JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 00, Number 00, 2024

Pp. 000–000
DOI: 10.1089/cmb.2024.0669

Protocol for Designing *De Novo* Noncanonical Peptide Binders in OSPREY

HENRY CHILDS,[1] NATHAN GUERIN,[2] PEI ZHOU,[3] and BRUCE R. DONALD[1–4]

ABSTRACT

D-peptides, the mirror image of canonical L-peptides, offer numerous biological advantages that make them effective therapeutics. This article details how to use DexDesign, the newest OSPREY-based algorithm, for designing these D-peptides *de novo*. OSPREY physics-based models precisely mimic energy-equivariant reflection operations, enabling the generation of D-peptide scaffolds from L-peptide templates. Due to the scarcity of D-peptide:L-protein structural data, DexDesign calls a geometric hashing algorithm, Method of Accelerated Search for Tertiary Ensemble Representatives, as a subroutine to produce a synthetic structural dataset. DexDesign enables mixed-chirality designs with a new user interface and also reduces the conformation and sequence search space using three new design techniques: Minimum Flexible Set, Inverse Alanine Scanning, and K*-based Mutational Scanning.

Keywords: protocol, protein:ligand binding, *de novo* peptide design, D-peptides, OSPREY.

1. INTRODUCTION

The 20 proteinogenic L-amino acids (termed *canonical*) in biological systems serve as the foundation for all proteins. However, exploration of noncanonical amino acids may yield benefits not available via standard ribosomal pathways. Specifically, D-peptides, the mirror image of L-peptides, have been shown to exhibit decreased protease recognition (Di, 2014), low immunogenicity (Benkirane et al., 1993), favorable bioavailability (Craik et al., 2013), and high binding affinity (Angelini et al., 2012). Furthermore, modeling and designing noncanonical amino acids has become increasingly important for the design of high-affinity binders (Chen et al., 2009; Holt et al., 2023; Stevens et al., 2006; Wang, 2021; Wang et al., 2022). However, the number of previous algorithms for designing noncanonical peptides is sparse (Donald, 2011; Elkin et al., 2000; Garton et al., 2017; Renfrew et al., 2012). For these reasons, a new algorithm and protocol were needed to make designs with noncanonical amino acids as facile as the OSPREY 3.0 pipeline (Hallen et al., 2018) for design

[1]Department of Chemistry, Duke University, Durham, North Carolina, USA.
[2]Department of Computer Science, Duke University, Durham, North Carolina, USA.
[3]Department of Biochemistry, Duke University School of Medicine, Durham, North Carolina, USA.
[4]Department of Mathematics, Duke University, Durham, North Carolina, USA.
An early version of this article was published as part of the 2024 Annual International Conference on Research in Computational Molecular Biology (RECOMB).

using the standard proteinogenic L-amino acids. Therefore, we have developed a new algorithm, DexDesign (Guerin et al., 2024), for design of mixed-chirality complexes in our open-source protein redesign software suite, OSPREY.

2. PREVIOUS WORK

A detailed literature review comparing DexDesign to previous methods is provided in the Background section (p. 2) of Guerin et al. (2024). In brief, Elkin et al. used multiple copy simultaneous search (Miranker and Karplus, 1991) to predict D-peptide inhibitors of hepatitis delta antigen dimerization (Elkin et al., 2000). Phillip Kim's group developed a computational method for designing D-peptide binders by inverting the PDB (Garton et al., 2017). Additionally, recent Rosetta versions include noncanonical design capabilities (Bhardwaj et al., 2016; Renfrew et al., 2012). The number of available computational tools for noncanonical design is notably sparse. Furthermore, while previous techniques have noncanonical design functionality, these lack geometric substructure search (Zhou and Grigoryan, 2015), continuous flexibility (Gainza et al., 2012), multistate design using partition functions over molecular ensembles (Georgiev et al., 2008; Hallen et al., 2018), and the provable guarantees on accuracy and computational complexity (Hallen et al., 2018; Hallen and Donald, 2019) compared to the DexDesign framework. For further details, refer to the Discussion section (p. 11) of Guerin et al. (2024).

3. METHODS

DexDesign uses energy-equivariant reflection operations, a protein substructure search [Method of Accelerated Search for Tertiary Ensemble Representatives (MASTER) (Zhou and Grigoryan, 2015)], the K* algorithm (Georgiev et al., 2008; Hallen et al., 2018), and novel design techniques [Minimum Flexible Set (MFS), Inverse Alanine Scanning (IAS), K*-based Mutational Scanning (K*MS)] to design noncanonical peptides for biological targets.

