36.99	26.71	11	6
532	CYS	HID	4.44E-06	18.49	14.39	37.09	26.54	7	6
532	CYS	HIP	4.44E-06	18.49	14.51	37.09	26.79	7	6
532	CYS	HIE	4.44E-06	18.49	12.48	37.09	25.2	7	6
532	CYS	ILE	1.16E-06	18.49	18.82	37.09	34.28	7	6
532	CYS	VAL	2.11E-06	18.49	18.7	37.09	34.77	7	6
471	VAL	HID	6.58E-07	18.65	17.61	37.26	33.79	6	7
505	LEU	GLY	6.90E-08	18.59	18.58	36.85	34.62	5	7
505	LEU	GLN	1.84E-06	18.59	18.6	36.85	34.82	5	7
514	LEU	HIE	3.37E-07	18.57	17.72	37.1	27.35	12	7
514	LEU	GLY	3.88E-08	18.57	18.03	37.1	33.68	12	7
514	LEU	ALA	9.24E-07	18.57	18.09	37.1	34.25	12	7
529	THR	HID	6.68E-08	18.58	18.58	36.99	-10.11	11	7
529	THR	HIE	6.68E-08	18.58	18.58	36.99	-4.75	11	7
529	THR	ASP	2.37E-06	18.58	18.51	36.99	34.7	11	7
531	TRP	PRO	5.44E-07	18.51	16.76	37.07	0	1	7
505	LEU	ALA	5.28E-07	18.59	18.58	36.85	35	5	8
529	THR	HIP	6.68E-08	18.58	18.58	36.99	-6.63	11	8

Table 8: All Resistor resistance mutation predictions for BRAF with vemurafenib. “Pos” is the position of the residue. “WT AA” is the wildtype identity of the amino acid. “Mut AA” is the resistance mutation. “Sig Prob” is the mutational signature probability for the mutation from “WT AA” to ”Mut AA” in lung adenocarcinoma. “ATP WT’” and “ATP Mut” are the K* scores of the endogenous ligand with the wildtype and mutant residues, respectively. “Drug WT” and “Drug Mut” are the K* scores of vemurafenib with the wildtype and mutant residues, respectively. “Count” is the number of resistance mutations at the position. “Rank” is the Pareto rank of the mutation. Note: K* scores are in $\log_{10}$ units where possible and 0 where there is predicted to be no binding.

Pos	WT AA	Mut AA	Sig Prob	ATP WT	ATP Mut	Drug WT	Drug Mut	Count	Rank
471	VAL	LEU	0.000443	18.65	19.67	33.41	29.39	4	1
481	ALA	THR	0.0177	18.58	18.99	33.27	30.69	9	1
529	THR	ILE	0.0202	18.58	18.57	33.45	29.33	10	1
535	SER	PRO	0.0013	18.72	18.65	33.65	0	1	1
593	GLY	PHE	1.99E-06	18.66	20.07	33.47	0	16	1
593	GLY	TYR	3.58E-05	18.66	19.86	33.47	0	16	1
593	GLY	ARG	0.00078	18.66	16.17	33.47	-231.93	16	1
593	GLY	ASN	0.00134	18.66	19.16	33.47	-103.95	16	1
593	GLY	ASP	0.0163	18.66	18.89	33.47	-26.78	16	1
593	GLY	CYS	0.00166	18.66	19.06	33.47	19.89	16	1
593	GLY	VAL	0.000935	18.66	19.18	33.47	21.09	16	1
593	GLY	ILE	4.55E-05	18.66	19.8	33.47	23.08	16	1
593	GLY	SER	0.0609	18.66	18.83	33.47	31.33	16	1
463	ILE	TYR	2.55E-05	18.6	15.94	33.42	-124.43	4	2
481	ALA	GLU	0.00121	18.58	17.92	33.27	11.12	9	2
481	ALA	VAL	0.00146	18.58	17.77	33.27	28.52	9	2
505	LEU	PHE	0.00544	18.59	18.58	33.45	27.01	3	2
505	LEU	ARG	0.000861	18.59	18.58	33.45	27.41	3	2
508	THR	ARG	0.000295	18.59	18.59	33.45	12.91	2	2
508	THR	LYS	0.000916	18.59	18.59	33.45	21.18	2	2
514	LEU	ILE	0.00165	18.57	18.4	33.43	30.08	11	2
532	CYS	ARG	0.000812	18.49	13.04	33.28	-10.66	9	2
593	GLY	HIE	1.66E-05	18.66	19.58	33.47	0	16	2
593	GLY	GLU	0.000276	18.66	18.73	33.47	-12.82	16	2
593	GLY	THR	2.67E-05	18.66	19.06	33.47	20.72	16	2
481	ALA	LEU	7.06E-05	18.58	18.68	33.27	18.19	9	3
481	ALA	LYS	0.000274	18.58	17.32	33.27	18.46	9	3
514	LEU	ARG	5.22E-05	18.57	17.18	33.43	22.28	11	3
514	LEU	GLN	8.77E-05	18.57	17.02	33.43	27.9	11	3

514	LEU	SER	0.000227	18.57	18.1	33.43	30.81	11	3
529	THR	MET	1.74E-05	18.58	18.65	33.45	-8.52	10	3
529	THR	PHE	3.77E-05	18.58	15.91	33.45	5.09	10	3
593	GLY	HIP	1.66E-05	18.66	19.56	33.47	0	16	3
481	ALA	ILE	5.14E-05	18.58	14.2	33.27	24.57	9	4
593	GLY	HID	1.66E-05	18.66	19.55	33.47	0	16	4
471	VAL	MET	8.44E-06	18.65	18.96	33.41	31.15	4	5
481	ALA	ARG	1.53E-05	18.58	17.02	33.27	19.55	9	5
481	ALA	ASP	4.39E-06	18.58	19.21	33.27	24.15	9	5
514	LEU	PHE	1.53E-05	18.57	18.19	33.43	0	11	5
593	GLY	TRP	4.78E-06	18.66	18.83	33.47	0	16	5
593	GLY	LEU	3.87E-07	18.66	19.35	33.47	-237.44	16	5
463	ILE	HIE	5.14E-06	18.6	18.06	33.42	30.11	4	6
463	ILE	HID	5.14E-06	18.6	18.08	33.42	30.75	4	6
466	GLY	PRO	1.77E-07	18.8	18.27	33.48	0	1	6
471	VAL	PRO	5.74E-07	18.65	17.88	33.41	0	4	6
471	VAL	GLU	9.01E-07	18.65	18.82	33.41	31.34	4	6
481	ALA	GLN	1.80E-06	18.58	18.37	33.27	-15.45	9	6
514	LEU	HIP	3.37E-07	18.57	17.68	33.43	25.66	11	6
514	LEU	HID	3.37E-07	18.57	17.72	33.43	25.78	11	6
514	LEU	GLY	3.88E-08	18.57	18.03	33.43	30.29	11	6
514	LEU	ALA	9.24E-07	18.57	18.09	33.43	30.82	11	6
516	PHE	ARG	4.47E-06	18.59	18.58	33.51	29.7	1	6
529	THR	TYR	1.12E-06	18.58	18.58	33.45	-114.94	10	6
529	THR	ARG	4.98E-07	18.58	18.6	33.45	-34.39	10	6
529	THR	LYS	1.74E-06	18.58	18.57	33.45	-7.92	10	6
529	THR	LEU	3.10E-06	18.58	18.56	33.45	28.04	10	6
532	CYS	HIP	4.44E-06	18.49	14.51	33.28	-0.94	9	6
532	CYS	HIE	4.44E-06	18.49	12.48	33.28	-2.8	9	6
532	CYS	ILE	1.16E-06	18.49	18.82	33.28	27.39	9	6
532	CYS	VAL	2.11E-06	18.49	18.7	33.28	29.58	9	6
463	ILE	HIP	5.14E-06	18.6	17.51	33.42	30.23	4	7
505	LEU	MET	3.58E-07	18.59	18.6	33.45	25.88	3	7
514	LEU	HIE	3.37E-07	18.57	17.72	33.43	25.91	11	7
529	THR	HIE	6.68E-08	18.58	18.58	33.45	10.11	10	7

531	TRP	PRO	5.44E-07	18.51	16.76	33.25	0	1	7	
532	CYS	HID	4.44E-06	18.49	14.39	33.28	-0.66	9	7	
532	CYS	THR	2.67E-07	18.49	18.32	33.28	30.91	9	7	
514	LEU	GLU	5.81E-08	18.57	17.36	33.43	28	11	8	
529	THR	HIP	6.68E-08	18.58	18.58	33.45	10.86	10	8	
529	THR	HID	6.68E-08	18.58	18.58	33.45	11.36	10	8	

Table 9: All Resistor resistance mutation predictions for BRAF with encorafenib. “Pos” is the position of the residue. “WT AA” is the wildtype identity of the amino acid. “Mut AA” is the resistance mutation. “Sig Prob” is the mutational signature probability for the mutation from “WT AA” to ”Mut AA” in lung adenocarcinoma. “ATP WT’” and “ATP Mut” are the K* scores of the endogenous ligand with the wildtype and mutant residues, respectively. “Drug WT” and “Drug Mut” are the K* scores of encorafenib with the wildtype and mutant residues, respectively. “Count” is the number of resistance mutations at the position. “Rank” is the Pareto rank of the mutation. Note: K* scores are in $\log_{10}$ units where possible and 0 where there is predicted to be no binding.

Pos	WT AA	Mut AA	Sig Prob	ATP WT	ATP Mut	Drug WT	Drug Mut	Count	Rank
471	VAL	LEU	0.000443	18.65	19.67	38.16	28.68	10	1
481	ALA	LEU	7.06E-05	18.58	18.68	38.13	-24.05	8	1
481	ALA	GLU	0.00121	18.58	17.92	38.13	10.6	8	1
529	THR	ILE	0.0202	18.58	18.57	38.12	31.23	12	1
532	CYS	ARG	0.000812	18.49	13.04	38.06	-19.78	9	1
535	SER	PRO	0.0013	18.72	18.65	38.18	0	1	1
593	GLY	TYR	3.58E-05	18.66	19.86	38.09	0	17	1
593	GLY	ARG	0.00078	18.66	16.17	38.09	-2.33	17	1
593	GLY	ILE	4.55E-05	18.66	19.8	38.09	2.23	17	1
593	GLY	VAL	0.000935	18.66	19.18	38.09	7.05	17	1
593	GLY	ASN	0.00134	18.66	19.16	38.09	28.19	17	1
593	GLY	ASP	0.0163	18.66	18.89	38.09	28.87	17	1
593	GLY	PHE	1.99E-06	18.66	20.07	38.09	30.05	17	1
593	GLY	CYS	0.00166	18.66	19.06	38.09	34.56	17	1
593	GLY	SER	0.0609	18.66	18.83	38.09	35.04	17	1
471	VAL	PHE	0.000785	18.65	16.37	38.16	26	10	2
481	ALA	LYS	0.000274	18.58	17.32	38.13	7.23	8	2
514	LEU	ARG	5.22E-05	18.57	17.18	38.14	20.73	10	2
529	THR	MET	1.74E-05	18.58	18.65	38.12	-18.87	12	2
529	THR	PHE	3.77E-05	18.58	15.91	38.12	0.72	12	2
529	THR	ASN	0.000996	18.58	18.55	38.12	33.77	12	2
536	SER	ASN	0.0137	18.63	18.53	38.02	34.35	3	2
583	PHE	TYR	0.00408	18.6	18.68	38.1	29.81	9	2
593	GLY	HIE	1.66E-05	18.66	19.58	38.09	0	17	2
593	GLY	GLU	0.000276	18.66	18.73	38.09	22.97	17	2
593	GLY	THR	2.67E-05	18.66	19.06	38.09	24.19	17	2
593	GLY	ALA	0.000254	18.66	18.8	38.09	35.61	17	2
463	ILE	TYR	2.55E-05	18.6	15.94	38.08	9.07	3	3
481	ALA	ILE	5.14E-05	18.58	14.2	38.13	30.63	8	3

481	ALA	VAL	0.00146	18.58	17.77	38.13	34.23	8	3
505	LEU	HIP	0.00146	18.59	18.59	38.08	35.98	2	3
514	LEU	GLN	8.77E-05	18.57	17.02	38.14	31.64	10	3
536	SER	ASP	5.03E-05	18.63	18.21	38.02	33.54	3	3
583	PHE	VAL	0.000214	18.6	17.05	38.1	33.2	9	3
583	PHE	ILE	0.00316	18.6	17.45	38.1	34.49	9	3
583	PHE	SER	0.00262	18.6	16.86	38.1	34.31	9	3
593	GLY	HIP	1.66E-05	18.66	19.56	38.09	0	17	3
505	LEU	HID	0.00146	18.59	18.59	38.08	36.08	2	4
593	GLY	HID	1.66E-05	18.66	19.55	38.09	0	17	4
471	VAL	PRO	5.74E-07	18.65	17.88	38.16	0	10	5
471	VAL	ARG	3.03E-07	18.65	18.99	38.16	18.89	10	5
471	VAL	MET	8.44E-06	18.65	18.96	38.16	21.35	10	5
481	ALA	GLN	1.80E-06	18.58	18.37	38.13	-7.07	8	5
481	ALA	ARG	1.53E-05	18.58	17.02	38.13	1.87	8	5
481	ALA	ASP	4.39E-06	18.58	19.21	38.13	30.87	8	5
514	LEU	PHE	1.53E-05	18.57	18.19	38.14	-25.54	10	5
529	THR	ARG	4.98E-07	18.58	18.6	38.12	-6.65	12	5
532	CYS	HIP	4.44E-06	18.49	14.51	38.06	-40.5	9	5
532	CYS	HIE	4.44E-06	18.49	12.48	38.06	-40.74	9	5
533	GLU	PRO	3.22E-07	18.58	18.63	38.12	0	1	5
593	GLY	LEU	3.87E-07	18.66	19.35	38.09	19.61	17	5
593	GLY	TRP	4.78E-06	18.66	18.83	38.09	19.1	17	5
463	ILE	HIE	5.14E-06	18.6	18.06	38.08	35.28	3	6
471	VAL	GLU	9.01E-07	18.65	18.82	38.16	35.77	10	6
529	THR	TYR	1.12E-06	18.58	18.58	38.12	19.14	12	6
529	THR	LYS	1.74E-06	18.58	18.57	38.12	20.27	12	6
529	THR	HIE	6.68E-08	18.58	18.58	38.12	21.51	12	6
529	THR	LEU	3.10E-06	18.58	18.56	38.12	22.42	12	6
531	TRP	PRO	5.44E-07	18.51	16.76	38.04	0	1	6
532	CYS	HID	4.44E-06	18.49	14.39	38.06	-40.35	9	6
532	CYS	ILE	1.16E-06	18.49	18.82	38.06	30.25	9	6
532	CYS	VAL	2.11E-06	18.49	18.7	38.06	30.4	9	6
583	PHE	ARG	5.91E-06	18.6	17.17	38.1	32.32	9	6
583	PHE	THR	1.13E-05	18.6	16.91	38.1	32.78	9	6

583	PHE	MET	5.94E-06	18.6	18.39	38.1	35.48	9	6
463	ILE	HID	5.14E-06	18.6	18.08	38.08	35.51	3	7
471	VAL	HIP	6.58E-07	18.65	17.42	38.16	30.52	10	7
471	VAL	HID	6.58E-07	18.65	17.61	38.16	31.08	10	7
471	VAL	TYR	1.33E-06	18.65	16.37	38.16	31.46	10	7
514	LEU	HIP	3.37E-07	18.57	17.68	38.14	31.73	10	7
514	LEU	HID	3.37E-07	18.57	17.72	38.14	31.87	10	7
514	LEU	LYS	1.68E-06	18.57	17.37	38.14	33.97	10	7
514	LEU	MET	2.51E-06	18.57	18.42	38.14	35.5	10	7
529	THR	HID	6.68E-08	18.58	18.58	38.12	20.75	12	7
529	THR	ASP	2.37E-06	18.58	18.51	38.12	34.9	12	7
536	SER	LEU	5.47E-07	18.63	17.44	38.02	30.59	3	7
583	PHE	TRP	9.45E-07	18.6	13.08	38.1	24.17	9	7
583	PHE	GLY	1.52E-06	18.6	16.63	38.1	33.81	9	7
471	VAL	HIE	6.58E-07	18.65	17.28	38.16	30.93	10	8
514	LEU	HIE	3.37E-07	18.57	17.72	38.14	32.04	10	8
529	THR	HIP	6.68E-08	18.58	18.58	38.12	20.97	12	8
532	CYS	THR	2.67E-07	18.49	18.32	38.06	35.12	9	8
514	LEU	GLU	5.81E-08	18.57	17.36	38.14	32.45	10	9
514	LEU	GLY	3.88E-08	18.57	18.03	38.14	35.47	10	9

Table 10: All Resistor resistance mutation predictions for BRAF with PLX8394. “Pos” is the position of the residue. “WT AA” is the wildtype identity of the amino acid. “Mut AA” is the resistance mutation. “Sig Prob” is the mutational signature probability for the mutation from “WT AA” to ”Mut AA” in lung adenocarcinoma. “ATP WT'” and “ATP Mut” are the K* scores of the endogenous ligand with the wildtype and mutant residues, respectively. “Drug WT” and “Drug Mut” are the K* scores of PLX8394 with the wildtype and mutant residues, respectively. “Count” is the number of resistance mutations at the position. “Rank” is the Pareto rank of the mutation. Note: K* scores are in $\log_{10}$ units where possible and 0 where there is predicted to be no binding.

Pos	WT AA	Mut AA	Sig Prob	ATP WT	ATP Mut	Drug WT	Drug Mut	Count	Rank
471	VAL	LEU	0.000443	18.645	19.67	31.624	29.38	3	1
513	ILE	PHE	0.00146	18.569	18.57	31.663	0	1	1
529	THR	ILE	0.0202	18.583	18.57	32.196	12.61	14	1
535	SER	PRO	0.0013	18.717	18.65	31.813	0	5	1
535	SER	LEU	0.00156	18.717	19.09	31.813	24.04	5	1
593	GLY	PHE	1.99E-06	18.659	20.07	31.991	0	16	1
593	GLY	TYR	3.58E-05	18.659	19.86	31.991	0	16	1
593	GLY	ARG	0.00078	18.659	16.17	31.991	-248.98	16	1
593	GLY	GLU	0.000276	18.659	18.73	31.991	-61.35	16	1
593	GLY	ASN	0.00134	18.659	19.16	31.991	-56.56	16	1
593	GLY	ASP	0.0163	18.659	18.89	31.991	-29.16	16	1
593	GLY	VAL	0.000935	18.659	19.18	31.991	6.66	16	1
593	GLY	CYS	0.00166	18.659	19.06	31.991	12.77	16	1
593	GLY	ILE	4.55E-05	18.659	19.8	31.991	22.76	16	1
593	GLY	SER	0.0609	18.659	18.83	31.991	28.13	16	1
463	ILE	TYR	2.55E-05	18.596	15.94	31.621	-260.1	5	2
481	ALA	LEU	7.06E-05	18.575	18.68	32.195	2.47	8	2
481	ALA	LYS	0.000274	18.575	17.32	32.195	3.05	8	2
481	ALA	GLU	0.00121	18.575	17.92	32.195	12.04	8	2
505	LEU	ARG	0.000861	18.589	18.58	31.429	24.96	4	2
508	THR	ARG	0.000295	18.594	18.59	31.461	-115.96	2	2
508	THR	LYS	0.000916	18.594	18.59	31.461	12.12	2	2
514	LEU	ILE	0.00165	18.57	18.4	31.667	21.52	10	2
514	LEU	ARG	5.22E-05	18.57	17.18	31.667	21.09	10	2
529	THR	PHE	3.77E-05	18.583	15.91	32.196	-106.19	14	2
529	THR	MET	1.74E-05	18.583	18.65	32.196	-28.68	14	2
529	THR	VAL	4.99E-05	18.583	18.72	32.196	28.05	14	2
529	THR	ASN	0.000996	18.583	18.55	32.196	27.98	14	2
532	CYS	ARG	0.000812	18.49	13.04	31.578	-0.71	9	2