Reflection is an energy-equivariant geometric transformation, meaning that this operation corresponds to symmetry in the energy field of the protein structure (Noether, 1983). DexDesign mimics the physics of this operation precisely: the energy function will calculate the same energy for an L-protein and its mirror image (the corresponding D-protein) (Guerin et al., 2024). Furthermore, MASTER (Zhou and Grigoryan, 2015) is a geometric search algorithm that rapidly searches over a protein database to return substructures with structural similarities to our protein structure, called the query. This search is guaranteed to return protein substructures below a user-specified RMSD given an optimal superposition of the motif onto database structures. MASTER is used to return L-substructures with high geometric similarity to a D-peptide. Finally, the algorithms that design for affinity by predicting and maximizing binding affinity over sequence and conformation space for both canonical and noncanonical structures are applied (Donald, 2011; Frey et al., 2010; Georgiev et al., 2008; Hallen et al., 2018; Hallen and Donald, 2016; Holt et al., 2023; Jou et al., 2020; Jou et al., 2016; Lilien et al., 2005; Ojewole et al., 2018; Reeve et al., 2015).

The K* algorithm (Georgiev et al., 2008; Hallen et al., 2018) computes a provably good ε-approximation to the binding affinity constant, K_a. K* computes partition functions over molecular ensembles of a continuously flexing backbone (Hallen et al., 2013; Hallen and Donald, 2017) and sidechains (Gainza et al., 2012) while translating and rotating the ligand. This algorithm first calculates the Boltzmann-weighted partition function (q) for the protein (P), ligand (L), and protein:ligand complex (PL) up to the user-specified accuracy (ε). This approximation is q^*, where $q^* \geq (q)(1-\varepsilon)$. For the given state $X \in \{P, L, PL\}$ and an arbitrary sequence (s), the partition function is defined in Eq. (1) for each conformation c in the conformation space Q:

$$q_x(s) = \sum_{c \in Q_x(s)} \exp\left(-\frac{E(c)}{RT}\right). \tag{1}$$

R and T are the ideal gas constant and temperature, respectively. $E(c)$ is the energy of a single conformation, c. The K* score is determined by the ratio of bound and unbound states:

$$K^*(s) = \frac{q_{PL}(s)}{q_P(s)q_L(s)}. \quad (2)$$

A higher K* score predicts tighter binding affinity and is therefore desirable to maximize. This is a challenging objective, as a search over the sequence space for a given peptide scales exponentially in the number of mutable residues. Novel design techniques are implemented to reduce the computational cost of multistate design in both the sequence and conformation space. These techniques are the MFS, IAS, and K*MS (Guerin et al., 2024). MFS updates MASTER-returned substructures to new chemical environments, IAS returns point mutations that are predicted to improve binding affinity (increase the K* score), giving optimistic flexibility, and K*MS returns multiple simultaneous mutations that are predicted to improve binding affinity.

4. HOW TO USE DEXDESIGN TO DESIGN *DE NOVO* NONCANONICAL PEPTIDES

In this section, we describe how to prepare and run OSPREY to design *de novo* D-peptides for L-targets using the DexDesign algorithm. Video tutorials are available as references. All example commands are written for a Linux distribution.

1. Substructure search with MASTER
   1.1. *Compiling MASTER*. Download the source code for MASTER (Zhou and Grigoryan, 2015) from the Grigoryan Lab website (grigoryanlab.org/master). This website includes an INSTALL file for compiling on Linux or MacOS alongside video tutorials.
   1.2. *Creating a database*. A database of PDB files must be created to perform a search using MASTER. We recommend using the PDB Advanced Search Interface to filter out low-resolution crystal structures, DNA, RNA, and small molecules. We obtained 119,160 protein crystal structures (median resolution = 1.9 Å) using this method, but structural data can be curated for diverse design objectives. Create a PDS (Protein Data Structure) file with this structural data using MASTER's *createPDS* executable. For a single file (ex. structure.pdb, replace *database* with the directory location of the structural data):

```
createPDS \
  --type target \
  --pdb database/structure.pdb \
  --pds database/structure.pds
```

   Write a for loop to create PDS files for all PDB files in the *database*. Also create a lookup db.txt file for these structures with the directory location. For example, db.txt may read:

```
database/structure1.pds
database/structure2.pds
...
```

   1.3. *Preparing the query*. Obtain a high-resolution structural model of an L-peptide in complex with an L-protein. Simply delete the target protein coordinates from the PDB file to isolate the L-peptide. Reflect to a D-peptide using OSPREY and save the resulting PDB file:

```
osprey3 invert L-peptide.pdb > D-peptide.pdb
```

   With these requisites, compile the D-peptide structural file:

```
createPDS \
  --type query \
  --pdb D-peptide.pdb \
  --pds D-peptide.pds
```

This *query* will be aligned to *targets* in our database for a substructure search.

1.4. *Searching with MASTER*. Perform a MASTER search using the D-peptide query:

```
master \
  --query D-peptide.pds \
  --targetList db.txt \
  --rmsdCut 0.5 \
  --outType full \
  --matchOut fullbb.txt \
  --structOut fullbb
```

The *targetList* points to the location of the database lookup file, while *rmsdCut* specifies the upper bound on alignment error (0.5 Å provided as an example). The remaining flags are for MASTER output directory location and type (full complex or substructure only).

1.5. *Generating a mixed-chirality complex*. After obtaining the scaffold PDB files (saved in ./fullbb in the example command of Section 1.4), review the match alignment error data (fullbb.txt). Select a substructure for redesign and reflect the L-peptide into D-space, again using OSPREY (see Section 1.3). With your method of choice, align the D-peptide to the endogenous L-peptide from the original L-peptide:L-protein structure. This will place the D-peptide into the L-protein binding pocket. Delete the endogenous L-peptide atoms from the aligned PDB file, producing a D-peptide in complex with an L-protein.

2. Optimizing binding affinity with DexDesign

2.1. *Starting OSPREY*. After installing Java Development Kit Version 19, start the OSPREY user interface to access the DexDesign algorithm setup:

```
osprey3 setup-design
```

See Supplementary Video S1 for a demonstration of this setup process on Ubuntu 20.04.6 LTS.

2.2. *Preparing the PDB and OMOL files*. Scaffolds returned by MASTER need to be screened for accurate atomic labeling, chirality assignment, and protonation. We now offer these functions directly in the OSPREY interface. See Supplementary Video S2 for a demonstration of this preparation process.

2.2.1. *Import PDB*. In the top left of the interface, select *File > Import PDB*. Import your complex.

2.2.2. *Prepare PDB*. In the new menu, select *Prepare*. Review the options for *Filter*, *Chirality*, *Duplicated Atoms*, *Missing Atoms*, *Bonds*, and *Protonation* to ensure your PDB file is correctly prepared. Be sure to check the box labeling the peptide as D-space. We recommend updating any missing atoms, reviewing bonds, and re-protonating any structure using OSPREY before continuing.

2.2.3. *Save PDB and OMOL*. In the same menu, select *File > Save OMOL* and *File > Export PDB*. The OMOL (OSPREY Molecule) filetype will save the prepared file in a format amenable to further design specifications. Save the prepared PDB file for use in Section 2.5. Close this menu by selecting *File > Close*.

2.3. *Preparing sequence and conformation spaces*. With an OMOL file ready, we may assign amino acid libraries, mutations, and flexibility to our design. This section serves solely as a reference for later design techniques; skip to Section 2.5 for details on how to apply these procedures.

2.3.1. *Create a conformation space*. In the interface, select *File > New Conformation Space* and select your OMOL file.

2.3.2. *Assign conformation libraries*. DexDesign implements a general, in contrast to application-specific, approach to protein design. Users may now upload custom modeling templates and flexibility, allowing conformation space specification for any noncanonical amino acids [e.g., sulfated tyrosine (Holt et al., 2023)]. Assign a library by selecting *Edit > Conformation Libraries > Chain # > Add*. For this design, we will assign the *Amino Acids* library to our L-target chain and the *D-Amino Acids* library to our D-peptide chain.

2.3.3. *Select mutations*. To assign mutants to the sequence search space, select *Edit* > *Mutations* > *Chain #* > *Add* > *Protein*. Select a residue and assign mutants via the *Mutations* tab.