535	SER	ILE	0.000391	18.717	19.02	31.813	28.17	5	2
535	SER	TYR	0.00262	18.717	18.68	31.813	29.19	5	2
593	GLY	HIE	1.66E-05	18.659	19.58	31.991	0	16	2
593	GLY	THR	2.67E-05	18.659	19.06	31.991	6.63	16	2
471	VAL	PHE	0.000785	18.645	16.37	31.624	27.32	3	3
481	ALA	ILE	5.14E-05	18.575	14.2	32.195	16.28	8	3
481	ALA	VAL	0.00146	18.575	17.77	32.195	27.92	8	3
514	LEU	VAL	0.000522	18.57	18.3	31.667	27.8	10	3
514	LEU	PRO	0.00121	18.57	18.12	31.667	29.31	10	3
593	GLY	HIP	1.66E-05	18.659	19.56	31.991	0	16	3
593	GLY	HID	1.66E-05	18.659	19.55	31.991	0	16	4
471	VAL	MET	8.44E-06	18.645	18.96	31.624	30.12	3	5
481	ALA	ARG	1.53E-05	18.575	17.02	32.195	-7	8	5
481	ALA	ASP	4.39E-06	18.575	19.21	32.195	26.65	8	5
505	LEU	TYR	8.68E-06	18.589	18.59	31.429	13.95	4	5
514	LEU	PHE	1.53E-05	18.57	18.19	31.667	0	10	5
535	SER	ARG	1.91E-06	18.717	18.84	31.813	26.28	5	5
593	GLY	TRP	4.78E-06	18.659	18.83	31.991	0	16	5
593	GLY	LEU	3.87E-07	18.659	19.35	31.991	-172.14	16	5
463	ILE	HIE	5.14E-06	18.596	18.06	31.621	14.78	5	6
463	ILE	HIP	5.14E-06	18.596	17.51	31.621	14.5	5	6
463	ILE	HID	5.14E-06	18.596	18.08	31.621	23.91	5	6
481	ALA	GLN	1.80E-06	18.575	18.37	32.195	-57.06	8	6
516	PHE	THR	3.77E-06	18.593	18.56	32.276	29.08	1	6
528	VAL	ARG	5.57E-07	18.569	18.61	32.223	21.75	1	6
529	THR	TYR	1.12E-06	18.583	18.58	32.196	0	14	6
529	THR	ARG	4.98E-07	18.583	18.6	32.196	-73.94	14	6
529	THR	LEU	3.10E-06	18.583	18.56	32.196	-40.25	14	6
529	THR	LYS	1.74E-06	18.583	18.57	32.196	-16.49	14	6
532	CYS	HIP	4.44E-06	18.49	14.51	31.578	10.43	9	6
532	CYS	HIE	4.44E-06	18.49	12.48	31.578	8.45	9	6
532	CYS	ILE	1.16E-06	18.49	18.82	31.578	28.7	9	6
532	CYS	VAL	2.11E-06	18.49	18.7	31.578	29	9	6
505	LEU	MET	3.58E-07	18.589	18.6	31.429	28.4	4	7
505	LEU	GLN	1.84E-06	18.589	18.6	31.429	29.15	4	7

514	LEU	HIP	3.37E-07	18.57	17.68	31.667	24.27	10	7
514	LEU	HIE	3.37E-07	18.57	17.72	31.667	24.93	10	7
514	LEU	HID	3.37E-07	18.57	17.72	31.667	25.66	10	7
529	THR	HIP	6.68E-08	18.583	18.58	32.196	-75.31	14	7
529	THR	HID	6.68E-08	18.583	18.58	32.196	-74.57	14	7
529	THR	HIE	6.68E-08	18.583	18.58	32.196	-73.8	14	7
529	THR	ASP	2.37E-06	18.583	18.51	32.196	27.31	14	7
529	THR	CYS	1.13E-07	18.583	18.53	32.196	29.46	14	7
531	TRP	PRO	5.44E-07	18.512	16.76	31.57	0	1	7
532	CYS	HID	4.44E-06	18.49	14.39	31.578	10.57	9	7
514	LEU	GLU	5.81E-08	18.57	17.36	31.667	26.07	10	8
514	LEU	LYS	1.68E-06	18.57	17.37	31.667	28.06	10	8

Appendix B: More Details on Applying DexDesign to Predict *de novo* D-peptide Inhibitors to CALP and MAST2

In this appendix, we provide further information pertinent to the DexDesigned *de novo* D-peptide inhibitors targeting CALP and MAST2 presented in Chapter 3.

B.1 K Score Normalization*

The K* algorithm[21,47] and the K* scores it predicts have been validated experimentally in many previous works[20,22–24,28–30,47,72]. While sometimes we have observed the K* scores to correlate quantitively (Pearson) with K_a[29,72,214], we have greater evidence that K* scores better correlate with K_a using a ranking (Spearman) paradigm[24,35]. For example, we recently analyzed the accuracy of EWAK* (an accelerated K*-derivative algorithm) predictions on 41 c-Raf-RBD variants binding to KRas and found that the K* score and experimental ranks correlated with a Spearman ρ of 0.81[24]. Therefore, when we convert K* scores to physical quantities, such as ΔG, we normalize the K* scores based on available empirical evidence (when available). For example, for the CALP-PEPs, we scale our ΔG values based on existing binding affinity data for CFTR (K_i = 420 ± 80 μM)[183] and kCAL01 (K_i = 2.3 ± 0.2 μM)[23], for which we have also predicted K* scores (see Table 12).

The necessity to normalize K* scores when converted to physical quantities is due to simplifications necessary for modeling the systems computationally. We have previously described normalization in Wang (2022)[26], and review the reasons here for the reader. First, since it is computationally intractable to model large continuous movements of each atom in a molecule while simultaneously searching over protein sequences and conformations, our K* computations focus on modeling continuous movements for residues within the PDZ binding site. Second, the OSPREY energy function models solvent using a residue-pairwise approximation to the energy field, namely the Effective Energy Function (EEF1) for proteins in solution[230]. Analysis of previous K* designs has shown that the EEF1 contribution to the OSPREY energy function can overestimate van der Waals terms. Lastly, limitations in the input model, as well as the user-specified *conformation space* (Section 2.3.1.1 in the main text defines conformation space), can cause an over- or underestimation in the K*-predicted enthalpy or entropy of the bound protein:peptide complex.

B.2 The DPR scaffolds, CALP-PEPs, and MAST2-PEPs

Table 11: The DPR scaffolds for each of the D-peptide redesigns. Each of the DPR scaffolds was extracted from an extant empirical protein structure from the Protein Databank[175,231]. **Name**: the name of the DPR scaffold. **Source PDB ID**: the PDB ID that can be used to retrieve the empirical structure from the Protein Databank. **Source Residues**: The chain and amino acid range in the empirical structure uniquely identifying the source of the DPR. **Template Peptide**: The PDZ-binding peptide used as a search query to MASTER[57], which returned a result set containing the DPR scaffold. **Backbone RMSD (Å)**: the full backbone RMSD between the template peptide and the DPR scaffold. **Sequence**: The DPR scaffold's peptide sequence.

Name	Source PDB ID	Source Residues	Template Peptide	Backbone RMSD (Å)	Sequence
CALP-DPR1	1g1k	A117-122	kCAL01[22]	0.91779	DGGAFG
CALP-DPR2	3u0o	A137-142	kCAL01	0.88123	AGGHSI
CALP-DPR3	4m6r	A45-51	kCAL01	0.88412	GGGISL
CALP-DPR4	7bjt	B403-409	kCAL01	0.89913	QGGVAI
CALP-DPR5	1c8n	C61-66	kCAL01	0.98744	AGGFVT
MAST2-DPR1	3s4k	A76-81	PTEN[174]	0.91439	EGGSVV
MAST2-DPR2	6zec	H5-10	PTEN	0.9417	QQWGAG
MAST2-DPR3	6tc2	L22-27	PTEN	0.95832	RGFFYT

Table 12: K* scores and structural features validation of the CALP-PEPs, kCAL01, and the CFTR C-terminal SLiM. ID: the ID of the DexDesign-generated D-peptide inhibitor, or the endogenous ligands. **DPR ID**: The scaffold from which the CALP-PEP was generated (see Table 11 for the structural source of the DPR) or, in the case of kCAL01 and the CFTR C-term SLiM, the PDB ID of the structure. **Sequence**: the amino acid sequence of the peptide. K* score [bounds]: The OSPREY-computed $\log_{10}$ K* score (see Section 1.2.2 for a description of the K* algorithm[17,21]) and the computed lower- and upper-bounds. The K* scores were computed to an ε of 0.9. **ΔK* score**: The increase in the CALP-PEP's $\log_{10}$ K* score over its source DPR scaffold, indicating the predicted improvement in binding K* and DexDesign were able to achieve through sequence optimization. **CBL H-bonds**: The number of H-bonds formed between CALP's carboxylate binding loop (CBL) and the peptide's C-terminal carboxylate. **β-strand H-bonds**: the number of H-bonds formed between CALP's β2 strand and the peptide mainchain. **P^{-1} AA**: The type of amino acid filling CALP's hydrophobic pocket at the peptide's position P^{-1} († indicates position P^{0} in L-peptides).

ID	DPR ID	Sequence	K* score [bounds]	ΔK* score	CBL H-bonds	β-strand H-bonds	P^{-1} AA
KCAL01	6ov7[22]	WQVTRV	30.4 [30.4 - 31.1]	N/A	3	3	Ile†
CFTR C-TERM	2lob[182]	VQDTRL	16.2 [15.7 - 16.6]	N/A	3	3	Leu†
CALP-PEP1	CALP-DPR1	RMGRFK	23.2 [22.4 - 24.2]	11.1	0	3	Phe
CALP-PEP2	CALP-DPR1	DMGRFK	21.8 [21.6 - 22.8]	9.7	1	2	Phe
CALP-PEP3	CALP-DPR1	RGGRFK	22.3 [22.0 - 23.3]	10.2	0	2	Phe
CALP-PEP4	CALP-DPR2	RQGRHI	26.0 [25.1 - 27.0]	13.0	2	2	His
CALP-PEP5	CALP-DPR2	AQGRHM	25.4 [24.6 - 26.4]	12.4	2	2	His
CALP-PEP6	CALP-DPR2	AQGRHI	25.2 [24.8 - 26.2]	12.2	2	2	His
CALP-PEP7	CALP-DPR3	RGGIHK	24.0 [23.9 - 25.0]	11.2	3	3	His
CALP-PEP8	CALP-DPR3	RGGRHL	25.3 [25.0 - 26.2]	12.5	3	3	His
CALP-PEP9	CALP-DPR3	RGGRHK	26.1 [25.4 - 27.1]	13.3	3	3	His
CALP-PEP10	CALP-DPR4	RQGRHM	23.6 [22.6 - 24.6]	10.1	4	2	His
CALP-PEP11	CALP-DPR4	QQGRHM	23.4 [22.5 - 24.4]	9.9	4	3	His
CALP-PEP12	CALP-DPR4	RQGVHM	22.6 [21.8 - 23.6]	9.1	4	3	His
CALP-PEP13	CALP-DPR5	AGGRHR	18.9 [17.9 - 19.9]	7.6	1	3	His
CALP-PEP14	CALP-DPR5	AGGRHM	18.8 [17.8 - 19.8]	7.5	1	2	His
CALP-PEP15	CALP-DPR5	AGGRMK	18.7 [17.8 - 19.7]	7.4	1	2	Met

Table 13: K* scores and structural features validation of the MAST2-PEPs and PTEN6. ID: the ID of the DexDesign-generated D-peptide inhibitor, or endogenous ligands. **DPR ID**: The scaffold from which the MAST-PEP was generated (see Table 11 for the structural source of the DPR) or, in the case of PTEN6, the PDB ID of the structure. **Sequence**: the amino acid sequence of the peptide. **K* score [bounds]**: The OSPREY-computed $\log_{10}$ K* score (see Section 1.2.2 for a description of the K* algorithm[17,21]) and the computed lower- and upper-bounds. The K* scores were computed to an ε of 0.9. **ΔK* score**: The increase in the MAST2-PEP's $\log_{10}$ K* score over its source DPR scaffold, indicating the predicted improvement in binding K* and DexDesign were able to achieve through sequence optimization. **Hydrophobic Pocket (location)**: The type and location of the amino acid filling MAST2's hydrophobic pocket. - indicates this residue is absent.

ID	DPR ID	Sequence	K* score [bounds]	ΔK* score	Hydrophobic Pocket (location)
PTEN6	2kyl[174]	TQITKV	28.8 [28.4 - 29.8]	N/A	Val (P^{0})
MAST2-PEP1	MAST2-DPR1	EMGDMD	30.9 [30.8 - 31.8]	20.2	Met (P^{-1})
MAST2-PEP2	MAST2-DPR1	EEGDMD	30.5 [30.4 - 31.1]	19.9	Met (P^{-1})
MAST2-PEP3	MAST2-DPR1	EQGDMD	30.3 [30.2 - 31.0]	19.7	Met (P^{-1})
MAST2-PEP4	MAST2-DPR2	EGEEDL	32.7 [32.7 - 33.0]	22.0	Leu (P^{0})
MAST2-PEP5	MAST2-DPR2	YGEEDL	32.6 [32.6 - 32.9]	21.9	Leu (P^{0})
MAST2-PEP6	MAST2-DPR2	FGEEDL	32.2 [32.2 - 32.3]	21.5	Leu (P^{0})
MAST2-PEP7	MAST2-DPR3	EDDEYE	30.7 [30.7 - 30.7]	21.9	-
MAST2-PEP8	MAST2-DPR3	EEDEYE	30.7 [30.7 - 30.7]	21.8	-
MAST2-PEP9	MAST2-DPR3	DDDEYE	29.4 [29.4 - 29.4]	20.5	-
MAST2-PEP10	MAST2-DPR2	IGEEDL	32.2 [32.2 - 32.3]	21.5	Leu (P^{0})
MAST2-PEP11	MAST2-DPR2	HGEEDL	31.6 [31.6 - 32.4]	20.8	Leu (P^{0})
MAST2-PEP12	MAST2-DPR2	QGEEDL	31.4 [31.4 - 31.9]	20.7	Leu (P^{0})
MAST2-PEP13	MAST2-DPR2	VGEEDL	31.2 [31.2 - 31.2]	20.5	Leu (P^{0})
MAST2-PEP14	MAST2-DPR2	MGEEDL	31.0 [31.0 - 31.7]	20.3	Leu (P^{0})
MAST2-PEP15	MAST2-DPR1	EYGDMD	30.3 [30.2 - 31.0]	19.7	Met (P^{-1})

Table 14: Results of validation and control experiments with D-amino acid GyGlanvdessG, DPRV, and ST0929 control experiment scaffold bound to streptavidin. Peptide: the name of the D-peptide. **Sequence**: the sequence of the D-peptide. **K* score**: the $\log_{10}$ K* score indicating the predicted binding affinity using the K* algorithm (see Section 1.2.2). **Trp79 contacts in β barrel**: boolean value stating if the D-peptide establishes hydrophobic interactions with streptavidin Trp79. **# H-bonds in flexible loop**: the number of hydrogen bonds established between the D-peptide and the flexible loop, comprised of streptavidin Ser45-Ser52.

Peptide	Sequence	K* score	Trp79 contacts in β barrel	# H-bonds in flexible loop
GYGLANVDESSG	GLANVDESS	32.2	Yes	1
ST0929 (CONTROL)	GLANVDESS	26.6	Yes	1
DPRV	WWMIGDWND	32.8	Yes	2

B.3 The MASTER database, D- vs. L-sidechain comparison, and DexDesign validation

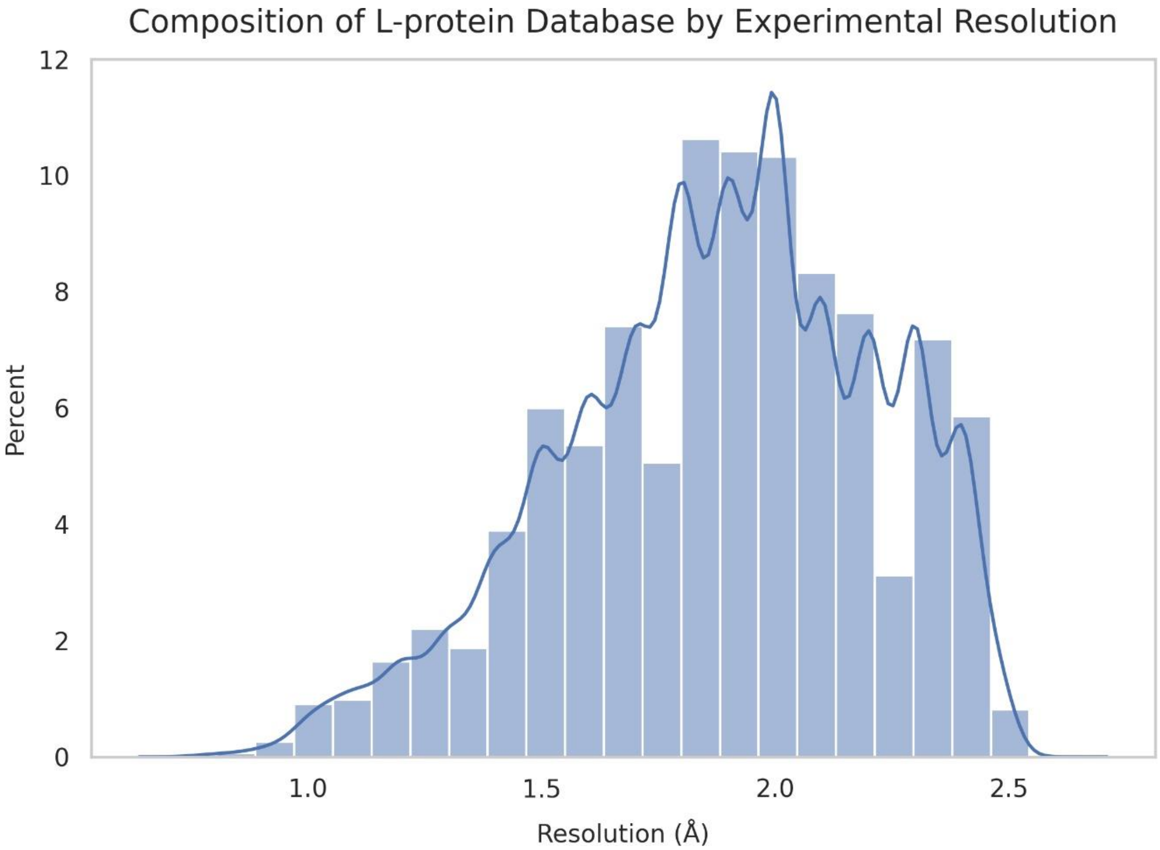


Figure 16: Composition of DexDesign's MASTER search database by resolution. Step 3 of the DexDesign search algorithm executes a search over a protein structure database to identify L-protein segments with backbones similar (by RMSD) to the D-peptide query. For this, DexDesign uses MASTER[57] to search over a protein structure database. We created a database of high-resolution L-protein structures by mining the RCSB PDB[175] for crystallographically determined structures omitting DNA, RNA, and small molecules with a resolution of at most 2.5 Å. This resulted in a database containing 119,160 structures of varying resolutions. The histogram above shows the distribution of the resolution of the structures in the database we created. The mean, median, and model resolutions are 1.88 Å, 1.9 Å, 2.0 Å, respectively. The database was generated on 01/24/2023.

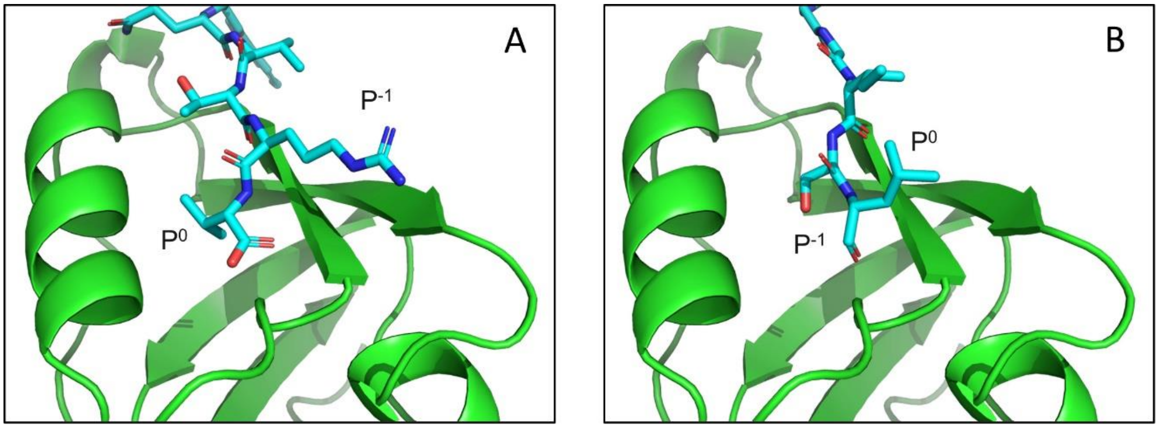


Figure 17: The sidechains of backbone-aligned L and D peptides point in opposite directions. (A) P^0 and P^{-1} orientation for kCAL01 bound to CALP (PDB ID 6ov7)[22]. The nonpolar P^0 residue (Val) points towards CALP, filling the hydrophobic pocket. The charged P^{-1} residue (Arg) is oriented towards the CBL, where it forms three hydrogen bonds (not shown). **(B)** P^0 and P^{-1} orientation for CALP-DPR3 (D-form, PDB ID 4m6r, residues A45-51, GGGISL) with CALP (L-form). In contrast to **(A)**, the P^0 residue (Leu) points toward the CBL. Furthermore, the P^{-1} residue (Ser) points toward the hydrophobic pocket. As expected, the residue positions are oriented in opposite directions between the L and D peptide, with an angle of 151° between kCAL01 and CALP-DPR3's P^0 residue and 155° between kCAL01 and CALP-DPR3's P^{-1} residue. Therefore, redesign methods for CALP-DPR3 focus on filling the hydrophobic pocket with mutations and continuous flexibility at P^{-1} and establishing adequate hydrogen bonds between P^0 and the CBL.