2.3.4. *Select flexibility*. To set a residue as flexible, select *Edit* > *Flexibility* > *Chain #*. Under *Flexible Positions*, select *Add* > *Protein* and select a residue. Note that flexibility for mutants is selected under *Mutable Positions*: select a mutable residue, then click *Edit* and select desired continuous rotamers.

2.3.5. *Select molecular motion*. To enable ligand translation and rotation, select *Add* under *Molecule Motions* in the *Flexibility Editor*. Ensure *Translation & Rotation* is selected.

2.3.6. *Save conformation space files*. The conformation space must be saved for the protein, ligand, and complex. After selecting desired libraries, mutations, flexibility, and motions, save the complex by selecting *File* > *Save Conformation Space*. Save the ligand and protein by selecting *File* > *Split Conformation Space* > *Chain #* > *Save*.

2.3.7. *Compile conformation spaces*. Compile the conformation space for the protein, ligand, and complex. Select *File* > *Open Conformation Space* > *file.confspace*. Then, click *Compile Conformation Space* > *Compile* > *Save*. Repeat this process twice (replace *file.confspace* with the .confspace files for the protein, ligand, and complex). Output files will have filetype .ccsx (compiled conformation space).

2.3.8. *Run K**. A K* score can only be computed using .ccsx filetypes. Pass the compiled conformation spaces to OSPREY to predict the binding affinity of the ligand to the protein. Set the directory for ensemble outputs with *–ensemble-dir* and the ε value with *-e*. A higher K* score correlates to better predicted binding affinity.

```
osprey3 kstar \
   --complex-confspace complex.ccsx \
   --target-confspace target.ccsx \
   --design-confspace design.ccsx \
   --ensemble-dir ensembles \
   -e 0.3
```

2.4. *Backbone flexibility*. Algorithms for incorporating backbone flexibility similar to Hallen and Donald (2017) and Hallen et al. (2013) can be used with DexDesign, but many users will prefer trying a more lightweight protocol first. We recommend using additional backbone sampling and remodeling, which has been shown to increase native sequence recovery and predicted binding affinity (Guerin et al., 2024). DexDesign incorporates a constrained molecular dynamics simulation, SANDER (Case et al., 2023), for backbone and sidechain atomic movements. This is available in the *Prepare* > *Minimization* menu of the OSPREY interface. Alternatively, multiple MASTER-returned backbones can be used as a template for redesign. See Section 4.2.1 of Guerin et al. (2024) for more information.

2.5. *Running MFS*. In order to produce a complex amenable to K* maximization, visualize and resolve the steric clashes between the D-ligand and L-target residues using the prepared PDB file from Section 2.2.3. See Supplementary Video S3 for a demonstration of the Minimum Flexible Set.

2.5.1. *Locate steric clashes*. Obtain and install the Donald lab Protein Design Plugin (Jou et al., 2023), which includes a streamlined implementation of ProbeDots (Word et al., 1999) for steric clash assessment in PyMOL (Schrödinger, 2015). Set peptide and target residues in the OSPREY interface participating in steric clashes as flexible (Section 2.3.4). These are labeled in PyMOL as the group bad_overlap.

2.5.2. *Run MFS*. To produce a flexible D-peptide, compile the conformation spaces and run the K* algorithm (Hallen et al., 2018; Lilien et al., 2005), passing the compiled conformation spaces from Section 2.5.1 to OSPREY (see Section 2.3.8).

2.5.3. *Evaluate*. The design technique of MFS is complete when ensemble outputs (located as PDB files in the directory *ensembles*; see example command in Section 2.3.8) from OSPREY contain no steric overlaps between the D-peptide and L-target. If the MFS has an intractable runtime, this indicates an infeasible scaffold. A different MASTER-returned substructure should be selected for design. This ensemble structure will be used as input to IAS.

2.6. *Running IAS*. With a competent starting structure produced using the Minimum Flexible Set, we will locate point mutations that will increase the binding affinity of the noncanonical peptide for the target. Let us first assess for large improvements in predicted binding affinity given reduced geometric constraints. See Supplementary Video S4 for a demonstration of IAS.

2.6.1. *Design with optimistic geometry*. For our D-peptide ligand to have optimal sidechain flexibility and ligand translation/rotation, we can mutate all residues, modulo one, to alanine. To do this, mutate all residues (except the N-term) to alanine (Section 2.3.3). Assign mutations to the N-terminus for all 20 D-amino acids. Set target residues that are ≤4 Å from the ligand N-terminus as flexible (Section 2.3.4).