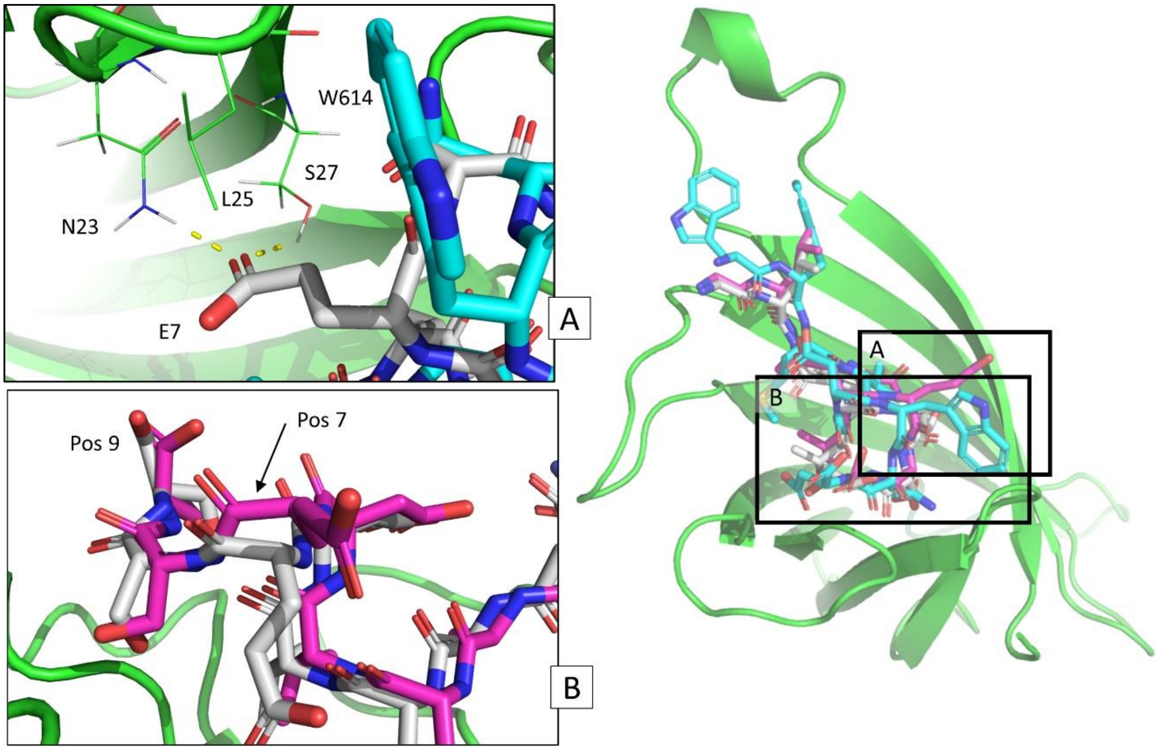


Figure 18: Backbone and hydrogen bonds at ligand residue positions 7 and 9 result in unique binding geometry and chemistry among D-amino acid GyGlanvdessG, DPRV, and ST0929 control experiment scaffold bound to streptavidin. (A) Comparison of backbone orientation and hydrogen bonds between D-peptide GyGlanvdessG:streptavidin Glu7 and DPRV:streptavidin Trp614, which are both the 7th residue from the N-terminus. The ST0929 control ligand is omitted for clarity. While GyGlanvdessG (grey) orients its Glu7 towards residues Asn23 and Ser27 in streptavidin (green cartoon and lines) to make two hydrogen bonds (yellow dashes), DPRV's (cyan) Trp7 is unable to make contact with these residues. Were DPRV Trp614 to rotate towards these residues, it would encounter steric clashes with streptavidin residue Leu25. This is an example of irrecoverable ligand interactions due to different backbone geometries. **(B)** Backbone geometry differences at residue positions 7 and 9 between GyGlanvdessG:streptavidin and the ST0929 control experiment scaffold bound to streptavidin. DPRV is omitted for clarity. The full backbone RMSD of the ST0929 control scaffold (magenta) to GyGlanvdessG (grey) is 0.48 Å. This seemingly small alignment difference yields a starting scaffold where 100% native sequence recovery results in a suboptimal binder, with the most notable backbone positions differences occurring at the C-terminus and 7th residue from the N-terminus. The C-terminal oxygens point in different directions, with a distance of 1.8 Å between them. Further, the C_α carbons at the 7th residue from the N-terminus are 1.2 Å apart. As described for DPRV in **(A)**, residues at the C-terminus and 7th residue position from the N-terminus exhibit either loss or generation of hydrogen bonds after application of the DexDesign protocol. Unlike GyGlanvdessG and similar to DPRV, the ST0929 control experiment scaffold is unable to make hydrogen bonds with streptavidin Ser27 (not shown). However, similar to GyGlanvdessG, it lacks a hydrogen bond in the flexible binding loop. Overall, this illustrates how differences in two key backbone positions can influence designability and predicted binding affinity.

Appendix C: Protocol for Predicting Drug Resistant Protein Mutations to an ERK2 Inhibitor using Resistor

Shortly after our primary article on Resistor was published in Cell Systems[30], STAR Protocols (another Cell Press journal) invited us to submit a step-by-step protocol for using Resistor to predict resistance mutations. As I had already received several queries from colleagues asking for guidance on using Resistor, STAR Protocol's invitation to systematize and document its usage seemed like a sensible thing to do. And in retrospect I am glad that we did. The contents of this appendix are now often one of the first things we share with new protein designers using OSPREY, including those not particularly interested in predicting resistance. The reason for this is that not only is this protocol a step-by-step instruction manual for using Resistor, but since Resistor calls K* as a subroutine, it is also a step-by-step instruction manual for running what is OSPREY's most popular design algorithm: K*.

For this protocol, I decided not to merely rehash the Resistor predictions we made and validated for the Cell System's publication, but rather thought it would be more interesting to apply Resistor to predict resistance mutations in a new (to me) kinase and inhibitor: ERK2 and the ERK1/2 inhibitor SCH779284 used to treat melanoma. This appendix is adapted from the following publication:

> Guerin, N., Kaserer, T. & Donald, B. R. Protocol for predicting drug-resistant protein mutations to an ERK2 inhibitor using RESISTOR. *STAR Protocols* **4,** 102170 (2023).

C.1 Before you begin

This section describes the minimal hardware and operating system requirements, where to obtain the requisite software and its installation procedure, and the file formats of the sequence and structural inputs required to run Resistor. For the purposes of demonstration, we use Resistor to predict resistance mutations on the ERK2 kinase to the inhibitor SCH772984 (hereafter referred to as SCH7). Previously, we have used Resistor to prospectively predict resistance mutations in EGFR and BRAF, which we then validated experimentally[30]. In addition, we have employed aspects of Resistor, including multistate K* design and mutational signature probabilities, in other applications, such as our development of algorithms like BBK* (Branch and Bound Over K*)[38] and our predictions of resistance-conferring mutations to inhibitors of kinases such as KIT, EGFR, ABL1, and ALK[28].

Here we offer abbreviated definitions and references for the terminology we use throughout this protocol. *Positive design* is the use of computational protein design algorithms to improve an objective, such as ligand binding. *Negative design* is the opposite, i.e., the goal is to make an objective worse, such as to ablate binding. Resistor uses *multistate design*[29,36,46,72,73], or both positive and negative design in parallel, to mimic how mutations affect the competitive balance between a protein's endogenous ligand and a competitive inhibitor. Resistance can occur via a protein's increased activity with its endogenous ligand, decreased binding with an inhibitor, or a combination of these factors[28–30,72].

Resistor employs Pareto optimization over positive and negative design, mutational signature probabilities, and hotspot scores to rank prospective resistance mutants. The positive and negative design portions use the *K* algorithm*[21] implemented in OSPREY[35], which generates low-energy molecular ensembles to compute the partition functions the algorithm uses to provably approximate binding affinity, K_a[21,46]. *Mutational signature probabilities* are derived from data provided by Alexandrov et al.[81] and denote the probability a DNA base will mutate to another base in a given sequence context and cancer type. A *hotspot score* is the number of sequences with a mutation at a particular residue location which multistate design criteria predicts as structural resistance mutations[28,30].

Figure 19 contains a conceptual overview of the Resistor protocol's main phases.

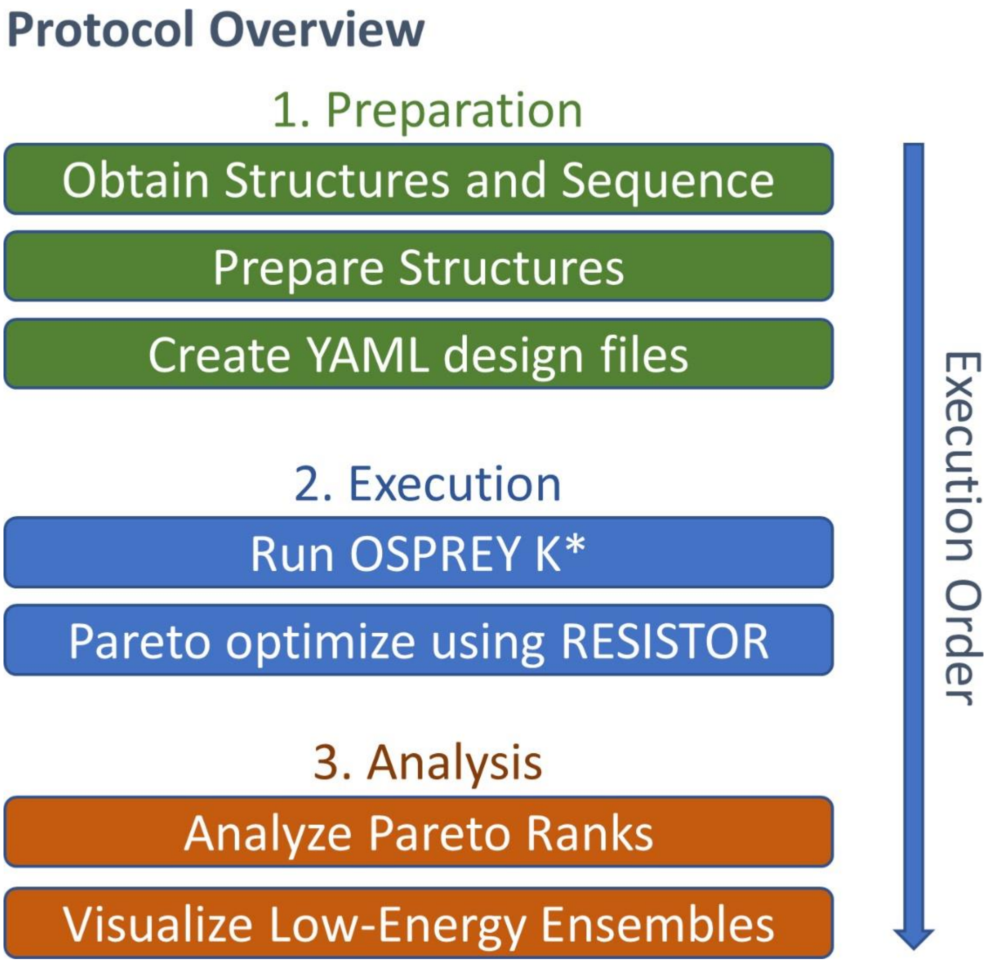


Figure 19: Conceptual overview of the main steps involved in executing the Resistor protocol. The first phase, Preparation, involves obtaining the positive and negative design structure files in PDB format, along with the corresponding cDNA sequence. The structures need to be prepared for OSPREY K* design. Finally, the structures and additional inputs (outlined below) must be collected into a YAML design file. The second phase, Execution, involves using the OSPREY K* algorithm to compute provable approximations to the binding constant, $\boldsymbol{K_a}$, and using Resistor to filter the results, assign mutational probabilities, and Pareto optimize. The final phase, Analysis, is where the user examines the Resistor-provided output of Pareto ranks and low-energy molecular ensembles. The details involved in each of these steps are explained comprehensively in this appendix.

C.1.1 Hardware and software

Resistor requires a minimum of 32 GiB of RAM and 5 GiB of free hard disk space. You also will need to have a good text editor on your computer: vim, emacs, or any other text editor that can be used for editing ASCII characters will do; programs like Microsoft Word or LibreOffice Writer will not. Our demonstration of the protocol is on a

Linux operating system, although with minor adjustments the process below could be carried out on Windows and macOS operating systems.

C.1.2 Installing the software dependencies

Resistor requires Java 17, Miniconda, AmberTools, Julia, and OSPREY.

1. Install Java 17.
    a. Download an archive for the latest version of Java 17 for your platform from https://jdk.java.net/archive/
    b. Extract the archive to a location on your computer, e.g., `$HOME/java/jdk-17.0.2`.
    c. In your shell's profile, set the `JAVA_HOME` environment variable to the location you extracted the archive to, and add the java executable to your path, e.g.,

```
> export JAVA_HOME=$HOME/java/jdk-17.0.2
> export PATH=$PATH:$JAVA_HOME/bin
```

    d. Verify java is available on your shell's path by opening a new terminal window, typing `java -version`, and hitting enter. You should see output like the following:

```
> java -version
openjdk version "17.0.2" 2022-01-18
OpenJDK Runtime Environment (build 17.0.2+8-86)
OpenJDK 64-Bit Server VM (build 17.0.2+8-86, mixed mode, sharing)
```

    Movie S1 in from Guerin et al.[31] demonstrates this procedure on Fedora Linux.

2. Install Python using Miniconda

a. Download the latest Python 3 version of Miniconda from https://docs.conda.io/en/latest/miniconda.html.

b. Run the interactive installer, e.g.:

```
> sh Miniconda3-latest-Linux-x86_64.sh
```

c. When the installer asks you *“Do you wish the installer to initialize Miniconda3 by running conda init? [yes|no]”*, type `yes` and hit enter.

d. Close and re-open your shell.

e. Now, when you log into your shell, a Miniconda environment is activated. Run the following command so its environment is *not* activated by default.

```
> conda config --set auto_activate_base false
```

Movie S2 from Guerin et al.[31] demonstrates these steps on Fedora Linux.

3. Install AmberTools

a. Install AmberTools22 using the installation instructions for *“Binary distribution via Conda”* on https://ambermd.org/GetAmber.php#ambertools. In short:

```
> conda create --name AmberTools22
> conda activate AmberTools22
(AmberTools22) > conda install -c conda-forge ambertools=22 compilers
```

b. To verify that AmberTools is correctly installed, type:

```
> antechamber -h
```

A help message for the antechamber program should be displayed. Movie S3 from Guerin et al.[31] demonstrates these steps on Fedora Linux.

Optional: You can install the conda-packaged yamllint program into this conda environment. Yamllint is used to check the syntactic validity of YAML documents:

```
> conda activate AmberTools22
(AmberTools22)> conda install -c conda-forge yamllint
```

If you choose to install yamllint, you should also create a default configuration file that disables its line length check. To do so, create the file `$HOME/.config/yamllint/config` (and the intermediary directories as necessary), and add the following content:

```
extends: default

rules:
  line-length: disable
```

4. Install Julia
    a. Download and extract the latest stable release of Julia. Resistor was developed using Julia v1.6, but any v1 release of Julia post Julia 1.6 should work.
        i. Go to https://julialang.org/downloads/ to get the latest Julia package.
        ii. Download the architecture-specific Linux archive to your computer.
        iii. Extract the archive to a location on your computer, e.g., `$HOME/julia/julia1.8`

b. In your shell’s profile, add the executable to your path, e.g.,

```
export JULIA_HOME=$HOME/julia/julia1.8
export PATH=$PATH:$JULIA_HOME/bin
```

c. Close and re-open your shell. Then, to verify that Julia is correctly installed, type:

```
> julia --version
```

which should print out the version of Julia you downloaded. Movie S4 from Guerin et al.[31] demonstrates these steps on Fedora Linux.

5. Install OSPREY with Resistor

a. Download OSPREY version 3.3 from https://github.com/donaldlab/OSPREY3/releases/3.3-resistor

b. Extract the OSPREY distribution:

```
> tar --file osprey-3.3.tar --extract
```

c. Add the OSPREY executable on your PATH for simplified access. Assuming you have extracted the archive in the previous step in your home directory, add the following line to your shell’s profile file:

```
export OSPREY_HOME=$HOME/osprey-3.3
export PATH=$PATH:$OSPREY_HOME/bin
```

d. Verify you have OSPREY on your path by executing the following command in the terminal, which should display help text:

```
> osprey affinity --help
```

Movie S5 from Guerin et al.[31] demonstrates this procedure on Fedora Linux. If you do not see the help text, see Problem/Potential Solution 1 (C.5.1).

C.1.3 Obtaining the sequence and structure files

6. Download your positive and negative design structure files
    a. Navigate to the Protein Data Bank (https://www.rcsb.org/) in your browser.
    b. Search for the protein of interest. You will need to download a structure of the protein bound to the drug and the protein interacting with the endogenous ligand.
    c. For ERK2 bound to SCH7, search the Protein Data Bank for by PDB ID 4qta[232] and download the file in PDB format.
    d. For ERK2 bound to AMP-PNP (adenylyl-imidodiphosphate, an analogue of ATP), search the Protein Data Bank by PDB ID 2y9q[233] and download the file in PDB format.
7. Download the coding DNA sequence

Note: There are many places on the internet to download DNA sequences. For sequences of proteins implicated in carcinogenesis, such as ERK2, the COSMIC database[108] is one such good choice.

a. In a web browser, navigate to https://cancer.sanger.ac.uk/cosmic.
b. Search for ERK2 and go to the gene view.
c. Download the cDNA sequence (ENST00000215832.10) in FASTA file format.

8. Choose your cancer-type specific mutational probabilities JSON file
    a. Identify the probabilities file you need. For this protocol, we will use the melanoma probabilities file, *melanoma.json.*

b. Mark down the path to this file, which you will use in C.2.3.

Note: The Resistor directory within the OSPREY distribution (`osprey-3.3/resistor`) contains mutational probability files for melanoma, non-small cell lung cancer, stomach cancer and pancreatic cancer. It is also possible to create your own mutational probabilities file, which is covered in Section 2.2.3.

CRITICAL: Ensuring that the following prerequisites are met helps avoid downstream prediction problems: 1. When possible, use high-quality, high-resolution structures. While the cut-off for resolution is still a matter of discussion in the scientific community, previous successful designs have used X-Ray diffraction resolutions ranging between 1.4 and 3.15 Å[20,24,26,29,72]. We have also had success with cryo-EM resolutions between 3.4 and 11.5 Å. For NMR structures, we recommend that the structure determination use RDCs; 2. Check that the residue numbers and amino acid types in the positive and negative protein structures are the same, e.g. ALA 10 in the structure for the positive design and ALA 10 in the structure for the negative design refer to the same residue; and 3. that the cDNA sequence translates to the amino sequence in the structure files, i.e., they represent the same genetic variant. Furthermore, the FASTA file must begin with the codon that translates to residue number 1. In this example, PDB ID 4qta (ERK2:SCH7) has a resolution of 1.45 Å, PDB ID 2y9q (ERK2:AMP-PNP) has a resolution of 1.55 Å, and the residue numbering in the two structures are the same and correspond to the canonical numbering also used in the FASTA file. See Movie S6 from Guerin et al.[31] for a demonstration of carrying out these checks.

If your checks indicate discrepancies exist, you will need to manipulate the files to resolve them. As the PDB and FASTA file formats are standard in the fields of structural biology and bioinformatics there are many tools available for their manipulation, including Maestro[221] for manipulating structural information. Yet as both file formats are defined in human-readable ASCII text, oftentimes the simplest way to make any necessary tweaks in the files is with a standard text editor, such as emacs or vim.

In cases where empirical structures are not available, it is possible to use docking, homology modeling, or other computational modeling techniques to generate structures[23,26,29,30,72]. For example, computational tools such as Modeller[234] or Alphafold[13] could be used to predict an initial protein structure, and docking tools such as AutoDock Vina[11] or those included in Maestro[221] could be used to dock the positive and negative design ligands[28,30]. With the recent explosion of available structural models, such as the Alphafold Protein Structure Database[14] it may even be the case that a computationally predicted starting structure already exists. When taking such an approach, it is critical to have high confidence in the accuracy of any computationally generated structures as Resistor is very sensitive to variation in structural input.

C.1.4 Key Resources Table

Table 15 shows where you can obtain all the key resources needed to execute this demo.

Table 15: Key Resources for Resistor.

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited Data		
Model of ERK2:AMP-PNP protein structure	Garai et al., 2012	PDB ID 2y9q
Model of ERK2:SCH7 protein complex structure	Chaikuad et al., 2014	PDB ID 4qta
cDNA of ERK2	Tate et al., 2019	ENST00000215832.10
Software and Algorithms		
OSPREY 3.3	Hallen et al., 2018	https://github.com/donaldlab/OSPREY3/releases/3.3-resistor
AmberTools22	Case et al., 2022	http://ambermd.org/GetAmber.php
Maestro	Schrödinger, LLC	https://www.schrodinger.com/products/maestro
Miniconda	Anaconda, Inc.	https://docs.conda.io/en/latest/miniconda.html
Yamllint	Vergé, 2023	https://anaconda.org/conda-forge/yamllint

C.2 Step-by-step method details

Here we describe the step-by-step details of how to use Resistor. These steps include how to 1) specify the K* positive and negative designs; 2) run OSPREY to compute each mutant’s positive and negative K* scores; 3) process the data to assign mutational probabilities and hotspot scores; and 4) assign Pareto ranks to each prospective mutant. As a demonstration case, we use Resistor to predict ERK2 mutants likely to arise in melanoma that may ablate the efficacy of the ERK1/2 inhibitor SCH7.

C.2.1 Specifying the K* positive and negative designs

In this step, we create the YAML files that are used to specify the input for the positive and negative K* designs. Positive design refers to improving the interaction between a protein and its endogenous ligand, which in this context is ERK2 with ATP. Negative design refers to ablating the binding between a protein and its targeting inhibitor, here ERK2 and SCH7. By this point, we assume you have completed the steps in C.1, including having downloaded the PDB structure files *4qta.pdb* and *2y9q.pdb*, and the FASTA-formatted cDNA sequence file *ENST00000215832.10.fasta*.

1. Prepare each of the structure files
    a. Open a terminal shell and activate the AmberTools environment you created in C.1):

```
> conda activate AmberTools22
```

    b. Run *pdb4amber* on the two ERK2 structures to add any missing atoms:

```
> pdb4amber --add-missing-atoms -i 2y9q.pdb -o 2y9q.p4a.pdb
> pdb4amber --add-missing-atoms -i 4qta.pdb -o 4qta.p4a.pdb
```

Note: *pdb4amber* renumbers the residues in the input structures, starting from 1. We would like to keep our canonical residue numbering, and luckily *pdb4amber* outputs a mapping file from the original numbers to the new numbers it assigned the residues. This file is titled the name of the input file for *pdb4amber*, suffixed with *_renum.txt*, e.g., *2y9q.p4a_renum.txt*. Within the OSPREY distribution there's a program called *p4a-undo.py* (found in the `osprey3.3/resistor` directory) which re-assigns the original numbering and chain identifiers.

c. Using the same AmberTools22 conda environment, run *p4a-undo.py* with each of the two output structures from *pdb4amber*:

```
> python p4a-undo.py 2y9q.p4a.pdb 2y9q.p4a_renum.txt > 2y9q.renum.pdb
> python p4a-undo.py 4qta.p4a.pdb 4qta.p4a_renum.txt > 4qta.renum.pdb
```

d. Add hydrogens to the AMP-PNP and SCH7 structures using a molecular modeling program such as Maestro.

Note: Epik[235] in Maestro[221] is quite good at correctly predicting pK_a and protonation states for small molecules. You will also need to compute the net charge of the small molecules for Step 3, which Epik and Maestro provide. SCH7's net charge is +1, whereas AMP-PNP has a net charge of -4.

e. Save the resulting protonated structures as *2y9q.h.pdb* and *4qta.h.pdb*.

CRITICAL: Ensure when saving these protonated structures that the resulting PDB files do not contain trailing whitespace (if it does, remove it using your text editor) and the chain identifiers have been correctly preserved.

2. Split the structure files into their protein and ligand components
    a. Open *2y9q.h.pdb* in a text editor.
    b. Extract the ATOM records corresponding to ERK2 and save them to a file called *2y9q.erk2.pdb*.
    c. Extract the ATOM records corresponding to AMP-PNP and save them as a file called *2y9q.amppnp.pdb.*
    d. Do the same for *4qta.h.pdb*, saving the corresponding files as *4qta.erk2.pdb* and *4qta.sch7.pdb*.
3. Generate the forcefield parameters and connectivity templates for SCH7 and AMP-PNP
    a. Activate your AmberTools environment, as in Step 1a.
    b. Use the *antechamber* program from AmberTools to generate template files (files with a *.prepi* extension), and *parmchk2* to generate forcefield modification files (files with a *.frcmod* extension):

```
> antechamber -i 2y9q.amppnp.pdb -fi pdb \
    -o amppnp.prepi -fo prepi \
    -c bcc -nc -4
> parmkch2 -i amppnp.prepi -f prepi -a Y -o amppnp.frcmod
> antechamber -i 4qta.sch7.pdb -fi pdb \
    -o sch7.prepi -fo prepi \
    -c bcc -nc +1
> parmkch2 -i sch7.prepi -f prepi -a Y -o sch7.frcmod
```

Note: For more information about these and other possible flags to the antechamber and parmchk2 programs, see section 16.1 of the Amber 22 Reference Manual, available from http://www.ambermd.org[171].

4. Create template coordinates for the small molecules

    a. Locate the *gen-templ-coords.sh* script you will use to generate the template coordinates (in the `osprey3.3/resistor` directory of the OSPREY distribution).

    b. Add the executable bit to the script by running the following command:

```
chmod u+x gen-templ-coords.sh
```

    c. Run *gen-templ-coords.sh* once for each of the ligands, using Unix pipe redirection to save the output. *gen-templ-coords.sh* expects as input the path of the ligand structure and the three-letter residue name of the ligand used in the structure:

```
> ./gen-templ-coords.sh 2y9q.amppnp.pdb ANP > amppnp.tc
> ./gen-templ-coords.sh 4qta.sch7.pdb 38Z > sch7.tc
```

5. Generate rotamers for AMP-PNP and SCH7

    a. To allow the ligands to translate, rotate, and flex slightly, we define the flexible dihedrals for the ligands.

    b. Determine the molecule-specific dihedrals using Maestro or other molecular visualization software. Figure 20 demonstrates determining the dihedrals in Maestro.

    c. Create a text file listing the dihedrals. The format of the file, and the rotamer specification for AMP-PNP is shown in Figure 21.

    d. Save the file as *amppnp.rot*.

    e. Repeat steps a-d for SCH7, saving that file as *sch7.rot*.

6. Create a template YAML file for the ERK2:AMP-PNP positive K* design

    a. The OSPREY package contains a template K* affinity YAML file, located at `osprey3.3/resistor/affinity.yaml`. Make a copy of this file:

```
> cp affinity.yaml erk2-amppnp.yaml
```

    b. Open the new file in your text editor and incorporate the files you've created thus far into the YAML file:

        i. Copy the contents of *2y9q.erk2.pdb* as the value for the `protein.coordinates` key.

        ii. Copy the contents of *2y9q.amppnp.pdb* as the value for the `ligand.coordinates` key.

        iii. Copy the contents of *amppnp.tc* as the value for the `ligand.extra_template_coordinates` key.

        iv. Copy the contents of *amppnp.prepi* as the value of the `ligand.extra_templates` key.

v. Copy the contents of *amppnp.rot* as the value of the `ligand.extra_rotamers` key.

Optional: You can use a YAML syntax validator, such as yamllint[236] to verify you have input syntactically valid YAML.

c. To verify that you have created the YAML file correctly, run OSPREY to verify the design file:

```
> osprey affinity --design erk2-amppnp.yaml --verify-design
```

The output of the command should look like:

```
WARNING: Using incubator modules: jdk.incubator.foreign
Design file validated.
```

See Problem/Potential Solution 2 (C.5.2) if your output is different, and Problem/Potential Solution 4 (C.5.4) if the command output says it can't parse the YAML file.

7. Create a template YAML file for the ERK2:SCH7 negative K* design

a. As in Step 6a, copy the template K* affinity YAML file:

```
> cp affinity.yaml erk2-sch7.yaml
```

b. Open *erk2-sch7.yaml* in your text editor and incorporate the following files into the negative design specification:

i. Copy the contents of *4qta.erk2.pdb* as the value for the `protein.coordinates` key.

ii. Copy the contents of *4qta.sch7.pdb* as the value for the `ligand.coordinates` key.

iii. Copy the contents of *sch7.tc* as the value for the `ligand.extra_template_coordinates` key.

        iv. Copy the contents of *sch7.prepi* as the value for the `ligand.extra_templates` key.

        v. Copy the contents of *sch7.rot* as the value for the `ligand.extra_rotamers` key.

    c. To ensure that your YAML file is in the correct format for OSPREY, use the *affinity* command's `--verify-design` flag to check the design file. The output of the command should look like:

```
WARNING: Using incubator modules: jdk.incubator.foreign
Design file validated.
```

    See Movie S7 from Guerin et al.[31] for a demonstration of how to do this step and see Problem/Potential Solution 2 (C.5.2) if your output is different.

8. Choose residues to mutate and create mutational scan designs

    a. Taking the files you created in Steps 6 and 7, add a YAML list of objects representing these mutants as the value of the `scan.residues` key, as is shown in Figure 22.

Note: For this example, we have chosen to investigate residues Y36, A52, I56, R67, E71, Q105, D106, L107, M108, D111, K114, L156, and C166.

    b. In each of the files generated in Steps 6 and 7, set the ligand as flexible by adding it to the `ligand.residue_configurations` key in the YAML file. Figure 23 shows how this is set in the *erk2-amppnp.yaml* and *erk2-sch7.yaml* files.

c. After adding these fields, again verify the syntax of the design files is correct:

```
> osprey affinity --design erk2-sch7.yaml --verify-design
> osprey affinity --design erk2-amppnp.yaml --verify-design
```

9. Generate the K* affinity designs for each of the point mutants

a. Using the files you modified in Step 8, use OSPREY to generate the positive and negative designs for each of the mutants:

```
> osprey affinity --design erk2-sch7.yaml \
    --do-scan --scan-flex-distance 2.2
> osprey affinity --design erk2-amppnp.yaml \
    --do-scan --scan-flex-distance 2.2
```

Note: The `--do-scan` flag instructs OSPREY to generate a K* affinity design centered on each of the residues specified in the `scan.residues` key. These K* affinity designs each include a single mutable residue, which is set to mutate to all the other amino acids, and a flexible shell around the mutating residue. The optional `–scan-flex-distance` parameter denotes the radius of the OSPREY-generated flexible shell centered on the design's mutable residue. It defaults to 2 Å.

b. Verify that a positive and negative YAML design specification is created for each of the 13 residues of interest set in Step 8a. The naming format of these files is *{original-name}.{residue}.yaml*, e.g., *erk2-sch7.A36.yaml*. There should be a total of 26 newly created files.

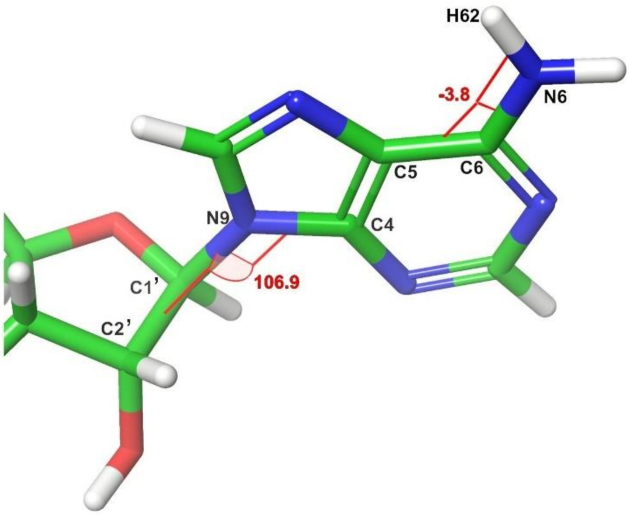
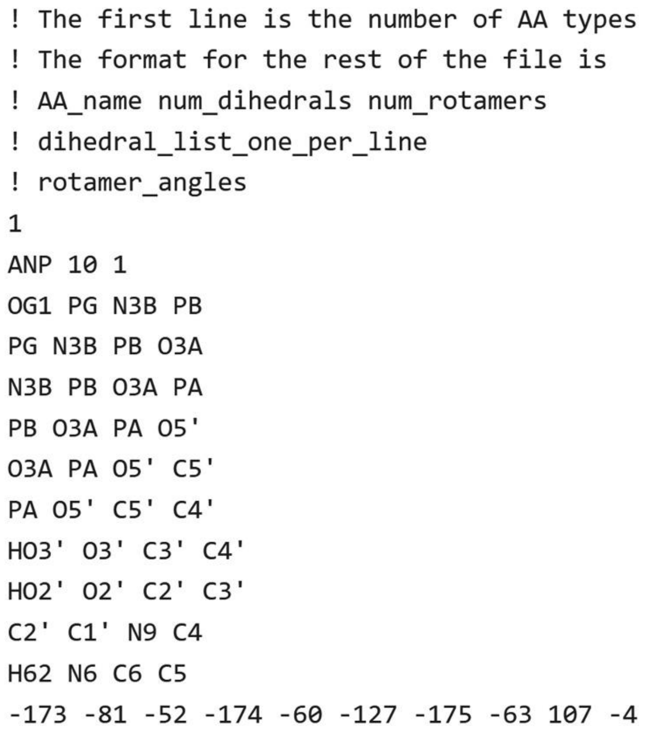


Figure 20: Demonstration of using Maestro to compute the H62-N6-C6-C5 and C2′-C1′-N9-C4 dihedral angles for the extra rotamers definition of AMP-PNP. The red lines and numbers show the dihedrals and the computed angles. In Figure 3, these dihedrals are included in the rotamer definition for AMP-PNP. The values -3.8 and 106.9 are rounded to their nearest whole value, -4 and 107, respectively.



```
! The first line is the number of AA types
! The format for the rest of the file is
! AA_name num_dihedrals num_rotamers
! dihedral_list_one_per_line
! rotamer_angles
1
ANP 10 1
OG1 PG N3B PB
PG N3B PB O3A
N3B PB O3A PA
PB O3A PA O5'
O3A PA O5' C5'
PA O5' C5' C4'
HO3' O3' C3' C4'
HO2' O2' C2' C3'
C2' C1' N9 C4
H62 N6 C6 C5
-173 -81 -52 -174 -60 -127 -175 -63 107 -4
```

Figure 21: Definition of a rotamer for AMP-PNP. We specify 10 dihedrals. These dihedrals allow K* in OSPREY to minimize continuously in a voxel around the dihedrals to search for low-energy conformations. This rotamer is defined by its atom names from the PDB file, 2y9q. Lines that begin with an exclamation point (!) are comments. The comments here explain the structure of the file.

```
scan:
  residues:
  - identity:
      chain: A
      res_num: 36
      aa_type: TYR
  - identity:
      chain: A
      res_num: 52
      aa_type: ALA
  -
    ...
```

Figure 22: Specification of the residues to scan. The value for the key is a list of objects representing residues in the structure. Each object has a chain key denoting the chain identifier in the structure, a `res_num` key denoting the residue number, and the `aa_type` key with the 3-letter amino acid code. In the example above, we specify that Y26 and A52 in chain A of the structure should be included in the scan. Below the ellipsis we would also include objects for I56, R67, E71, Q105, D106, L107, M108, D111, K114, L156, and C166.

```
ligand:
  residue_configurations:
  - mutability: [ ]
    flexibility:
      is_flexible: true
      include_structure_rotamer: true
      use_continuous: true
    identity:
      chain: B
      res_num: 441
      aa_type: 38Z
```

```
ligand:
  residue_configurations:
  - mutability: [ ]
    flexibility:
      is_flexible: true
      include_structure_rotamer: true
      use_continuous: true
    identity:
      chain: B
      res_num: 1359
      aa_type: ANP
```

Figure 23: Demonstration of how to specify that the ligand should be flexible in both the positive and negative designs. Left: residue 38Z on chain B at position 411 (which is SCH7) is set to be continuously flexible. Right: residue ANP on chain B at position 1359 (which is AMP-PNP) is set to be continuously flexible.

C.2.2 Running the K* predictions

The purpose of this step is to run the positive and negative K* mutant predictions with OSPREY. The range in expected time on this step is dependent on how many sequences you're predicting, the number of flexible residues you've configured in your conformation space, and the capabilities of your computer(s). For additional background information on the interpretation of K* values and how they are used in predicting resistance mutations, see the Results and STAR Methods sections of Guerin et al[30].

10. Run the positive and negative K* designs

    a. Set the amount of memory to dedicate to the OSPREY process by exporting the `JAVA_OPTS` environment variable. Set this as high as you can, given the hardware `you’re` running the design on. Here's how it could be set on a machine with 760GiB of RAM (while leaving some RAM for the operating system and other processes):

```
> export JAVA_OPTS="-Xmx720G -Xms720G"
```

    b. Execute the affinity command in OSPREY on each of the individual mutant design files that you generated in Step 9:

```
> osprey affinity --design {design-file} --frcmod {frcmod-file}
```

where *{design-file}* is the path to one of the design files you generated in Step 9, and *{frcmod-file}* is the path to the ligand-specific forcefield modification file you generated in Step 3b, e.g.:

```
> osprey affinity --design erk2-sch7.A36.yaml --frcmod sch7.frcmod
```

Note: There are optional flags you can pass to the *affinity* command that could be helpful for your predictions. These flags include `--save-confs`, `--ensemble-dir`, and `--cuda`. `--save-confs` takes an integer argument and denotes the number of low-energy conformations from the K* molecular ensemble that OSPREY should save of each sequence. It defaults to not outputting structures; if you want structures add this argument and specify a number greater than 0. `--ensemble-dir` takes a path as an argument and indicates where structures should be saved. And if you have access to CUDA-enabled Nvidia GPUs, you may find that the `--cuda` flag substantially decreases the amount of time needed to run your designs.

c. Execute the following command to print the per-residue type K* predictions to the terminal screen. If you also want to save the output (both standard out and standard error) to files, you can use Unix pipes to pipe the output to the *tee* program:

```
> osprey affinity --design erk2-sch7.A36.yaml \
    --frcmod sch7.frcmod > >(tee -a sch7.A36.stdout) \
    2> >(tee -a sch7.A36.stderr >&2)
```

Note: There is an optional parameter, `--epsilon`, which takes a double value as an argument and defaults to 0.683 (see the Supplemental Information of Ojewole et al. for justification for this default)[38].

Note: `--epsilon` must be between 0 and 1; values closer to 0 indicate a more accurate partition calculation and are thus more computationally expensive, and vice-versa. We recommend initially running K* affinity designs with an epsilon close to 1, such as 0.9999, and then gradually decreasing epsilon to obtain increasingly accurate K* scores while still in a reasonable amount of time.

C.2.3 Assign Pareto ranks

The purpose of this step is to compile and annotate the positive and negative K* mutant predictions with mutational signature probabilities and hotspot scores. In addition, we run the *resistor* program to compute the cutoff, *c*, from the K* predictions, and filter mutants that K* predicts not to be resistance mutants, or whose mutational probability is 0, and assign Pareto ranks.

Note: Your positive and negative K* predictions from Step 10 should be complete prior to beginning this step.

11. Compile the K* predictions

    d. Copy the template CSV file included in the OSPREY distribution (`osprey3.3/resistor/resistor.csv`) to *erk2-resistor.csv*.

    e. Using the output files from the predictions in Step 10, which contain the $\log_{10}$ K* scores for each of the sequences you evaluated at a particular residue location, fill out the following columns:

        i. *wild-type residue* should have the 3-letter amino acid code for the wild-type residue at *residue number*.

        ii. *residue number* should have the residue number of the residue.

        iii. *mutant residue* should have the 3-letter amino acid code for the mutant residue Resistor is evaluating.

iv. *wild-type K* (positive)* should have the $\log_{10}$ K* score computed on the ERK2:AMP-PNP structure for *residue number*.

v. *mutant K* (positive)* should have the $\log_{10}$ K* score computed on the ERK2:AMP-PNP structure for *residue number* when *mutant residue* is substituted for *wild-type residue*.

vi. *wild-type K* (negative)* should have the wild-type K* (negative) should have the $\log_{10}$ K* score computed on the ERK2:SCH7 structure for *residue number*.

vii. *mutant K* (positive)* should have the $\log_{10}$ K* score computed on the ERK2:SCH772894 structure for *residue number* when *mutant residue* is substituted for *wild-type residue*.

f. Complete a new row for each mutant sequence you evaluated in Step 10. Each positive/negative design pair from Step 10 evaluated 21 different residue types in each location, meaning we must complete 21 rows for each residue. See Table 16 for an example of a partially completed worksheet.