2.6.2. *Run and repeat.* Compile and run the K* algorithm (Section 2.3.8). Repeat this process, mutating all residues to alanine except for the next residue in the peptide chain. Mutate this residue to all 20 D-amino acids, setting target residues ≤4 Å from this residue as flexible. Repeat this procedure for all residues in the D-peptide.

2.6.3. *Find point mutations*. Sequences with the highest K* score (best-predicted binding affinity) should be noted for a later combinatorial search (Section 2.6.4). Record both favorable mutations and their corresponding flexible residues for each point mutation.

2.6.4. *Reduce the conformation space*. The K* algorithm returns a customizable number of lowest-energy conformations (default: 10) as an ensemble. Import the input structure (Section 2.6.1) and output ensemble (Section 2.6.2) into molecular visualization software. Inspect the sidechain dihedrals of the input and output structures. Remove flexible target residues from the combinatorial search that do not change rotamer conformation relative to the input structure. This results in a complexity speedup for later sidechain flexibility searches.

2.6.5. *Combinatorial search*. Run the K* algorithm on the reduced search space of mutable and flexible residues. Review and note the sequence with the best-predicted binding, which will be used as input to minimization.

2.7. *Minimization*. As noted in Section 2.4, DexDesign includes SANDER for constrained molecular dynamics calculations. Performing minimization on a post-IAS ensemble commonly produces lower-energy structures. See Supplementary Video S5 for a tutorial on running minimization via the OSPREY interface.

2.7.1. *Minimize the structure*. Find and save the IAS sequence with the best-predicted binding (Section 2.6.5). Import the PDB and select *Prepare > Minimize > Chain #*. Customize the number of steps using the slider and click *Minimize Selected Molecules*. Export the minimized PDB file by selecting *File > Export PDB*.

2.7.2. *Assess energy*. To assess the change in atomic positions, evaluate the minimized PDB file in molecular visualization software. Also run the K* algorithm, setting all clashing residues as flexible. Ensure ligand Translation & Rotation is enabled, as molecular geometries from minimization may alter the ligand position in the binding pocket. This ensemble output is used as input to K*MS.

2.8. *Running K*MS*. We will now identify favorable simultaneous mutants given peptide and target geometric constraints. See Supplementary Video S6 for a demonstration of K*MS.

2.8.1. *Scan for mutants*. Obtain the minimized ensemble output (Section 2.7.2). Mutate the N-term residue to all 20 D-amino acids, setting all target residues ≤4 Å from the ligand residue as flexible. Run the K* algorithm and obtain the K* scores for this residue.

2.8.2. *Mutate all residues*. Find the mutant that maximizes the K* score for the N-terminus and obtain the ensemble output for this sequence. Perform the scan procedure (Section 2.8.1) for the next residue in the peptide chain using the N-term mutant sequence as input. Repeat this process for all residues in the peptide, using the optimal mutant for the previous residue as input for each scan.

2.9. *Interpreting results*. The K* algorithm has demonstrated a high correlation [Spearman's $\rho = 0.81$ (Lowegard, 2019)] between computational and experimental measurements of affinity. The results of DexDesign can be used to rank candidate sequence for *in vitro*/*in vivo* validation.

5. COMPUTATIONAL COMPLEXITY

DexDesign includes several design techniques that compute a K* score from thermodynamic ensembles, which is computationally expensive. Previous sublinear K* maximization algorithms include BBK* (Ojewole et al., 2018) with MARK* (Jou et al., 2020), but these fail to adequately reduce the number of partition function calls for D-peptide redesigns (DPRs) enumerating a large search space. To reduce the number of K* computations by pruning prohibitively large sequence spaces, we implemented and analyzed the following techniques.

DPR scaffold generation: Starting with a given L-peptide:L-target complex, the DexDesign algorithm outputs a D-peptide query (Q) for input to the MASTER search algorithm. While the MASTER algorithm is worst-case exponential in the number of disjoint query segments (Zhou and Grigoryan, 2015), in DexDesign Q is a single segment, and therefore the time required to calculate the optimal rotation and translation matrix with the Kabsch algorithm (Kabsch, 1978, 1976) between Q and each contiguous, equally sized segment in a MASTER database with s residues is $O(s)$. MASTER can compute over a million Kabsch superimpositions per second, leading to empirical runtimes on the order of seconds. MASTER returns the u best results in order of backbone RMSD, so DexDesign generates u DPR scaffolds, with each scaffold containing a D-peptide P and a protein target T.