12. Run the *resistor* program to assign mutational signature probabilities, filter predicted benign mutations, and assign Pareto ranks.

g. Open a terminal and change into the `osprey3-3/resistor` directory.

h. Download the required Julia dependencies:

   i. Start the Julia interpreter with the following command:

```
> julia --project=.
```

   ii. Activate Julia's package manager by hitting the `']'` key.

   iii. Type `instantiate` and wait while the package manager downloads the dependencies.

   iv. Exist the interpreter by entering CTRL-d or typing `exit()` and hitting enter.

i. Run the program to assign the mutational probabilities and cDNA codons to each mutant sequence:

```
> julia --project=. main.jl --mut-prob {mut-prob-file} \
    --fasta {fasta-file} --identifier {id} \
    --csv-file {csv-file} --pareto-config {pareto-config}
```

where *{mut-prob-file}* is the path to the mutational probabilities file, *{fasta-file}* is the path to the cDNA file, *{id}* is the identifier of the sequence in *{fasta-file}*, *{csv-file}* is the path to the CSV file you created in Step 11, and *{pareto-config}* is the path to the default Pareto optimization configuration JSON, e.g.:

```
> julia --project=. main.jl \
    --mut-prob osprey3-3/resistor/mutational-signatures/melanoma.json \
    --fasta ./mapk1-cdna.fasta --identifier MAPK1 \
    --csv-file erk2-resistor.csv \
    --pareto-config osprey3-3/resistor/pareto-config.json
```

This command:

   i. Fills out the *signature probability* and *codon* columns

ii. Filters rows whose *mutant K* (positive)* is less than 0, as this indicates the loss of function with the endogenous ligand[28].

iii. Filters rows whose *signature probability* is 0, (indicating that the mutant can only occur with 3 base changes).

iv. Computes the cut-off *c*, as defined in Equation 4 in Guerin et al[30].

v. Filters mutants whose ratio of positive to negative K* scores are below the cut-off.

vi. Fills out the *hotspot count* column by counting how many resistance mutations remain at each position after the filtering in the prior steps.

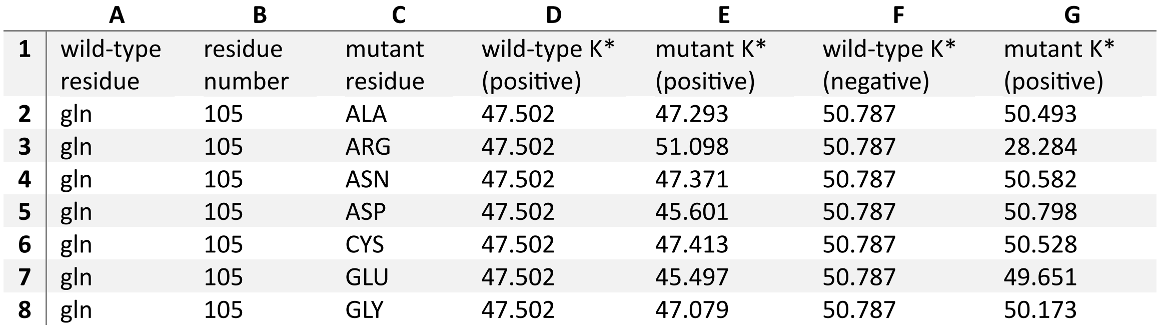
vii. Fills out the *rank* column by running Pareto optimization over the *mutant K* (positive)*, *mutant K* (negative)*, *signature probability*, and *hotspot count* columns.

It outputs the completed table to standard out. You can redirect it to a file using I/O redirection in Linux or by piping the output to the tee command. Table 17 provides an example of the output file.

Note: The Pareto JSON specification file is described in the *README.md*. By default, Resistor optimizes over mutational signature probability, the positive and negative K* scores, and the hotspot score. We've provided a template Pareto JSON specification file in the `resistor` directory, *pareto-config.json*, which specifies to optimize by maximizing a mutant's signature probability, positive design K* score, and hotspot score, and minimizing the mutant's negative design K* score. If you had other criteria to optimize over you could add these to this Pareto JSON specification file.

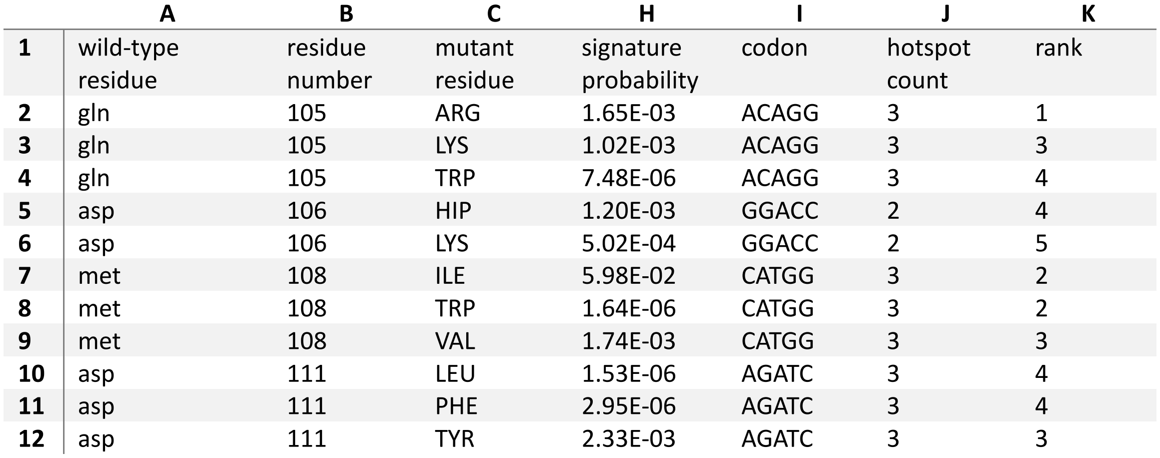
Note: There are two additional optional flags to the command above that may be helpful in some circumstances. These flags are `--debug` and `--c0`. The `--debug` flag prints out intermediary CSV files after each filtering and computational step. It also prints the computed cut-off c to standard error. The `--c0` flag allows you specify a different value for c_0, for more information as to what this value is see Guerin et al[30].

Table 16: A partially completed worksheet for Pareto optimization. The $\log_{10}$ K* scores for the positive designs (ERK2:AMP-PNP) and negative designs (ERK2:SCH7) are put in columns D-G. Put the K* score, for the wild-type sequence, e.g., Q105, bound to the endogenous ligand in column D, and the mutant sequence, e.g., Q105A, bound to the endogenous ligand in column E. In columns F and G do the same for ERK2 bound to SCH7.



	A	B	C	D	E	F	G
1	wild-type residue	residue number	mutant residue	wild-type K* (positive)	mutant K* (positive)	wild-type K* (negative)	mutant K* (positive)
2	gln	105	ALA	47.502	47.293	50.787	50.493
3	gln	105	ARG	47.502	51.098	50.787	28.284
4	gln	105	ASN	47.502	47.371	50.787	50.582
5	gln	105	ASP	47.502	45.601	50.787	50.798
6	gln	105	CYS	47.502	47.413	50.787	50.528
7	gln	105	GLU	47.502	45.497	50.787	49.651
8	gln	105	GLY	47.502	47.079	50.787	50.173

Table 17: The Pareto optimization output file format. Columns H-K are now filled out. Column H contains the computed signature probability, column I the corresponding codon from the cDNA FASTA file, column J the hotspot count, and column K the computed Pareto rank. The $\log_{10}$ K* scores for the positive designs (ERK2:AMP-PNP) and negative designs (ERK2:SCH7) are included in columns D-G but are omitted above due to space constraints.



	A	B	C	H	I	J	K
1	wild-type residue	residue number	mutant residue	signature probability	codon	hotspot count	rank
2	gln	105	ARG	1.65E-03	ACAGG	3	1
3	gln	105	LYS	1.02E-03	ACAGG	3	3
4	gln	105	TRP	7.48E-06	ACAGG	3	4
5	asp	106	HIP	1.20E-03	GGACC	2	4
6	asp	106	LYS	5.02E-04	GGACC	2	5
7	met	108	ILE	5.98E-02	CATGG	3	2
8	met	108	TRP	1.64E-06	CATGG	3	2
9	met	108	VAL	1.74E-03	CATGG	3	3
10	asp	111	LEU	1.53E-06	AGATC	3	4
11	asp	111	PHE	2.95E-06	AGATC	3	4
12	asp	111	TYR	2.33E-03	AGATC	3	3

C.3 Expected outcomes

Resistor provides a protocol for ranking potential resistance mutations to existing and prospective therapeutics. In an earlier publication[30], we used Resistor to successfully predict resistance mutations in BRAF and EGFR. In this example, we have applied Resistor to predicting resistance mutations to SCH7, an ERK1/2 inhibitor.

As an outcome, the predicted resistance mutations, as well as their Pareto ranks, are contained in the file output in Step 12. With that file, one can analyze the predicted change in a mutant's positive K* score and negative K* score, meaning that Resistor produces not only binary predictions of a mutant's resistance or sensitivity profile but also whether a mutation is resistant because of increased binding to the endogenous ligand, decreased binding to the therapeutic, or a combination of the two factors. It also uses a specific cancer type's mutational signature to predict how likely it is that a putative resistance mutation will occur in a specific cancer patient population. Additionally, as mentioned in Step 10, Resistor's use of OSPREY's K* algorithm allows us to output molecular ensembles of low energy conformations for structural analysis. See Figure 24 for an example of the OSPREY-generated low-energy structural ensemble.

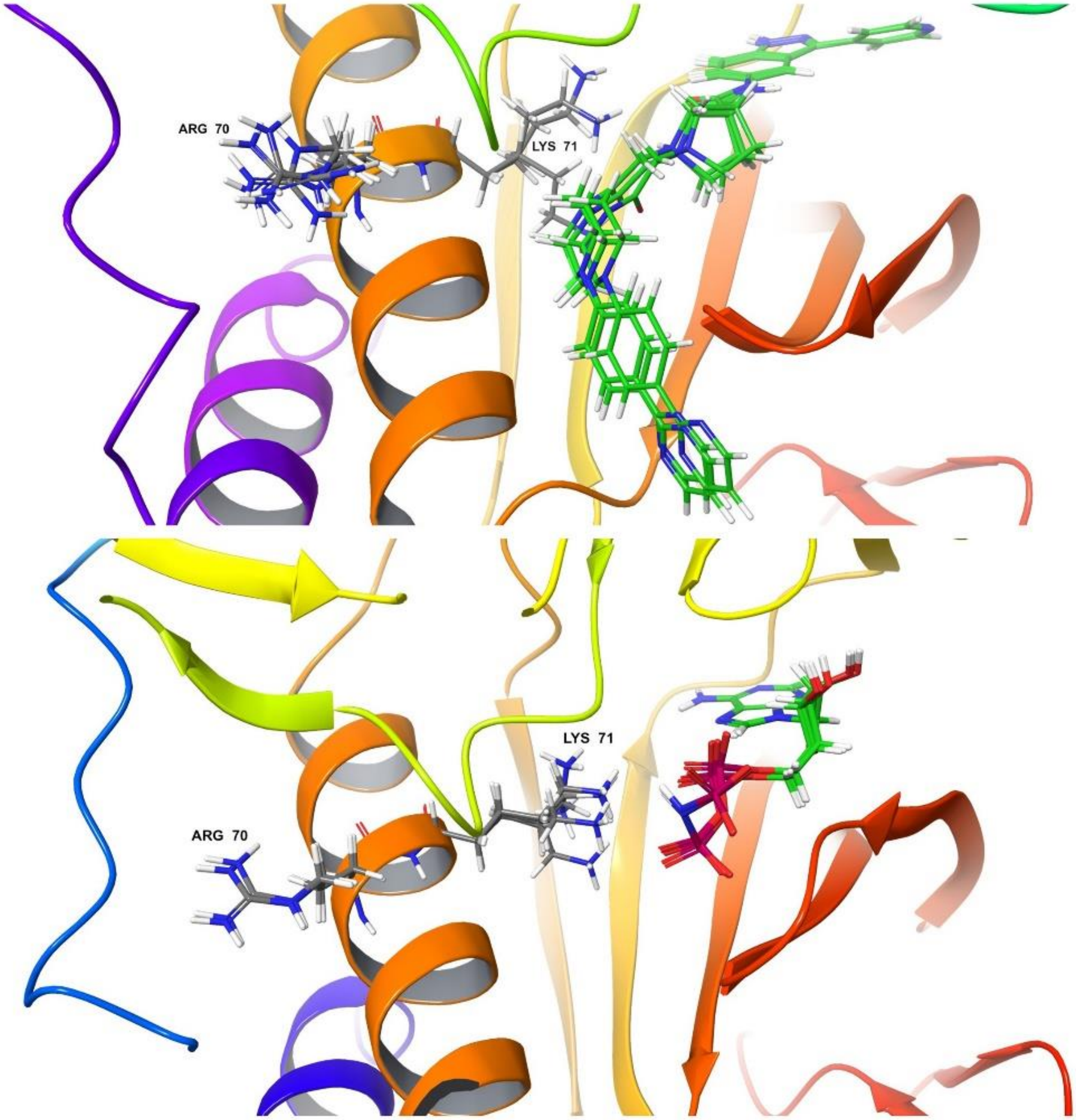


Figure 24: OSPREY-generated structural ensembles of ERK2 E71K. (top) ERK2 E71K with SCH7. R70 and 71K are labeled, and SCH7 is in green. (bottom) ERK2 E71K with AMP-PNP. R70 and 71K are labeled, and AMP-PNP is green and purple. According to Brenan et al.[237], the E71K mutation grants ERK2 resistance to SCH7. Resistor correctly predicts this resistance mutation and ranks it in top Pareto rank.

C.4 Limitations

In the example we provided above for ERK2 and SCH7, we investigated only potential resistance mutations occurring within the binding pocket of the ligands. Modeling allosteric pathways to resistance, for example mutations distant from the binding pocket on the opposite side of ERK2 causing large-scale conformational rearrangement, while a goal of OSPREY, is not something we've yet incorporated into Resistor. Additionally, Resistor does not model resistance caused by phenomena such as splice variants, amplification, or mutations in related genes, which have been shown to be important in N-RAS, MEK1, MEK2, and other genes[123]. Additional modeling to incorporate these causes of resistance is left to future work.

C.5 Troubleshooting

C.5.1 Problem/Potential Solution 1

Problem: You do not see help text when you run the `osprey affinity --help` command.

Potential Solution: There are different potential causes for this problem. If instead of help text you see the following printed out:

```
> osprey affinity --help

ERROR: JAVA_HOME is not set and no 'java' command could be found in
your PATH.

Please set the JAVA_HOME variable in your environment to match the
location of your Java installation.
```

then you have not correctly installed and configured Java 17 as detailed in Before You Begin, Step 1. Redo this step and try again. If the message is:

```
> osprey affinity --help
Error occurred during initialization of boot layer
java.lang.module.FindException: Module jdk.incubator.foreign not
found
```

then it is possible that you are using a version of java that is newer than Java 17. At the current time only Java 17 is supported. It is often the case that there are multiple versions of Java installed in an operating system, and the default version in your operating system may not be Java 17. You can confirm that you are running the correct version of Java for OSPREY by running the command:

```
$JAVA_HOME/bin/java -version
```

Below is a demonstration of the output of that command showing the incorrect version of Java:

```
> $JAVA_HOME/bin/java -version
openjdk version "19" 2022-09-20
OpenJDK Runtime Environment (build 19+36-2238)
OpenJDK 64-Bit Server VM (build 19+36-2238, mixed mode, sharing)
```

The remedy in this case is to ensure that you have downloaded and configured Java 17, as detailed in Before You Begin (C.1.2).

C.5.2 Problem/Potential Solution 2

Problem: When using the `--verify-design` option to the *affinity* command, you see output indicating that indicates a residue was deleted for not having a matching template.

Potential Solution: Note which residue the command says it deleted. The output tells you the atoms that it expects to find. Open the YAML file look at that corresponding residue in either the protein or ligand coordinates. Identify the missing atoms and add them into the structure using a molecular visualization program such as Maestro. If just the labeling is off, fix the labeling. See Movie S8 from Guerin et al.[31] for a demonstration of how to do this.

C.5.3 Problem/Potential Solution 3

Problem: When using the `--verify-design` option to the *affinity* command, you see output indicating that the residue does not exist.

Potential Solution: Look at the coordinates section of the YAML file for the residue mentioned in the error message. Ensure the residue's number and amino acid identifier matches that used in the scan. See Movie S9 from Guerin et al.[31] for a demonstration of this issue and resolution steps.

C.5.4 Problem/Potential Solution 4

Problem: OSPREY fails to parse the design file YAML specification.

Potential Solution: Use a YAML validator, such as yamllint, which can indicate on which line the YAML syntax is broken. Assuming you have installed yamllint as described in Step 3 of Installing the Software Dependencies, default invocation of yamllint would look as follows:

```
> yamllint {design-file}
```

Any errors will be identified with a description of the problem and the line number. Address them as appropriate. Additionally, the official YAML specification[238] is a good resource for learning how YAML documents are written and parsed.

C.5.5 Problem/Potential Solution 5

Problem: The *osprey affinity* command begins to run but after some time fails to complete with an error.

Potential Solution: The most common reason *osprey affinity* fails is that the design has run out of memory. The error output might look like this:

```
edu.duke.cs.osprey.parallelism.TaskExecutor$TaskException: A task
failed, no new tasks can be submitted
at
edu.duke.cs.osprey.parallelism.ConcurrentTaskExecutor.recordException
(ConcurrentTaskExecutor.java:106)
...
```

The important thing to remember is that error stack traces are read from the bottom to the top. Scroll to the bottom of the error and if you see a message that looks like:

```
Caused by: java.lang.OutOfMemoryError: Map failed
  at java.base/sun.nio.ch.FileChannelImpl.map0(Native Method)
  at
java.base/sun.nio.ch.FileChannelImpl.mapInternal(FileChannelImpl.java
:1100)
  ... 18 more
```

The *affinity* command failed because it ran out of memory. There are two potential solutions to try. The first is to increase the amount of memory allocated to OSPREY, if possible. This is defined in the `JAVA_OPTS` environment variable, e.g., to allocate 720 gigabytes to the Java heap, use:

```
export JAVA_OPTS="-Xmx720G -Xms720G"
```

For designs where you run out of memory, the first attempt should be to try to make more memory available to OSPREY. If that is not possible, then the second potential solution is to reduce the number of flexible residues in your design. Oftentimes removing one or two flexible residues will allow a previously difficult design to finish. This should only be done when absolutely necessary, as removing flexible residue can reduce the accuracy of the predictions.

Bibliography

1. May, E., Taylor, K., Gupta, L., and Miranda, W. (2023). Measuring the return from pharmaceutical innovation 2022 (Deloitte LLP).

2. Venter, J.C., Adams, M.D., Myers, E.W., Li, P.W., Mural, R.J., Sutton, G.G., Smith, H.O., Yandell, M., Evans, C.A., Holt, R.A., et al. (2001). The Sequence of the Human Genome. Science *291*, 1304–1351. 10.1126/science.1058040.

3. Leenay, R.T., Aghazadeh, A., Hiatt, J., Tse, D., Roth, T.L., Apathy, R., Shifrut, E., Hultquist, J.F., Krogan, N., Wu, Z., et al. (2019). Large dataset enables prediction of repair after CRISPR–Cas9 editing in primary T cells. Nat Biotechnol *37*, 1034–1037. 10.1038/s41587-019-0203-2.

4. Zhang, H., Zhang, L., Lin, A., Xu, C., Li, Z., Liu, K., Liu, B., Ma, X., Zhao, F., Jiang, H., et al. (2023). Algorithm for Optimized mRNA Design Improves Stability and Immunogenicity. Nature, 1–3. 10.1038/s41586-023-06127-z.

5. Zheng, W., Friedman, A.M., and Bailey-Kellogg, C. (2009). Algorithms for joint optimization of stability and diversity in planning combinatorial libraries of chimeric proteins. Journal of Computational Biology *16*, 1151–1168.

6. Parker, A.S., Choi, Y., Griswold, K.E., and Bailey-Kellogg, C. (2013). Structure-guided deimmunization of therapeutic proteins. Journal of Computational Biology *20*, 152–165.

7. Choi, Y., Griswold, K.E., and Bailey-Kellogg, C. (2013). Structure-based redesign of proteins for minimal T-cell epitope content. Journal of computational chemistry *34*, 879–891.

8. Griswold, K.E., and Bailey-Kellogg, C. (2016). Design and engineering of deimmunized biotherapeutics. Current opinion in structural biology *39*, 79–88.

9. Salvat, R.S., Verma, D., Parker, A.S., Kirsch, J.R., Brooks, S.A., Bailey-Kellogg, C., and Griswold, K.E. (2017). Computationally optimized deimmunization libraries yield highly mutated enzymes with low immunogenicity and enhanced activity. Proceedings of the National Academy of Sciences *114*, E5085–E5093.

10. Verdonk, M.L., Cole, J.C., Hartshorn, M.J., Murray, C.W., and Taylor, R.D. (2003). Improved protein-ligand docking using GOLD. Proteins *52*, 609–623. 10.1002/prot.10465.

11. Trott, O., and Olson, A.J. (2010). AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. Journal of Computational Chemistry *31*, 455–461. 10.1002/jcc.21334.

12. Trellet, M., Melquiond, A.S.J., and Bonvin, A.M.J.J. (2013). A Unified Conformational Selection and Induced Fit Approach to Protein-Peptide Docking. PLOS ONE *8*, e58769. 10.1371/journal.pone.0058769.

13. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. (2021). Highly accurate protein structure prediction with AlphaFold. Nature *596*, 583–589. 10.1038/s41586-021-03819-2.

14. Varadi, M., Anyango, S., Deshpande, M., Nair, S., Natassia, C., Yordanova, G., Yuan, D., Stroe, O., Wood, G., Laydon, A., et al. (2022). AlphaFold Protein Structure Database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. Nucleic Acids Research *50*, D439–D444. 10.1093/nar/gkab1061.

15. Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., and Walter, P. (2002). The Shape and Structure of Proteins. In Molecular Biology of the Cell. 4th edition (Garland Science).

16. Dill, K.A., and Bromberg, S. (2010). Molecular Driving Forces: Statistical Thermodynamics in Biology, Chemistry, Physics, and Nanoscience, 2nd Edition 2nd edition. (Garland Science).

17. Georgiev, I., Lilien, R.H., and Donald, B.R. (2008). The minimized dead-end elimination criterion and its application to protein redesign in a hybrid scoring and search algorithm for computing partition functions over molecular ensembles. Journal of Computational Chemistry *29*, 1527–1542. 10.1002/jcc.20909.

18. Georgiev, I., and Donald, B.R. (2007). Dead-end elimination with backbone flexibility. Bioinformatics *23*, i185–i194.

19. Stevens, B.W., Lilien, R.H., Georgiev, I., Donald, B.R., and Anderson, A.C. (2006). Redesigning the PheA Domain of Gramicidin Synthetase Leads to a New Understanding of the Enzyme's Mechanism and Selectivity. Biochemistry *45*, 15495–15504. 10.1021/bi061788m.

20. Chen, C.-Y., Georgiev, I., Anderson, A.C., and Donald, B.R. (2009). Computational structure-based redesign of enzyme activity. Proceedings of the National Academy of Sciences *106*, 3764–3769. 10.1073/pnas.0900266106.

21. Lilien, R.H., Stevens, B.W., Anderson, A.C., and Donald, B.R. (2005). A novel ensemble-based scoring and search algorithm for protein redesign and its application to modify the substrate specificity of the gramicidin synthetase a phenylalanine adenylation enzyme. Journal of Computational Biology *12*, 740–761.

22. Holt, G.T., Jou, J.D., Gill, N.P., Lowegard, A.U., Martin, J.W., Madden, D.R., and Donald, B.R. (2019). Computational Analysis of Energy Landscapes Reveals

Dynamic Features That Contribute to Binding of Inhibitors to CFTR-Associated Ligand. J. Phys. Chem. B *123*, 10441–10455. 10.1021/acs.jpcb.9b07278.

23. Roberts, K.E., Cushing, P.R., Boisguerin, P., Madden, D.R., and Donald, B.R. (2012). Computational Design of a PDZ Domain Peptide Inhibitor that Rescues CFTR Activity. PLoS Comput Biol *8*, e1002477. 10.1371/journal.pcbi.1002477.

24. Lowegard, A.U., Frenkel, M.S., Holt, G.T., Jou, J.D., Ojewole, A.A., and Donald, B.R. (2020). Novel, provable algorithms for efficient ensemble-based computational protein design and their application to the redesign of the c-Raf-RBD:KRas protein-protein interface. PLOS Computational Biology *16*, e1007447. 10.1371/journal.pcbi.1007447.

25. Reeve, S.M., Si, D., Krucinska, J., Yan, Y., Viswanathan, K., Wang, S., Holt, G.T., Frenkel, M.S., Ojewole, A.A., Estrada, A., et al. (2019). Toward Broad Spectrum Dihydrofolate Reductase Inhibitors Targeting Trimethoprim Resistant Enzymes Identified in Clinical Isolates of Methicillin Resistant Staphylococcus aureus. ACS Infect. Dis. *5*, 1896–1906. 10.1021/acsinfecdis.9b00222.

26. Wang, S., Reeve, S.M., Holt, G.T., Ojewole, A.A., Frenkel, M.S., Gainza, P., Keshipeddy, S., Fowler, V.G., Wright, D.L., and Donald, B.R. (2022). Chiral evasion and stereospecific antifolate resistance in Staphylococcus aureus. PLoS computational biology *18*, e1009855. 10.1371/journal.pcbi.1009855.

27. Ojewole, A., Lowegard, A., Gainza, P., Reeve, S.M., Georgiev, I., Anderson, A.C., and Donald, B.R. (2017). OSPREY Predicts Resistance Mutations Using Positive and Negative Computational Protein Design. In Computational Protein Design Methods in Molecular Biology., I. Samish, ed. (Springer), pp. 291–306. 10.1007/978-1-4939-6637-0_15.

28. Kaserer, T., and Blagg, J. (2018). Combining mutational signatures, clonal fitness, and drug affinity to define drug-specific resistance mutations in cancer. Cell Chemical Biology *25*, 1359-1371.e2. 10.1016/j.chembiol.2018.07.013.

29. Reeve, S.M., Gainza, P., Frey, K.M., Georgiev, I., Donald, B.R., and Anderson, A.C. (2015). Protein design algorithms predict viable resistance to an experimental antifolate. Proceedings of the National Academy of Sciences *112*, 749–754.

30. Guerin, N., Feichtner, A., Stefan, E., Kaserer, T., and Donald, B.R. (2022). Resistor: an algorithm for predicting resistance mutations via Pareto optimization over multistate protein design and mutational signatures. Cell Systems *13*, 830-843.e3. 10.1016/j.cels.2022.09.003.

31. Guerin, N., Kaserer, T., and Donald, B.R. (2023). Protocol for predicting drug-resistant protein mutations to an ERK2 inhibitor using RESISTOR. STAR Protocols *4*, 102170. 10.1016/j.xpro.2023.102170.

32. Chuang, G.-Y., Geng, H., Pancera, M., Xu, K., Cheng, C., Acharya, P., Chambers, M., Druz, A., Tsybovsky, Y., Wanninger, T.G., et al. (2017). Structure-Based Design of a Soluble Prefusion-Closed HIV-1 Env Trimer with Reduced CD4 Affinity and Improved Immunogenicity. J Virol *91*, e02268-16. 10.1128/JVI.02268-16.

33. Georgiev, I.S., Rudicell, R.S., Saunders, K.O., Shi, W., Kirys, T., McKee, K., O’Dell, S., Chuang, G.-Y., Yang, Z.-Y., Ofek, G., et al. (2014). Antibodies VRC01 and 10E8 Neutralize HIV-1 with High Breadth and Potency Even with Ig-Framework Regions Substantially Reverted to Germline. The Journal of Immunology *192*, 1100–1106. 10.4049/jimmunol.1302515.

34. Rudicell, R.S., Kwon, Y.D., Ko, S.-Y., Pegu, A., Louder, M.K., Georgiev, I.S., Wu, X., Zhu, J., Boyington, J.C., Chen, X., et al. (2014). Enhanced potency of a broadly neutralizing HIV-1 antibody in vitro improves protection against lentiviral infection in vivo. J Virol *88*, 12669–12682. 10.1128/JVI.02213-14.

35. Hallen, M.A., Martin, J.W., Ojewole, A., Jou, J.D., Lowegard, A.U., Frenkel, M.S., Gainza, P., Nisonoff, H.M., Mukund, A., Wang, S., et al. (2018). OSPREY 3.0: Open-source protein redesign for you, with powerful new features. Journal of Computational Chemistry *39*, 2494–2507. 10.1002/jcc.25522.

36. Hallen, M.A., and Donald, B.R. (2016). COMETS (Constrained Optimization of Multistate Energies by Tree Search): A provable and efficient protein design algorithm to optimize binding affinity and specificity with respect to sequence. Journal of Computational Biology *23*, 311–321.

37. Jou, J.D., Holt, G.T., Lowegard, A.U., and others (2020). Minimization-Aware Recursive K*: A Novel, Provable Algorithm that Accelerates Ensemble-Based Protein Design and Provably Approximates the Energy Landscape. Journal of Computational Biology *27*, 550–564. 10.1089/cmb.2019.0315.

38. Ojewole, A.A., Jou, J.D., Fowler, V.G., and Donald, B.R. (2018). BBK* (Branch and Bound Over K*): a provable and efficient ensemble-based protein design algorithm to optimize stability and binding affinity over large sequence spaces. Journal of Computational Biology *25*, 726–739. 10.1089/cmb.2017.0267.

39. Simoncini, D., Allouche, D., de Givry, S., Delmas, C., Barbe, S., and Schiex, T. (2015). Guaranteed discrete energy optimization on large protein design problems. Journal of chemical theory and computation *11*, 5980–5989.

40. Pierce, N.A., and Winfree, E. (2002). Protein Design is NP-hard. Protein Engineering, Design and Selection *15*, 779–782. 10.1093/protein/15.10.779.

41. Kingsford, C.L., Chazelle, B., and Singh, M. (2005). Solving and analyzing side-chain positioning problems using linear and integer programming. Bioinformatics *21*, 1028–1039. 10.1093/bioinformatics/bti144.

42. Ruffini, M. (2021). Models and Algorithms for Computational Protein Design.

43. Nisonoff, H. (2015). Efficient Partition Function Estimation in Computational Protein Design: Probabalistic Guarantees and Characterization of a Novel Algorithm.

44. Desmet, J., Maeyer, M.D., Hazes, B., and Lasters, I. (1992). The dead-end elimination theorem and its use in protein side-chain positioning. Nature *356*, 539–542. 10.1038/356539a0.

45. Leach, A.R., and Lemon, A.P. (1998). Exploring the conformational space of protein side chains using dead-end elimination and the A* algorithm. Proteins: Structure, Function, and Bioinformatics *33*, 227–239. 10.1002/(SICI)1097-0134(19981101)33:2<227::AID-PROT7>3.0.CO;2-F.

46. Donald, B.R. (2011). Algorithms in Structural Molecular Biology (MIT Press).

47. Gainza, P., Roberts, K.E., and Donald, B.R. (2012). Protein design using continuous rotamers. PLoS computational biology *8*, e1002335. 10.1371/journal.pcbi.1002335.

48. Hart, P.E., Nilsson, N.J., and Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. IEEE Transactions on Systems Science and Cybernetics *4*, 100–107. 10.1109/TSSC.1968.300136.

49. Lovell, S.C., Word, J.M., Richardson, J.S., and Richardson, D.C. (2000). The penultimate rotamer library. Proteins *40*, 389–408.

50. Hallen, M.A., Keedy, D.A., and Donald, B.R. (2013). Dead-end elimination with perturbations (DEEPer): A provable protein design algorithm with continuous sidechain and backbone flexibility. Proteins: Structure, Function, and Bioinformatics *81*, 18–39.

51. Craik, D.J., Fairlie, D.P., Liras, S., and Price, D. (2013). The Future of Peptide-based Drugs. Chemical Biology & Drug Design *81*, 136–147. 10.1111/cbdd.12055.

52. Fosgerau, K., and Hoffmann, T. (2015). Peptide therapeutics: current status and future directions. Drug Discovery Today *20*, 122–128. 10.1016/j.drudis.2014.10.003.

53. Di, L. (2014). Strategic Approaches to Optimizing Peptide ADME Properties. AAPS J *17*, 134–143. 10.1208/s12248-014-9687-3.

54. Chen, S., Gfeller, D., Buth, S.A., Michielin, O., Leiman, P.G., and Heinis, C. (2013). Improving Binding Affinity and Stability of Peptide Ligands by Substituting Glycines with D-Amino Acids. ChemBioChem *14*, 1316–1322. 10.1002/cbic.201300228.

55. Angelini, A., Cendron, L., Chen, S., Touati, J., Winter, G., Zanotti, G., and Heinis, C. (2012). Bicyclic Peptide Inhibitor Reveals Large Contact Interface with a Protease Target. ACS Chem. Biol. *7*, 817–821. 10.1021/cb200478t.

56. Haugaard-Kedström, L.M., Clemmensen, L.S., Sereikaite, V., Jin, Z., Fernandes, E.F.A., Wind, B., Abalde-Gil, F., Daberger, J., Vistrup-Parry, M., Aguilar-Morante, D., et al. (2021). A High-Affinity Peptide Ligand Targeting Syntenin Inhibits Glioblastoma. J. Med. Chem. *64*, 1423–1434. 10.1021/acs.jmedchem.0c00382.

57. Zhou, J., and Grigoryan, G. (2015). Rapid search for tertiary fragments reveals protein sequence–structure relationships. Protein Sci *24*, 508–524. 10.1002/pro.2610.

58. Guerin, N., Kaserer, T., and Donald, B.R. (2022). Resistor: An Algorithm for Predicting Resistance Mutations Using Pareto Optimization over Multistate Protein Design and Mutational Signatures. In, I. Pe’er, ed. (Springer International Publishing), pp. 387–389.

59. Guerin, N., Kaserer, T., and Donald, B.R. (2022). RESISTOR: A new OSPREY module to predict resistance mutations. Journal of Computational Biology *29*, 1346–1352. 10.1089/cmb.2022.0254.

60. Röck, R., Mayrhofer, J.E., Torres-Quesada, O., Enzler, F., Raffeiner, A., Raffeiner, P., Feichtner, A., Huber, R.G., Koide, S., Taylor, S.S., et al. (2019). BRAF inhibitors promote intermediate BRAF (V600E) conformations and binary interactions with activated RAS. Science advances *5*, eaav8463.

61. Mayrhofer, J.E., Enzler, F., Feichtner, A., Röck, R., Fleischmann, J., Raffeiner, A., Tschaikner, P., Ogris, E., Huber, R.G., Hartl, M., et al. (2020). Mutation-oriented profiling of autoinhibitory kinase conformations predicts RAF inhibitor efficacies. Proceedings of the National Academy of Sciences *117*, 31105–31113.

62. Wagenaar, T.R., Ma, L., Roscoe, B., Park, S.M., Bolon, D.N., and Green, M.R. (2014). Resistance to vemurafenib resulting from a novel mutation in the BRAFV 600 E kinase domain. Pigment cell & melanoma research *27*, 124–133.

63. Centers for Disease Control and Prevention (2020). Antibiotic / Antimicrobial Resistance.

64. Housman, G., Byler, S., Heerboth, S., and others (2014). Drug resistance in cancer: an overview. Cancers *6*, 1769–1792. 10.3390/cancers6031769.

65. Zahreddine, H., and Borden, K. (2013). Mechanisms and insights into drug resistance in cancer. Frontiers in pharmacology *4*, 28.

66. Assaraf, Y.G., Brozovic, A., Goncalves, A.C., and others (2019). The multi-factorial nature of clinical multidrug resistance in cancer. Drug Resistance Updates *46*, 100645. 10.1016/j.drup.2019.100645.

67. Gupta, R.K., Jordan, M.R., Sultan, B.J., Hill, A., Davis, D.H., Gregson, J., Sawyer, A.W., Hamers, R.L., Ndembi, N., Pillay, D., et al. (2012). Global trends in antiretroviral resistance in treatment-naive individuals with HIV after rollout of antiretroviral treatment in resource-limited settings: a global collaborative study and meta-regression analysis. The Lancet *380*, 1250–1258.

68. Altman, M.D., Ali, A., Kumar Reddy, G.K., Nalam, M.N., Anjum, S.G., Cao, H., Chellappan, S., Kairys, V., Fernandes, M.X., Gilson, M.K., et al. (2008). HIV-1 protease inhibitors from inverse design in the substrate envelope exhibit subnanomolar binding to drug-resistant variants. Journal of the American Chemical Society *130*, 6099–6113.

69. Prabu-Jeyabalan, M., Nalivaika, E., and Schiffer, C.A. (2002). Substrate shape determines specificity of recognition for HIV-1 protease: analysis of crystal structures of six substrate complexes. Structure *10*, 369–381.

70. King, N.M., Prabu-Jeyabalan, M., Nalivaika, E.A., Wigerinck, P., De Béthune, M.-P., and Schiffer, C.A. (2004). Structural and thermodynamic basis for the binding of TMC114, a next-generation human immunodeficiency virus type 1 protease inhibitor. Journal of virology *78*, 12012–12021.

71. Shen, Y., Altman, M.D., Ali, A., Nalam, M.N., Cao, H., Rana, T.M., Schiffer, C.A., and Tidor, B. (2013). Testing the substrate-envelope hypothesis with designed pairs of compounds. ACS chemical biology *8*, 2433–2441.

72. Frey, K.M., Georgiev, I., Donald, B.R., and Anderson, A.C. (2010). Predicting resistance mutations using protein design algorithms. Proceedings of the National Academy of Sciences *107*, 13707–13712.

73. Gainza, P., Nisonoff, H.M., and Donald, B.R. (2016). Algorithms for protein design. Current opinion in structural biology *39*, 16–26.

74. Yan, X.-E., Zhu, S.-J., Liang, L., Zhao, P., Choi, H.G., and Yun, C.-H. (2017). Structural basis of mutant-selectivity and drug-resistance related to CO-1686. Oncotarget *8*, 53508.

75. Yun, C.-H., Mengwasser, K.E., Toms, A.V., Woo, M.S., Greulich, H., Wong, K.-K., Meyerson, M., and Eck, M.J. (2008). The T790M mutation in EGFR kinase causes drug resistance by increasing the affinity for ATP. Proceedings of the National Academy of Sciences *105*, 2070–2075.

76. Yoshikawa, S., Kukimoto-Niino, M., Parker, L., Handa, N., Terada, T., Fujimoto, T., Terazawa, Y., Wakiyama, M., Sato, M., Sano, S., et al. (2013). Structural basis for the altered drug sensitivities of non-small cell lung cancer-associated mutants of human epidermal growth factor receptor. Oncogene *32*, 27–38.

77. Reeve, S.M., Scocchera, E.W., Narendran, G., Keshipeddy, S., Krucinska, J., Hajian, B., Ferreira, J., Nailor, M., Aeschlimann, J., Wright, D.L., et al. (2016). MRSA isolates from united states hospitals carry dfrg and dfrk resistance genes and succumb to propargyl-linked antifolates. Cell chemical biology *23*, 1458–1467.

78. Choi, Y., Ndong, C., Griswold, K.E., and Bailey-Kellogg, C. (2016). Computationally driven antibody engineering enables simultaneous humanization and thermostabilization. Protein Engineering, Design and Selection *29*, 419–426.

79. Salvat, R.S., Parker, A.S., Choi, Y., Bailey-Kellogg, C., and Griswold, K.E. (2015). Mapping the Pareto optimal design space for a functionally deimmunized biotherapeutic candidate. PLoS Comput Biol *11*, e1003988.

80. He, L., Friedman, A.M., and Bailey-Kellogg, C. (2012). A divide-and-conquer approach to determine the Pareto frontier for optimization of protein engineering experiments. Proteins: Structure, Function, and Bioinformatics *80*, 790–806.

81. Alexandrov, L.B., Nik-Zainal, S., Wedge, D.C., Aparicio, S.A., Behjati, S., Biankin, A.V., Bignell, G.R., Bolli, N., Borg, A., Børresen-Dale, A.-L., et al. (2013). Signatures of mutational processes in human cancer. Nature *500*, 415–421.

82. Alexandrov, L.B., Kim, J., Haradhvala, N.J., Huang, M.N., Ng, A.W.T., Wu, Y., Boot, A., Covington, K.R., Gordenin, D.A., Bergstrom, E.N., et al. (2020). The repertoire of mutational signatures in human cancer. Nature *578*, 94–101.

83. Qi, Y., Martin, J.W., Barb, A.W., Thélot, F., Yan, A.K., Donald, B.R., and Oas, T.G. (2018). Continuous interdomain orientation distributions reveal components of binding thermodynamics. Journal of molecular biology *430*, 3412–3426.

84. Harrison, P.T., Vyse, S., and Huang, P.H. (2020). Rare epidermal growth factor receptor (EGFR) mutations in non-small cell lung cancer. In Seminars in cancer biology (Elsevier), pp. 167–179.

85. Lynch, T.J., Bell, D.W., Sordella, R., Gurubhagavatula, S., Okimoto, R.A., Brannigan, B.W., Harris, P.L., Haserlat, S.M., Supko, J.G., Haluska, F.G., et al. (2004). Activating mutations in the epidermal growth factor receptor underlying responsiveness of non–small-cell lung cancer to gefitinib. New England Journal of Medicine *350*, 2129–2139.

86. Davies, H., Bignell, G.R., Cox, C., Stephens, P., Edkins, S., Clegg, S., Teague, J., Woffendin, H., Garnett, M.J., Bottomley, W., et al. (2002). Mutations of the BRAF gene in human cancer. Nature *417*, 949–954.

87. Dowell, J., Minna, J.D., and Kirkpatrick, P. (2005). Erlotinib hydrochloride. Nature Reviews Drug Discovery *4*.

88. Herbst, R.S., Fukuoka, M., and Baselga, J. (2004). Gefitinib—a novel targeted approach to treating cancer. Nature Reviews Cancer *4*, 956–965.

89. Soria, J.-C., Ohe, Y., Vansteenkiste, J., Reungwetwattana, T., Chewaskulyong, B., Lee, K.H., Dechaphunkul, A., Imamura, F., Nogami, N., Kurata, T., et al. (2018). Osimertinib in untreated EGFR-mutated advanced non–small-cell lung cancer. New England journal of medicine *378*, 113–125.