Minimum Flexible Set: In contrast to the upper bound time complexities given for other redesign methods in DexDesign, MFS provides a lower bound on the size of the conformation space that must be searched or pruned by the downstream K* design. The MFS lower bound predicts the running time of the K* designs and can be used to eliminate infeasible designs. In this sense, MFS serves a role similar to TESS in BWM* as an efficiently computable metric of problem complexity that predicts designability [see Jou et al. (2016) for TESS proof]. Assume P has n residues and T has r residues. By computing the distances between the atoms in P and T, the MFS can be computed in $O(nr)$ time. P is often much smaller than T and, in such cases, it is more efficient to compute a bounding ball around the peptide inflated by 4 Å in $O(n)$ time and clip T's atoms to those that lie within the ball in $O(r)$ time. Let d be the number of T's residues in the ball, then the MFS can be computed in $O(r + nd)$ time. The MFS is comprised of c clashing peptide residues. We prune DPR scaffolds where the ratio of c to n exceeds 3/4, reducing the number of DPR scaffolds u for which we compute IAS and K*-MS. Furthermore, c contributes to the lower bound on the size of the conformation space required to be searched or pruned by the K*-based Mutational Scans. Let r be the minimum number of rotamers for a clashing residue, then the size of the conformation space input to the K* search is $O(r^c)$. This is important because in practice the time required to compute an ε-accurate partition function for a protein sequence is dependent on the size of the conformation space (Jou et al., 2020; Nisonoff, 2015; Ojewole et al., 2018), so MFS aids in pruning DPR scaffolds for which partition functions would be challenging to compute. Empirically, c averages 2.0 for kCAL01 and 3.7 for MAST2 (Guerin et al., 2024).

Inverse Alanine Scanning and K-based Mutational Scan*: For each of the n residue in P, IAS mutates the amino acid at that residue to the 19 other amino acids, while mutating all other peptide amino acids to alanine. This generates a total of $20n$ sequences, for which DexDesign computes K* scores. IAS limits the size of the conformation space, k, by limiting the flexible residues to P's mutating residue and nearby residues on T; in practice, the median k for CALP was 3705 conformations and for MAST2 8580 conformations (Guerin et al., 2024). A mutation that IAS predicts to ablate peptide:target binding affinity is pruned from further consideration in the K*-based Mutational Scan. With an amino acid library of size a, the number of possible peptide sequences is a^n. In contrast, IAS can reduce the sequence space to $(a - R)^{n-r}$, leading to an exponential reduction in the number of sequences. In the case of CALP and MAST2, R and r are on average 16.2 and 2.6, respectively, and DexDesign reduces the number of possible peptide sequences by a factor of 1.4×10^{-4} (Guerin et al., 2024). By using bounded partition function sizes and sparse residue interaction graphs (Jou et al., 2016; Lilien et al., 2005), we can compute the K* score in time $O(nw^2q^{\frac{3}{2}w} + kn \log q)$, where w is the branch width, q is the number of rotamers per residue, and k is the number of conformations in a partition function. When w is O(1), this is polynomial time (Jain et al., 2017; Jou et al., 2016). For the DexDesign experiments we describe, we found that our ε-accurate algorithms (Donald, 2011; Gainza et al., 2012; Georgiev et al., 2008) completed with a median time of 5.4 minutes over all designs and were therefore fast enough for a preclinical pipeline (Guerin et al., 2024).

6. SUMMARY

We provide the DexDesign algorithm for the *de novo* design of peptides and proteins containing noncanonical amino acids. While this article outlines a procedure for D-peptides, DexDesign is a general algorithm for designing peptides containing noncanonical amino acids to bind to L-protein targets. OSPREY is free and open source, and we encourage others to design noncanonical peptides for affinity to diverse biochemical systems. Algorithms for incorporating L and D residues on the same chain are currently under development. Potential future applications include designing novel antifungal, antimicrobial, antineoplastic, or antibiotic D-peptides.

ACKNOWLEDGMENTS

The authors thank all members of the Donald lab for helpful discussions and the National Institutes of Health (grants R35-GM144042 to B.R.D. and R01-AI139216 to P.Z.) for funding.