90. Ballantyne, A.D., and Garnock-Jones, K.P. (2013). Dabrafenib: first global approval. Drugs *73*, 1367–1376.

91. Bollag, G., Tsai, J., Zhang, J., Zhang, C., Ibrahim, P., Nolop, K., and Hirth, P. (2012). Vemurafenib: the first drug approved for BRAF-mutant cancer. Nature reviews Drug discovery *11*, 873–886.

92. Shirley, M. (2018). Encorafenib and binimetinib: first global approvals. Drugs *78*, 1277–1284.

93. Janku, F., Sherman, E., Parikh, A., Feun, L., Tsai, F., Allen, E., Zhang, C., Severson, P., Inokuchi, K., Walling, J., et al. (2020). Interim results from a phase 1/2 precision medicine study of PLX8394-a next generation BRAF inhibitor. European Journal of Cancer *138*, S2–S3.

94. Yun, C.-H., Boggon, T.J., Li, Y., Woo, M.S., Greulich, H., Meyerson, M., and Eck, M.J. (2007). Structures of lung cancer-derived EGFR mutants and inhibitor complexes: mechanism of activation and insights into differential inhibitor sensitivity. Cancer cell *11*, 217–227.

95. Stamos, J., Sliwkowski, M.X., and Eigenbrot, C. (2002). Structure of the epidermal growth factor receptor kinase domain alone and in complex with a 4-anilinoquinazoline inhibitor. Journal of Biological Chemistry *277*, 46265–46272.

96. Yosaatmadja, Y., Squire, C., McKeage, C., and Flanagan, M. (2014). 1.85 angstrom structure of EGFR kinase domain with gefitinib. To Be Published.

97. Yosaatmadja, Y., Silva, S., Dickson, J.M., Patterson, A.V., Smaill, J.B., Flanagan, J.U., McKeage, M.J., and Squire, C.J. (2015). Binding mode of the breakthrough inhibitor AZD9291 to epidermal growth factor receptor revealed. Journal of structural biology *192*, 539–544.

98. Zhang, C., Spevak, W., Zhang, Y., Burton, E.A., Ma, Y., Habets, G., Zhang, J., Lin, J., Ewing, T., Matusow, B., et al. (2015). RAF inhibitors that evade paradoxical MAPK pathway activation. Nature *526*, 583–586.

99. Hodis, E., Watson, I.R., Kryukov, G.V., Arold, S.T., Imielinski, M., Theurillat, J.-P., Nickerson, E., Auclair, D., Li, L., Place, C., et al. (2012). A landscape of driver mutations in melanoma. Cell *150*, 251–263.

100. Yu, H.A., Arcila, M.E., Rekhtman, N., Sima, C.S., Zakowski, M.F., Pao, W., Kris, M.G., Miller, V.A., Ladanyi, M., and Riely, G.J. (2013). Analysis of tumor specimens at the time of acquired resistance to EGFR-TKI therapy in 155 patients with EGFR-mutant lung cancers. Clinical cancer research *19*, 2240–2247.

101. Avizienyte, E., Ward, R.A., and Garner, A.P. (2008). Comparison of the EGFR resistance mutation profiles generated by EGFR-targeted tyrosine kinase inhibitors and the impact of drug combinations. Biochemical Journal *415*, 197–206.

102. Chen, K., Zhou, F., Shen, W., Jiang, T., Wu, X., Tong, X., Shao, Y.W., Qin, S., and Zhou, C. (2017). Novel Mutations on EGFR Leu792 Potentially Correlate to Acquired Resistance to Osimertinib in Advanced NSCLC. Journal of Thoracic Oncology *12*, e65–e68. 10.1016/j.jtho.2016.12.024.

103. Yang, Z., Yang, N., Ou, Q., Xiang, Y., Jiang, T., Wu, X., Bao, H., Tong, X., Wang, X., Shao, Y.W., et al. (2018). Investigating novel resistance mechanisms to third-generation EGFR tyrosine kinase inhibitor osimertinib in non–small cell lung cancer patients. Clinical Cancer Research *24*, 3097–3107.

104. Ou, S.-H.I., Cui, J., Schrock, A.B., Goldberg, M.E., Zhu, V.W., Albacker, L., Stephens, P.J., Miller, V.A., and Ali, S.M. (2017). Emergence of novel and dominant acquired EGFR solvent-front mutations at Gly796 (G796S/R) together with C797S/G and L792F/H mutations in one EGFR (L858R/T790M) NSCLC patient who progressed on osimertinib. Lung Cancer *108*, 228–231.

105. Fairclough, S.R., Kiedrowski, L.A., Lin, J.J., Zelichov, O., Tarcic, G., Stinchcombe, T.E., Odegaard, J.I., Lanman, R.B., Shaw, A.T., and Nagy, R.J. (2019). Identification of osimertinib-resistant EGFR L792 mutations by cfDNA sequencing: oncogenic activity assessment and prevalence in large cfDNA cohort. Experimental hematology & oncology *8*, 1–6.

106. Li, D., Yang, D., Cui, S., Pan, E., Yang, P., and Dai, Z. (2021). NGS-Based ctDNA Profiling After the Resistance of Second-Line Osimertinib for Patient with EGFR-Mutated Pulmonary Adenocarcinoma. OncoTargets and therapy *14*, 4261.

107. Zheng, D., Hu, M., Bai, Y., Zhu, X., Lu, X., Wu, C., Wang, J., Liu, L., Wang, Z., Ni, J., et al. (2017). EGFR G796D mutation mediates resistance to osimertinib. Oncotarget *8*, 49671.

108. Tate, J.G., Bamford, S., Jubb, H.C., Sondka, Z., Beare, D.M., Bindal, N., Boutselakis, H., Cole, C.G., Creatore, C., Dawson, E., et al. (2019). COSMIC: the Catalogue Of Somatic Mutations In Cancer. Nucleic Acids Research *47*, D941–D947. 10.1093/nar/gky1015.

109. Thress, K.S., Paweletz, C.P., Felip, E., Cho, B.C., Stetson, D., Dougherty, B., Lai, Z., Markovets, A., Vivancos, A., Kuang, Y., et al. (2015). Acquired EGFR C797S

mutation mediates resistance to AZD9291 in non–small cell lung cancer harboring EGFR T790M. Nature medicine *21*, 560–562.

110. Arulananda, S., Do, H., Musafer, A., Mitchell, P., Dobrovic, A., and John, T. (2017). Combination Osimertinib and Gefitinib in C797S and T790M EGFR-Mutated Non–Small Cell Lung Cancer. Journal of Thoracic Oncology *12*, 1728–1732. 10.1016/j.jtho.2017.08.006.

111. Enzler, F., Tschaikner, P., Schneider, R., and Stefan, E. (2020). KinCon: cell-based recording of full-length kinase conformations. IUBMB life *72*, 1168–1174.

112. Sen, B., Peng, S., Tang, X., Erickson, H.S., Galindo, H., Mazumdar, T., Stewart, D.J., Wistuba, I., and Johnson, F.M. (2012). Kinase-impaired BRAF mutations in lung cancer confer sensitivity to dasatinib. Science translational medicine *4*, 136ra70.

113. Zheng, G., Tseng, L.-H., Chen, G., Haley, L., Illei, P., Gocke, C.D., Eshleman, J.R., and Lin, M.-T. (2015). Clinical detection and categorization of uncommon and concomitant mutations involving BRAF. BMC cancer *15*, 1–10.

114. Cerami, E., Gao, J., Dogrusoz, U., Gross, B.E., Sumer, S.O., Aksoy, B.A., Jacobsen, A., Byrne, C.J., Heuer, M.L., Larsson, E., et al. (2012). The cBio cancer genomics portal: an open platform for exploring multidimensional cancer genomics data. Cancer discovery *2*, 401–404.

115. Gao, J., Aksoy, B.A., Dogrusoz, U., Dresdner, G., Gross, B., Sumer, S.O., Sun, Y., Jacobsen, A., Sinha, R., Larsson, E., et al. (2013). Integrative analysis of complex cancer genomics and clinical profiles using the cBioPortal. Science signaling *6*, pl1.

116. Valiant, L.G. (1979). The complexity of computing the permanent. Theoretical computer science *8*, 189–201.

117. Viricel, C., Simoncini, D., Barbe, S., and Schiex, T. (2016). Guaranteed weighted counting for affinity computation: Beyond determinism and structure. In International Conference on Principles and Practice of Constraint Programming (Springer), pp. 733–750.

118. Jou, J.D., Jain, S., Georgiev, I.S., and others (2016). BWM*: A novel, provable, ensemble-based dynamic programming algorithm for sparse approximations of computational protein design. Journal of Computational Biology *23*, 413–424. 10.1089/cmb.2015.0194.

119. Lilien, R.H., Farid, H., and Donald, B.R. (2003). Probabilistic disease classification of expression-dependent proteomic data from mass spectrometry of human serum. Journal of computational biology *10*, 925–946.

120. Lyczek, A., Berger, B.-T., Rangwala, A.M., Paung, Y., Tom, J., Philipose, H., Guo, J., Albanese, S.K., Robers, M.B., Knapp, S., et al. (2021). Mutation in Abl kinase

with altered drug-binding kinetics indicates a novel mechanism of imatinib resistance. Proceedings of the National Academy of Sciences *118*, e2111451118.

121. Gorczynski, M.J., Grembecka, J., Zhou, Y., Kong, Y., Roudaia, L., Douvas, M.G., Newman, M., Bielnicka, I., Baber, G., Corpora, T., et al. (2007). Allosteric inhibition of the protein-protein interaction between the leukemia-associated proteins Runx1 and CBFβ. Chemistry & biology *14*, 1186–1197.

122. Sierra, J.R., Cepero, V., and Giordano, S. (2010). Molecular mechanisms of acquired resistance to tyrosine kinase targeted therapy. Molecular cancer *9*, 1–13.

123. Rizos, H., Menzies, A.M., Pupo, G.M., Carlino, M.S., Fung, C., Hyman, J., Haydu, L.E., Mijatov, B., Becker, T.M., Boyd, S.C., et al. (2014). BRAF inhibitor resistance mechanisms in metastatic melanoma: spectrum and clinical impact. Clinical Cancer Research *20*, 1965–1977. 10.1158/1078-0432.CCR-13-3122.

124. Martin, J., and Donald, B.R. Conformation Energy :: OSPREY. OSPREY Documentation. https://www2.cs.duke.edu/donaldlab/software/osprey/docs/contributing/architecture/conf-energy/.

125. Banting, F. Frederick G. Banting - Nobel Lecture. Nobel Prize Outreach AB 2023. https://www.nobelprize.org/prizes/medicine/1923/banting/lecture/.

126. Wang, L., Wang, N., Zhang, W., Cheng, X., Yan, Z., Shao, G., Wang, X., Wang, R., and Fu, C. (2022). Therapeutic peptides: current applications and future directions. Signal Transduct Target Ther *7*, 48. 10.1038/s41392-022-00904-4.

127. Liu, M., Li, C., Pazgier, M., Li, C., Mao, Y., Lv, Y., Gu, B., Wei, G., Yuan, W., Zhan, C., et al. (2010). D-peptide inhibitors of the p53-MDM2 interaction for targeted molecular therapy of malignant neoplasms. Proc Natl Acad Sci U S A *107*, 14321–14326. 10.1073/pnas.1008930107.

128. Lander, A.J., Jin, Y., and Luk, L.Y.P. (2023). D-Peptide and D-Protein Technology: Recent Advances, Challenges, and Opportunities. ChemBioChem *24*, e202200537. 10.1002/cbic.202200537.

129. Amacher, J.F., Brooks, L., Hampton, T.H., and Madden, D.R. (2020). Specificity in PDZ-peptide interaction networks: Computational analysis and review. Journal of Structural Biology: X *4*, 100022. 10.1016/j.yjsbx.2020.100022.

130. Lee, H.-J., and Zheng, J.J. (2010). PDZ domains and their binding partners: structure, specificity, and modification. Cell Communication and Signaling *8*, 8. 10.1186/1478-811X-8-8.

131. Ivarsson, Y. (2012). Plasticity of PDZ domains in ligand recognition and signaling. FEBS Lett *586*, 2638–2647. 10.1016/j.febslet.2012.04.015.

132. Jemth, P., and Gianni, S. (2007). PDZ Domains: Folding and Binding. Biochemistry *46*, 8701–8708. 10.1021/bi7008618.

133. Harris, B.Z., and Lim, W.A. (2001). Mechanism and role of PDZ domains in signaling complex assembly. Journal of Cell Science *114*, 3219–3231. 10.1242/jcs.114.18.3219.

134. Christensen, N.R., Čalyševa, J., Fernandes, E.F.A., Lüchow, S., Clemmensen, L.S., Haugaard-Kedström, L.M., and Strømgaard, K. (2019). PDZ Domains as Drug Targets. Adv Ther (Weinh) *2*, 1800143. 10.1002/adtp.201800143.

135. Davey, N.E., Travé, G., and Gibson, T.J. (2011). How viruses hijack cell regulation. Trends in Biochemical Sciences *36*, 159–169. 10.1016/j.tibs.2010.10.002.

136. Panel, N., Villa, F., Opuu, V., Mignon, D., and Simonson, T. (2021). Computational Design of PDZ-Peptide Binding. Methods Mol Biol *2256*, 237–255. 10.1007/978-1-0716-1166-1_14.

137. Mignon, D., Panel, N., Chen, X., Fuentes, E.J., and Simonson, T. (2017). Computational Design of the Tiam1 PDZ Domain and Its Ligand Binding. J Chem Theory Comput *13*, 2271–2289. 10.1021/acs.jctc.6b01255.

138. Nakariyakul, S., Liu, Z.-P., and Chen, L. (2014). A sequence-based computational approach to predicting PDZ domain-peptide interactions. Biochim Biophys Acta *1844*, 165–170. 10.1016/j.bbapap.2013.04.008.

139. Opuu, V., Sun, Y.J., Hou, T., Panel, N., Fuentes, E.J., and Simonson, T. (2020). A physics-based energy function allows the computational redesign of a PDZ domain. Sci Rep *10*, 11150. 10.1038/s41598-020-67972-w.

140. Zheng, F., Jewell, H., Fitzpatrick, J., Zhang, J., Mierke, D.F., and Grigoryan, G. (2015). Computational design of selective peptides to discriminate between similar PDZ domains in an oncogenic pathway. J Mol Biol *427*, 491–510. 10.1016/j.jmb.2014.10.014.

141. Smith, C.A., Shi, C.A., Chroust, M.K., Bliska, T.E., Kelly, M.J.S., Jacobson, M.P., and Kortemme, T. (2013). Design of a phosphorylatable PDZ domain with peptide-specific affinity changes. Structure *21*, 54–64. 10.1016/j.str.2012.10.007.

142. Smith, C.A., and Kortemme, T. (2010). Structure-based prediction of the peptide sequence space recognized by natural and synthetic PDZ domains. J Mol Biol *402*, 460–474. 10.1016/j.jmb.2010.07.032.

143. Melero, C., Ollikainen, N., Harwood, I., Karpiak, J., and Kortemme, T. (2014). Quantification of the transferability of a designed protein specificity switch reveals extensive epistasis in molecular recognition. Proc Natl Acad Sci U S A *111*, 15426–15431. 10.1073/pnas.1410624111.

144. Dougherty, P.G., Wellmerling, J.H., Koley, A., Lukowski, J.K., Hummon, A.B., Cormet-Boyaka, E., and Pei, D. (2020). Cyclic Peptidyl Inhibitors against CAL/CFTR Interaction for Treatment of Cystic Fibrosis. J. Med. Chem. *63*, 15773–15784. 10.1021/acs.jmedchem.0c01528.

145. Caillet-Saguy, C., Maisonneuve, P., Delhommel, F., Terrien, E., Babault, N., Lafon, M., Cordier, F., and Wolff, N. (2015). Strategies to interfere with PDZ-mediated interactions in neurons: What we can learn from the rabies virus. Progress in Biophysics and Molecular Biology *119*, 53–59. 10.1016/j.pbiomolbio.2015.02.007.

146. Delhommel, F., Chaffotte, A., Terrien, E., Raynal, B., Buc, H., Delepierre, M., Cordier, F., and Wolff, N. (2015). Deciphering the unconventional peptide binding to the PDZ domain of MAST2. Biochem J *469*, 159–168. 10.1042/BJ20141198.

147. Valiente, M., Andrés-Pons, A., Gomar, B., Torres, J., Gil, A., Tapparel, C., Antonarakis, S.E., and Pulido, R. (2005). Binding of PTEN to Specific PDZ Domains Contributes to PTEN Protein Stability and Phosphorylation by Microtubule-associated Serine/Threonine Kinases *. Journal of Biological Chemistry *280*, 28936–28943. 10.1074/jbc.M504761200.

148. Khan, Z., Terrien, E., Delhommel, F., Lefebvre-Omar, C., Bohl, D., Vitry, S., Bernard, C., Ramirez, J., Chaffotte, A., Ricquier, K., et al. (2019). Structure-based optimization of a PDZ-binding motif within a viral peptide stimulates neurite outgrowth. J Biol Chem *294*, 13755–13768. 10.1074/jbc.RA119.008238.

149. Préhaud, C., Wolff, N., Terrien, E., Lafage, M., Mégret, F., Babault, N., Cordier, F., Tan, G.S., Maitrepierre, E., Ménager, P., et al. (2010). Attenuation of Rabies Virulence: Takeover by the Cytoplasmic Domain of Its Envelope Protein. Science Signaling *3*, ra5. 10.1126/scisignal.2000510.

150. Holt, G.T., Gorman, J., Wang, S., Lowegard, A.U., Zhang, B., Liu, T., Lin, B.C., Louder, M.K., Frenkel, M.S., McKee, K., et al. (2023). Improved HIV-1 neutralization breadth and potency of V2-apex antibodies by in silico design. Cell Reports *42*, 112711. 10.1016/j.celrep.2023.112711.

151. Wang, S. (2021). Computational Protein Design with Non-proteinogenic Amino Acids and Small Molecule Ligands, with Applications to Protein-protein Interaction Inhibitors, Anti-microbial Enzyme Inhibitors, and Antibody Design.

152. Doti, N., Mardirossian, M., Sandomenico, A., Ruvo, M., and Caporale, A. (2021). Recent Applications of Retro-Inverso Peptides. Int J Mol Sci *22*, 8677. 10.3390/ijms22168677.

153. Miles, J.J., Tan, M.P., Dolton, G., Edwards, E.S., Galloway, S.A., Laugel, B., Clement, M., Makinde, J., Ladell, K., Matthews, K.K., et al. (2018). Peptide mimic for influenza vaccination using nonnatural combinatorial chemistry. J Clin Invest *128*, 1569–1580. 10.1172/JCI91512.

154. Wang, H., Feng, Z., and Xu, B. (2017). D-amino acid-containing supramolecular nanofibers for potential cancer therapeutics. Adv Drug Deliv Rev *110–111*, 102–111. 10.1016/j.addr.2016.04.008.

155. Wei, X., Zhan, C., Shen, Q., Fu, W., Xie, C., Gao, J., Peng, C., Zheng, P., and Lu, W. (2015). A D-peptide ligand of nicotine acetylcholine receptors for brain-targeted drug delivery. Angew Chem Int Ed Engl *54*, 3023–3027. 10.1002/anie.201411226.

156. Zhou, X., Zuo, C., Li, W., Shi, W., Zhou, X., Wang, H., Chen, S., Du, J., Chen, G., Zhai, W., et al. (2020). A Novel d-Peptide Identified by Mirror-Image Phage Display Blocks TIGIT/PVR for Cancer Immunotherapy. Angew Chem Int Ed Engl *59*, 15114–15118. 10.1002/anie.202002783.

157. Miranker, A., and Karplus, M. (1991). Functionality maps of binding sites: A multiple copy simultaneous search method. Proteins: Structure, Function, and Bioinformatics *11*, 29–34. 10.1002/prot.340110104.

158. Elkin, C.D., Zuccola, H.J., Hogle, J.M., and Joseph-McCarthy, D. (2000). Computational design of D-peptide inhibitors of hepatitis delta antigen dimerization. J Comput Aided Mol Des *14*, 705–718. 10.1023/a:1008146015629.

159. Renfrew, P.D., Choi, E.J., Bonneau, R., and Kuhlman, B. (2012). Incorporation of noncanonical amino acids into Rosetta and use in computational protein-peptide interface design. PLoS One *7*, e32637. 10.1371/journal.pone.0032637.

160. Bhardwaj, G., Mulligan, V.K., Bahl, C.D., Gilmore, J.M., Harvey, P.J., Cheneval, O., Buchko, G.W., Pulavarti, S.V.S.R.K., Kaas, Q., Eletsky, A., et al. (2016). Accurate de novo design of hyperstable constrained peptides. Nature *538*, 329–335. 10.1038/nature19791.

161. Garton, M., Sayadi, M., and Kim, P.M. (2017). A computational approach for designing D-proteins with non-canonical amino acid optimised binding affinity. PLOS ONE *12*, e0187524. 10.1371/journal.pone.0187524.

162. Garton, M., Nim, S., Stone, T.A., Wang, K.E., Deber, C.M., and Kim, P.M. (2018). Method to generate highly stable D-amino acid analogs of bioactive helical peptides using a mirror image of the entire PDB. Proceedings of the National Academy of Sciences *115*, 1505–1510. 10.1073/pnas.1711837115.