SOFTWARE AVAILABILITY

The DexDesign source code is available at https://github.com/donaldlab/OSPREY3. The compiled code is available at https://donaldlab.cs.duke.edu/osprey.versions.php. The Protein Design Plugin for visualizing steric clashes is available at github.com/donaldlab/ProteinDesignPlugin.

AUTHOR DISCLOSURE STATEMENT

B.R.D. is a founder of Ten63 Therapeutics, Inc. N.G. is employed by Ten63 Therapeutics, Inc. All other authors have no conflict of interest.

FUNDING INFORMATION

We received funding from the NIH (grants R35-GM144042 to B.R.D. and R01-AI139216 to P.Z.).

SUPPLEMENTARY MATERIAL

Supplementary Video S1
Supplementary Video S2
Supplementary Video S3
Supplementary Video S4
Supplementary Video S5
Supplementary Video S6

REFERENCES

Angelini A, Cendron L, Chen S, et al. Bicyclic peptide inhibitor reveals large contact interface with a protease target. ACS Chem Biol 2012;7(5):817–821; doi: 10.1021/cb200478t

Benkirane N, Friede M, Guichard G, et al. Antigenicity and immunogenicity of modified synthetic peptides containing D-Amino acid residues. Antibodies to a D-Enantiomer do recognize the parent L-Hexapeptide and reciprocally. J Biol Chem 1993;268(35):26279–26285.

Bhardwaj G, Mulligan VK, Bahl CD, et al. Accurate de novo design of hyperstable constrained peptides. Nature 2016; 538(7625):329–335; doi: 10.1038/nature19791

Case DA, Aktulga HM, Belfon K, et al. AmberTools. J Chem Inf Model 2023;63(20):6183–6191; doi: 10.1021/acs .jcim.3c01153

Chen C-Y, Georgiev I, Anderson AC, et al. Computational structure-based redesign of enzyme activity. Proc Natl Acad Sci U S A 2009;106(10):3764–3769; doi: 10.1073/pnas.0900266106

Craik DJ, Fairlie DP, Liras S, et al. The future of peptide-based drugs. Chem Biol Drug Des 2013;81(1):136–147; doi: 10.1111/cbdd.12055

Di L. Strategic approaches to optimizing peptide ADME properties. AAPS J 2014;17(1):134–143; doi: 10.1208/s12248-014-9687-3

Donald BR. Algorithms in Structural Molecular Biology. MIT Press; 2011.

Elkin CD, Zuccola HJ, Hogle JM, et al. Computational design of D-Peptide inhibitors of hepatitis delta antigen dimerization. J Comput Aided Mol Des 2000;14(8):705–718; doi: 10.1023/a:1008146015629

Frey KM, Georgiev I, Donald BR, et al. Predicting resistance mutations using protein design algorithms. Proc Natl Acad Sci U S A 2010;107(31):13707–13712.

Gainza P, Roberts KE, Donald BR. Protein design using continuous rotamers. PLoS Comput Biol 2012;8(1):e1002335; doi: 10.1371/journal.pcbi.1002335

Garton M, Sayadi M, Kim PM. A computational approach for designing D-Proteins with non-canonical amino acid optimised binding affinity. PLoS One 2017;12(11):e0187524; doi: 10.1371/journal.pone.0187524

Georgiev I, Lilien RH, Donald BR. 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. J Comput Chem 2008;29(10):1527–1542; doi: 10.1002/jcc.20909

Guerin N, Childs H, Zhou P, et al. DexDesign: An OSPREY-based algorithm for designing de Novo D-Peptide inhibitors. Protein Eng Des Sel 2024;37 Accepted, In Press.

Hallen MA, Donald BR. CATS (Coordinates of Atoms by Taylor Series): Protein design with backbone flexibility in all locally feasible directions. Bioinformatics 2017;33(14):i5–i12.

Hallen MA, Donald BR. 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. J Comput Biol 2016;23(5):311–321.

Hallen MA, Donald BR. Protein design by provable algorithms. Commun ACM 2019;62(10):76–84; doi: 10.1145/3338124

Hallen MA, Keedy DA, Donald BR. Dead-End elimination with perturbations (DEEPer): A provable protein design algorithm with continuous sidechain and backbone flexibility. Proteins Struct Funct Bioinforma 2013;81(1):18–39.