163. Valiente, P.A., Wen, H., Nim, S., Lee, J., Kim, H.J., Kim, J., Perez-Riba, A., Paudel, Y.P., Hwang, I., Kim, K.-D., et al. (2021). Computational Design of Potent D-Peptide Inhibitors of SARS-CoV-2. J. Med. Chem. *64*, 14955–14967. 10.1021/acs.jmedchem.1c00655.

164. Valiente, P.A., Nim, S., Lee, J., Kim, S., and Kim, P.M. (2022). Targeting the Receptor-Binding Motif of SARS-CoV-2 with D-Peptides Mimicking the ACE2

Binding Helix: Lessons for Inhibiting Omicron and Future Variants of Concern. J Chem Inf Model *62*, 3618–3626. 10.1021/acs.jcim.2c00500.

165. Noether, E. (1983). Gesammelte Abhandlungen - Collected Papers 1st ed. (Springer Berlin, Heidelberg).

166. Wang, F., Langley, R., Gulten, G., Wang, L., and Sacchettini, J.C. (2007). Identification of a type III thioesterase reveals the function of an operon crucial for Mtb virulence. Chem Biol *14*, 543–551. 10.1016/j.chembiol.2007.04.005.

167. Williams, C.J., Headd, J.J., Moriarty, N.W., Prisant, M.G., Videau, L.L., Deis, L.N., Verma, V., Keedy, D.A., Hintze, B.J., Chen, V.B., et al. (2018). MolProbity: More and better reference data for improved all-atom structure validation. Protein Science *27*, 293–315.

168. Word, J.M., Lovell, S.C., LaBean, T.H., Taylor, H.C., Zalis, M.E., Presley, B.K., Richardson, J.S., and Richardson, D.C. (1999). Visualizing and quantifying molecular goodness-of-fit: small-probe contact dots with explicit hydrogen atoms11Edited by J. Thornton. Journal of Molecular Biology *285*, 1711–1733. 10.1006/jmbi.1998.2400.

169. Jou, J.D., Guerin, N., and Roberts, K.E. Protein Design Plugin.

170. Kugler, V., Lieb, A., Guerin, N., Donald, B.R., Stefan, E., and Kaserer, T. (2023). Disruptor: Computational identification of oncogenic mutants disrupting protein-protein and protein-DNA interactions. Commun Biol *6*, 1–6. 10.1038/s42003-023-05089-2.

171. Case, D.A., Aktulga, H.M., Belfon, K., Ben-Shalom, I.Y., and Berryman, S.R. (2022). Amber 2022.

172. Hallen, M.A., and Donald, B.R. (2017). CATS (Coordinates of Atoms by Taylor Series): protein design with backbone flexibility in all locally feasible directions. Bioinformatics *33*, i5–i12.

173. Georgiev, I., and Donald, B.R. (2009). OSPREY User Manual v1.0.

174. Terrien, E., Wolff, N., Cordier, F., Simenel, C., Bernard, A., Lafon, M., Delepierre, M., Buc, H., Prehaud, C., and Lafage, M. (2010). Solution structure of MAST2-PDZ complexed with the C-terminus of PTEN. https://doi.org/10.2210/pdb2KQF/pdb.

175. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., and Bourne, P.E. (2000). The Protein Data Bank. Nucleic Acids Research *28*, 235–242. 10.1093/nar/28.1.235.

176. Schrödinger, LLC (2015). The PyMOL Molecular Graphics System, Version 1.8.

177. Oda, Y., Saeki, K., Takahashi, Y., Maeda, T., Naitow, H., Tsukihara, T., and Fukuyama, K. (2000). Crystal structure of tobacco necrosis virus at 2.25 Å resolution11Edited by R. Huber. Journal of Molecular Biology *300*, 153–169. 10.1006/jmbi.2000.3831.

178. Skelton, N.J., Koehler, M.F.T., Zobel, K., Wong, W.L., Yeh, S., Pisabarro, M.T., Yin, J.P., Lasky, L.A., and Sidhu, S.S. (2003). Origins of PDZ Domain Ligand Specificity: STRUCTURE DETERMINATION AND MUTAGENESIS OF THE ERBIN PDZ DOMAIN *. Journal of Biological Chemistry *278*, 7645–7654. 10.1074/jbc.M209751200.

179. Nardella, C., Visconti, L., Malagrinò, F., Pagano, L., Bufano, M., Nalli, M., Coluccia, A., La Regina, G., Silvestri, R., Gianni, S., et al. (2021). Targeting PDZ domains as potential treatment for viral infections, neurodegeneration and cancer. Biol Direct *16*, 15. 10.1186/s13062-021-00303-9.

180. Tahti, E.F., Blount, J.M., Jackson, S.N., Gao, M., Gill, N.P., Smith, S.N., Pederson, N.J., Rumph, S.N., Struyvenberg, S.A., Mackley, I.G.P., et al. (2023). Additive energetic contributions of multiple peptide positions determine the relative promiscuity of viral and human sequences for PDZ domain targets. Protein Science *32*, e4611. 10.1002/pro.4611.

181. Doyle, D.A., Lee, A., Lewis, J., Kim, E., Sheng, M., and MacKinnon, R. (1996). Crystal structures of a complexed and peptide-free membrane protein-binding domain: molecular basis of peptide recognition by PDZ. Cell *85*, 1067–1076. 10.1016/s0092-8674(00)81307-0.

182. Piserchio, A., Fellows, A., Madden, D.R., and Mierke, D.F. (2012). PDZ Domain of CAL (Cystic Fibrosis Transmembrane Regulator-Associated Ligand). https://doi.org/10.2210/pdb2LOB/pdb.

183. Amacher, J.F., Cushing, P.R., Bahl, C.D., Beck, T., and Madden, D.R. (2013). Stereochemical Determinants of C-terminal Specificity in PDZ Peptide-binding Domains. J Biol Chem *288*, 5114–5126. 10.1074/jbc.M112.401588.

184. Terrien, E., Chaffotte, A., Lafage, M., Khan, Z., Préhaud, C., Cordier, F., Simenel, C., Delepierre, M., Buc, H., Lafon, M., et al. (2012). Interference with the PTEN-MAST2 Interaction by a Viral Protein Leads to Cellular Relocalization of PTEN. Science Signaling *5*, ra58. 10.1126/scisignal.2002941.

185. Lyamichev, V.I., Goodrich, L.E., Sullivan, E.H., Bannen, R.M., Benz, J., Albert, T.J., and Patel, J.J. (2017). Stepwise Evolution Improves Identification of Diverse Peptides Binding to a Protein Target. Sci Rep *7*, 12116. 10.1038/s41598-017-12440-1.

186. Freitag, S., Le Trong, I., Klumb, L., Stayton, P.S., and Stenkamp, R.E. (1997). Structural studies of the streptavidin binding loop. Protein Sci *6*, 1157–1166.

187. Brower, M.S., Brakel, C.L., and Garry, K. (1985). Immunodetection with streptavidin-acid phosphatase complex on Western blots. Analytical Biochemistry *147*, 382–386. 10.1016/0003-2697(85)90286-6.

188. Freitag, S., Le Trong, I., Klumb, L.A., Chu, V., Chilkoti, A., Stayton, P.S., and Stenkamp, R.E. (1999). X-ray crystallographic studies of streptavidin mutants binding to biotin. Biomolecular Engineering *16*, 13–19. 10.1016/S1050-3862(99)00048-0.

189. Hyre, D.E., Le Trong, I., Merritt, E.A., Eccleston, J.F., Green, N.M., Stenkamp, R.E., and Stayton, P.S. (2006). Cooperative hydrogen bond interactions in the streptavidin–biotin system. Protein Sci *15*, 459–467. 10.1110/ps.051970306.

190. Cielo, C.B.C., Okazaki, S., Suzuki, A., Mizushima, T., Masui, R., Kuramitsu, S., and Yamane, T. (2010). Structure of ST0929, a putative glycosyl transferase from Sulfolobus tokodaii. Acta Crystallogr Sect F Struct Biol Cryst Commun *66*, 397–400. 10.1107/S1744309110006354.

191. Zhou, J., Panaitiu, A.E., and Grigoryan, G. (2020). A general-purpose protein design framework based on mining sequence–structure relationships in known protein structures. Proc Natl Acad Sci U S A *117*, 1059–1068. 10.1073/pnas.1908723117.

192. Du, H., Bing, J., Hu, T., and others (2020). Candida auris: Epidemiology, biology, antifungal resistance, and virulence. PLoS pathogens *16*, e1008921. 10.1371/journal.ppat.1008921.

193. Pillay, D., and Zambon, M. (1998). Antiviral drug resistance. BMJ *317*, 660–662.

194. Lampejo, T. (2020). Influenza and antiviral resistance: an overview. European Journal of Clinical Microbiology & Infectious Diseases *39*, 1201–1208. 10.1007/s10096-020-03840-9.

195. Maier, J.A., Martinez, C., Kasavajhala, K., Wickstrom, L., Hauser, K.E., and Simmerling, C. (2015). ff14SB: Improving the Accuracy of Protein Side Chain and Backbone Parameters from ff99SB. J Chem Theory Comput *11*, 3696–3713. 10.1021/acs.jctc.5b00255.

196. Huynh, K. (2023). Discovery and Characterization of Novel Thanatin Orthologs Against Escherichia coli LptA and Pseudomonas aeruginosa LptH.

197. Bouvette, J., Huang, Q., Riccio, A.A., Copeland, W.C., Bartesaghi, A., and Borgnia, M.J. (2022). Automated systematic evaluation of cryo-EM specimens with SmartScope. eLife *11*, e80047. 10.7554/eLife.80047.

198. Kryshtafovych, A., Moult, J., Albrecht, R., Chang, G.A., Chao, K., Fraser, A., Greenfield, J., Hartmann, M.D., Herzberg, O., Josts, I., et al. (2021). Computational models in the service of X-ray and cryo-electron microscopy structure determination. Proteins: Structure, Function, and Bioinformatics *89*, 1633–1646. 10.1002/prot.26223.

199. Laine, E., Eismann, S., Elofsson, A., and Grudinin, S. (2021). Protein sequence-to-structure learning: Is this the end(-to-end revolution)? Proteins *89*, 1770–1786. 10.1002/prot.26235.

200. Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., et al. (2023). Evolutionary-scale prediction of atomic-level protein structure with a language model. Science *379*, 1123–1130. 10.1126/science.ade2574.

201. Madani, A., Krause, B., Greene, E.R., Subramanian, S., Mohr, B.P., Holton, J.M., Olmos, J.L., Xiong, C., Sun, Z.Z., Socher, R., et al. (2023). Large language models generate functional protein sequences across diverse families. Nat Biotechnol *41*, 1099–1106. 10.1038/s41587-022-01618-2.

202. Gainza, P., Wehrle, S., Van Hall-Beauvais, A., Marchand, A., Scheck, A., Harteveld, Z., Buckley, S., Ni, D., Tan, S., Sverrisson, F., et al. (2023). De novo design of protein interactions with learned surface fingerprints. Nature *617*, 176–184. 10.1038/s41586-023-05993-x.

203. Shin, J.-E., Riesselman, A.J., Kollasch, A.W., McMahon, C., Simon, E., Sander, C., Manglik, A., Kruse, A.C., and Marks, D.S. (2021). Protein design and variant prediction using autoregressive generative models. Nat Commun *12*, 2403. 10.1038/s41467-021-22732-w.

204. Bryant, D.H., Bashir, A., Sinai, S., Jain, N.K., Ogden, P.J., Riley, P.F., Church, G.M., Colwell, L.J., and Kelsic, E.D. (2021). Deep diversification of an AAV capsid protein by machine learning. Nat Biotechnol *39*, 691–696. 10.1038/s41587-020-00793-4.

205. Das, P., Sercu, T., Wadhawan, K., Padhi, I., Gehrmann, S., Cipcigan, F., Chenthamarakshan, V., Strobelt, H., Dos Santos, C., Chen, P.-Y., et al. (2021). Accelerated antimicrobial discovery via deep generative models and molecular dynamics simulations. Nat Biomed Eng *5*, 613–623. 10.1038/s41551-021-00689-x.

206. Anishchenko, I., Pellock, S.J., Chidyausiku, T.M., Ramelot, T.A., Ovchinnikov, S., Hao, J., Bafna, K., Norn, C., Kang, A., Bera, A.K., et al. (2021). De novo protein design by deep network hallucination. Nature *600*, 547–552. 10.1038/s41586-021-04184-w.

207. Mason, D.M., Friedensohn, S., Weber, C.R., Jordi, C., Wagner, B., Meng, S.M., Ehling, R.A., Bonati, L., Dahinden, J., Gainza, P., et al. (2021). Optimization of therapeutic antibodies by predicting antigen specificity from antibody sequence via deep learning. Nat Biomed Eng *5*, 600–612. 10.1038/s41551-021-00699-9.

208. Gainza, P., Sverrisson, F., Monti, F., Rodolà, E., Boscaini, D., Bronstein, M.M., and Correia, B.E. (2020). Deciphering interaction fingerprints from protein molecular

surfaces using geometric deep learning. Nat Methods *17*, 184–192. 10.1038/s41592-019-0666-6.

209. Hallen, M.A., Jou, J.D., and Donald, B.R. (2017). LUTE (Local unpruned tuple expansion): Accurate continuously flexible protein design with general energy functions and rigid Rotamer-Like efficiency. Journal of Computational Biology *24*, 536–546.

210. AlQuraishi, M. (2019). AlphaFold at CASP13. Bioinformatics *35*, 4862–4865. 10.1093/bioinformatics/btz422.

211. Kryshtafovych, A., Schwede, T., Topf, M., Fidelis, K., and Moult, J. (2021). Critical assessment of methods of protein structure prediction (CASP)-Round XIV. Proteins *89*, 1607–1617. 10.1002/prot.26237.

212. Heo, L., Janson, G., and Feig, M. (2021). Physics-Based Protein Structure Refinement in the Era of Artificial Intelligence. Proteins *89*, 1870–1887. 10.1002/prot.26161.

213. Evans, R., O'Neill, M., Pritzel, A., Antropova, N., Senior, A., Green, T., Žídek, A., Bates, R., Blackwell, S., Yim, J., et al. (2022). Protein complex prediction with AlphaFold-Multimer. 2021.10.04.463034. 10.1101/2021.10.04.463034.

214. Lowegard, A. (2019). Novel Algorithms and Tools for Computational Protein Design with Applications to Drug Resistance Prediction, Antibody Design, Peptide Inhibitor Design, and Protein Stability Prediction.

215. Pak, M.A., Markhieva, K.A., Novikova, M.S., Petrov, D.S., Vorobyev, I.S., Maksimova, E.S., Kondrashov, F.A., and Ivankov, D.N. (2023). Using AlphaFold to predict the impact of single mutations on protein stability and function. PLoS One *18*, e0282689. 10.1371/journal.pone.0282689.

216. DeepMind and EMBL-EBI (2022). AlphaFold Protein Structure Database. Frequently asked questions. https://alphafold.ebi.ac.uk/faq.

217. Pak, M.A., and Ivankov, D.N. (2022). Best templates outperform homology models in predicting the impact of mutations on protein stability. Bioinformatics *38*, 4312–4320. 10.1093/bioinformatics/btac515.

218. Heroux, M.A., Bernholdt, D.E., McInnes, L.C., Cary, J.R., Katz, D.S., Raybourn, E.M., and Rouson, D. (2023). Basic Research Needs in The Science of Scientific Software Development and Use: Investment in Software is Investment in Science 10.2172/1846009.

219. Martin, R.C. (2008). Clean Code: A Handbook of Agile Software Craftsmanship 1st edition. (Pearson).

220. Sastry, G.M., Adzhigirey, M., Day, T., Annabhimoju, R., and Sherman, W. (2013). Protein and ligand preparation: parameters, protocols, and influence on virtual screening enrichments. Journal of computer-aided molecular design *27*, 221–234.

221. Schrödinger, LLC, New York, NY (2022). Schrödinger Release 2022-4: Maestro.

222. Farid, R., Day, T., Friesner, R.A., and Pearlstein, R.A. (2006). New insights about HERG blockade obtained from protein modeling, potential energy mapping, and docking studies. Bioorganic & medicinal chemistry *14*, 3160–3173.

223. Sherman, W., Beard, H.S., and Farid, R. (2006). Use of an induced fit receptor structure in virtual screening. Chemical biology & drug design *67*, 83–84.

224. Sherman, W., Day, T., Jacobson, M.P., Friesner, R.A., and Farid, R. (2006). Novel procedure for modeling ligand/receptor induced fit effects. Journal of medicinal chemistry *49*, 534–553.

225. Jones, G., Willett, P., Glen, R.C., Leach, A.R., and Taylor, R. (1997). Development and validation of a genetic algorithm for flexible docking. Journal of molecular biology *267*, 727–748.

226. Haling, J.R., Sudhamsu, J., Yen, I., Sideris, S., Sandoval, W., Phung, W., Bravo, B.J., Giannetti, A.M., Peck, A., Masselot, A., et al. (2014). Structure of the BRAF-MEK complex reveals a kinase activity independent role for BRAF in MAPK signaling. Cancer cell *26*, 402–413.

227. Xu, W., Doshi, A., Lei, M., Eck, M.J., and Harrison, S.C. (1999). Crystal structures of c-Src reveal features of its autoinhibitory mechanism. Molecular cell *3*, 629–638.

228. Chemical Computing Group ULC (2019). Molecular Operating Environment (MOE).

229. Case, D.A., Belfon, K., Ben-Shalom, I., Brozell, S.R., Cerutti, D., Cheatham, T., Cruzeiro, V.W.D., Darden, T., Duke, R.E., Giambasu, G., et al. (2021). Amber 2021.

230. Lazaridis, T., and Karplus, M. (1999). Effective energy function for proteins in solution. Proteins: Structure, Function, and Bioinformatics *35*, 133–152.

231. Berman, H., Henrick, K., and Nakamura, H. (2003). Announcing the worldwide protein data bank. Nature Structural & Molecular Biology *10*, 980–980. 10.1038/nsb1203-980.

232. Chaikuad, A., Tacconi, E., Zimmer, J., Liang, Y., Gray, N.S., Tarsounas, M., and Knapp, S. (2014). A unique inhibitor binding site in ERK1/2 is associated with slow binding kinetics. Nat Chem Biol *10*, 853–860. 10.1038/nchembio.1629.

233. Garai, Á., Zeke, A., Gógl, G., Törö, I., Ferenc, F., Blankenburg, H., Bárkai, T., Varga, J., Alexa, A., Emig, D., et al. (2012). Specificity of linear motifs that bind to a common mitogen-activated protein kinase docking groove. Sci Signal *5*, ra74. 10.1126/scisignal.2003004.

234. Webb, B., and Sali, A. (2016). Comparative protein structure modeling using MODELLER. Current Protocols in Protein Science *86*, 2.9.1-2.9.37. 10.1002/cpps.20.

235. Shelley, J.C., Cholleti, A., Frye, L.L., Greenwood, J.R., Timlin, M.R., and Uchimaya, M. (2007). Epik: a software program for pKa prediction and protonation state generation for drug-like molecules. J Comput Aided Mol Des *21*, 681–691. 10.1007/s10822-007-9133-z.

236. Vergé, A. (2023). yamllint - A linter for YAML files.

237. Brenan, L., Andreev, A., Cohen, O., Pantel, S., Kamburov, A., Cacchiarelli, D., Persky, N.S., Zhu, C., Bagul, M., Goetz, E.M., et al. (2016). Phenotypic characterization of a comprehensive set of MAPK1/ERK2 missense mutants. Cell Reports *17*, 1171–1183. 10.1016/j.celrep.2016.09.061.

238. YAML Language Development Team YAML Ain’t Markup Language (YAML^{TM}) revision 1.2.2. https://yaml.org/spec/1.2.2/.

Biography

Nathan Guerin took a rather unconventional path to computer science and computational protein design. He was admitted on a musical performance scholarship to the University of North Carolina at Chapel Hill in 2006, where he studied marimba with Lynn Glassock and jazz percussion with Thomas Taylor. A bad shoulder injury in his sophomore year put his future in musical performance in doubt and he switched majors to study Chinese. After graduation in 2010, like many humanities majors, Nathan could not find work in his area of study so took a job as a telephone support agent in a healthcare software company. He soon realized that the interesting work was in the software development department, and it was there that Nathan started studying computer science and software engineering. Over the subsequent 8 years, Nathan worked as a software engineer in a variety of industries, including in fast-growing European startups and cutting-edge biotech companies. During this period, he also earned a master's degree in software engineering from the Harvard Extension School under the guidance of James Frankel, where he focused on systems engineering and bioinformatics.

Nathan matriculated into the Duke Computer Science Department in 2018 and soon after joined Bruce Donald's lab, where he specializes in the development of algorithms and application of computational protocols for protein design. Under Donald's tutelage, Nathan has developed novel methods for predicting drug resistance and for generating *de novo* D-peptide inhibitors. A licensed pilot, Nathan enjoys flying in his free time when weather permits.