Hallen MA, Martin JW, Ojewole A, et al. OSPREY 3.0: Open-source protein redesign for you, with powerful new features. J Comput Chem 2018;39(30):2494–2507; doi: 10.1002/jcc.25522

Holt GT, Gorman J, Wang S, et al. Improved HIV-1 Neutralization breadth and potency of V2-apex antibodies by in silico design. Cell Rep 2023;42(7):112711; doi: 10.1016/j.celrep.2023.112711

Jain S, Jou JD, Georgiev IS, et al. A critical analysis of computational protein design with sparse residue interaction graphs. PLoS Comput Biol 2017;13(3):e1005346; doi: 10.1371/journal.pcbi.1005346

Jou JD, Guerin N, Roberts KE. Protein Design Plugin. 2023.

Jou JD, Holt GT, Lowegard AU, et al. Minimization-aware recursive K*: A novel, provable algorithm that accelerates ensemble-based protein design and provably approximates the energy landscape. J Comput Biol 2020;27(4):550–564; doi: 10.1089/cmb.2019.0315

Jou JD, Jain S, Georgiev IS, et al. BWM*: A novel, provable, ensemble-based dynamic programming algorithm for sparse approximations of computational protein design. J Comput Biol 2016;23(6):413–424; doi: 10.1089/cmb.2015.0194

Kabsch W. A discussion of the solution for the best rotation to relate two sets of vectors. Acta Cryst A 1978;34(5):827–828; doi: 10.1107/S0567739478001680

Kabsch W. A solution for the best rotation to relate two sets of vectors. Acta Cryst A 1976;32(5):922–923; doi: 10.1107/S0567739476001873

Lilien RH, Stevens BW, Anderson AC, et al. 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. J Comput Biol 2005;12(6):740–761.

Lowegard A. Novel Algorithms and Tools for Computational Protein Design with Applications to Drug Resistance Prediction, Antibody Design, Peptide Inhibitor Design, and Protein Stability Prediction. PhD Thesis. Duke University; 2019.

Miranker A, Karplus M. Functionality maps of binding sites: A multiple copy simultaneous search method. Proteins Struct Funct Bioinforma 1991;11(1):29–34; doi: 10.1002/prot.340110104

Nisonoff H. Efficient Partition Function Estimation in Computational Protein Design: Probabalistic Guarantees and Characterization of a Novel Algorithm. Bachelor's Thesis. Duke University; 2015.

Noether E. Gesammelte Abhandlungen–Collected Papers. 1st ed. Springer Collected Works in Mathematics. Springer Berlin, Heidelberg: Berlin; 1983.

Ojewole AA, Jou JD, Fowler VG, et al. 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. J Comput Biol 2018;25(7):726–739; doi: 10.1089/cmb.2017.0267

Reeve SM, Gainza P, Frey KM, et al. Protein design algorithms predict viable resistance to an experimental antifolate. Proc Natl Acad Sci U S A 2015;112(3):749–754.

Renfrew PD, Choi EJ, Bonneau R, et al. Incorporation of noncanonical amino acids into rosetta and use in computational protein-peptide interface design. PLoS One 2012;7(3):e32637; doi: 10.1371/journal.pone.0032637

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

Stevens BW, Lilien RH, Georgiev I, et al. Redesigning the PheA domain of Gramicidin synthetase leads to a new understanding of the Enzyme's mechanism and selectivity. Biochemistry 2006;45(51):15495–15504; doi: 10.1021/bi061788m

Wang S. 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. Duke University; 2021.

Wang S, Reeve SM, Holt GT, et al. Chiral evasion and stereospecific antifolate resistance in *Staphylococcus aureus*. PLoS Comput Biol 2022;18(2):e1009855; doi: 10.1371/journal.pcbi.1009855

Word JM, Lovell SC, LaBean TH, et al. Visualizing and quantifying molecular goodness-of-fit: Small-probe contact dots with explicit hydrogen atoms1J Mol Biol 1999;285(4):1711–1733; doi: 10.1006/jmbi.1998.2400

Zhou J, Grigoryan G. Rapid search for tertiary fragments reveals protein sequence–structure relationships. Protein Sci 2015;24(4):508–524; doi: 10.1002/pro.2610

Address correspondence to:
Prof. Bruce R. Donald
Department of Chemistry
Duke University
Durham
NC 27708
USA

E-mail: brd+recomb24@cs.duke.edu