Chapter 3: Computational Methodology

Table of Contents

[3.1 Abstract 3](#_Toc169719654)

[3.2 Introduction 4](#_Toc169719655)

[3.3 Protein Design Algorithm 6](#_Toc169719656)

[3.3.1 Analysis of membrane protein PDBs 6](#_Toc169719657)

[3.3.2 Defining the Interface 8](#_Toc169719658)

[3.3.3 Developing the energy terms 9](#_Toc169719659)

[3.3.4 Sequence Search 11](#_Toc169719660)

[3.3.5 Backbone Refinement 12](#_Toc169719661)

[3.3.6 Mutating the interface 13](#_Toc169719662)

[3.4 Analysis 15](#_Toc169719663)

[3.4.1 Software 15](#_Toc169719664)

[3.4.2 Design Analysis 15](#_Toc169719665)

[3.4.3 Fluorescence Reconstruction 16](#_Toc169719666)

[3.4.4 TOXGREEN Conversion 17](#_Toc169719667)

[3.4.5 Determining proper membrane insertion 18](#_Toc169719668)

[3.4.6 Identifying proteins associating by designed interface 19](#_Toc169719669)

[3.4.7 Comparison to energetics 19](#_Toc169719670)

[3.4.8 Hydrogen bond mutations 20](#_Toc169719671)

[3.5 Conclusion 21](#_Toc169719672)

[3.6 Supplementary Info 22](#_Toc169719673)

[3.8 References 25](#_Toc169719674)

3.1 Abstract

With recent advancements in experimental techniques, researchers are more easily able to design experiments to retrieve high-throughput data. In conjunction with high-throughput data, computational tools and software have been invented to complement high-throughput data collection. However, understanding how to use and interpret these computational tools is a necessary skill to properly interpret and analyze high-throughput data. An important approach to designing and distributing these computational tools is to explain how these technologies work to effectively share them with the research community (Greener et al., 2022; Mougeot et al., 2022; Na, 2020; van Iterson et al., 2012). The ability to assess protein structures allows researchers to determine how and why proteins fold with unique structural features essential for function. Many computational algorithms have been developed as tools to predict and design unknown proteins using information extracted from solved structures. In this section, I discuss in detail how I created my protein design algorithm and the other programs utilized in my research.

3.2 Introduction

The complex nature of the lipid bilayer makes it difficult to directly study the forces involved in MP folding. As an initial approach, researchers aim to solve MP structures to identify features important for folding. However, solving MP structures is an inherently difficult task due to the need to express and solubilize MPs for experiments. Furthermore, many of these approaches cannot determine more than a single protein structure, making it difficult to understand dynamic structural changes involved in folding.

The protein data bank (PDB) was established to collaborate and share protein structures globally online. This tool lets researchers deposit solved protein structures for others to access and evaluate their findings (Berman et al., 2000). Initially, proteins structures were solved primarily using x-ray crystallography: X-ray crystallography has contributed to solving ~80% of MP structures (Kermani, 2021). However, producing crystals at high resolution remains a difficulty in crystallography. To bypass the need to crystallize proteins, many other MP structures have been solved by nuclear magnetic resonance (NMR). Solid-state NMR has bypassed the need for detergents in crystallography, obtaining structures of MPs less than 50 residues within lipid bilayers or nanodiscs (Liang & Tamm, 2016).

More recently cryo-EM has been used to solve MP structures. In addition to bilayers and nanodiscs, it is possible to solubilize and obtain the structures of MPs within detergents, saposin-lipoprotein nanoparticles, amphipols, and peptidiscs (Januliene & Moeller, 2021). Cryo-EM enables MP structures to be studied in a large variety of different environments, giving researchers the ability to study alternative structures of these proteins by changing the solubilization conditions.

Despite these advancements in MP structural characterization, many of these efforts take years to ascertain conditions that yield publishable results in high resolution. MPs make up only 4.6% of structures deposited in the PDB (April 2024; PDB). Lacking MP structures to evaluate, some groups focus on using previously solved protein structures to develop computational algorithms that can predict or design unknown proteins. Built into many of these algorithms are energetic terms that estimate the contributions of biophysical forces on MP stability and folding.

Computational tools have been developed to help assess our understanding of the forces that drive MP association. These tools allow researchers to predict and design MPs *in silico* that can then be tested using the previously mentioned *in vitro* and *in vivo* experimental techniques. Molecular dynamics simulations permit researchers to use established statistical and energetic potentials to simulate MP folding over time (Karplus & Petsko, 1990; MacKerell et al., 1998). Structure prediction uses known information from previously solved structures to estimate the structure of MP folded states (Elofsson & von Heijne, 2007). Protein design strategies build on structure prediction, building unknown structures as simple model systems to assess the current understanding of MP folding (Ghirlanda, 2009). MP design to study TMH systems has been successful: peptides were engineered to associate with the TM helix of integrins and a cytokine receptor (Mravic et al., 2024; Shandler et al., 2011; Yin et al., 2007), an integral MP successfully transferred electrons across the lipid bilayer (Korendovych et al., 2010), a 4-helix bundle was designed to transport Zn2+ across the bilayer (Joh et al., 2014), and phospholamban was redesigned using packing interactions and shown to successfully fold (Mravic et al., 2019). By extracting backbone conformations from the PDB, sampling different AA conformations on the interface of a standardized backbone sequence, and assessing different conformations using known energetic functions, I designed thousands of TMHs to study in high throughput.

3.3 Protein Design Algorithm

To investigate the impact of van der Waals packing on membrane protein association, I opted for a high-throughput design approach. I created a sequence search algorithm that designs 1000s of homodimer membrane protein structures using MSL v. 1.1, an open source C++ library that is freely available at http://msl-libraries.org (Kulp et al., 2012). I coupled this algorithm with a structural backbone refinement program also built in MSL. Below, I detail the algorithm alongside experiments and tests that aided in its development **(plan to add more figures in next draft)**.

3.3.1 Analysis of membrane protein PDBs

A diagram of a protein distribution

Description automatically generated with medium confidence

Using a program created in MSL, I extracted backbone geometries from all unique MPs found in the Orientations of Membrane Proteins (OPM) database (Lomize et al., 2006). This program reads an input PDB structure, identifies helical subunits of the protein composed of at least 13 AAs, and measures the distances between Cα carbons. Any two helices with at least 3 Cα carbons within 9Å of each other are extracted as individual helical pairs. To ensure that I did not extract helical pairs from redundant structures, the MPs from OPM were trimmed by sequence similarity. The PDB clusters protein sequences weekly by AA sequence similarity using MMseqs2 (Steinegger & Söding, 2017). I trimmed the OPM dataset for unique structures based on 30% sequence similarity from the PDB and compiled the geometric data. I then extracted two parameters the crossingAngle and xShift, which were plotted against each other for each protein and analyzed using kernel density estimation (figure).

The density plot identifies the most common interaction motifs for dimeric proteins. We expected the regions of highest density to be the most likely designable, as they are most often found in nature. There are three high density regions present in this dataset. The first region is present in the positive crossingAngle, or left-handed, region. Helical pairs interact frequently in the range between 8.5 to 10 Å of interhelical distance and 20 to 40° of crossing angle. We refer to this as the left-handed design region (Left). The other two high density regions are found in the negative crossingAngle, or right-handed, region. Helical pairs are found at a much broader range of distances from 6.5 to over 10 Å of interhelical distance. The region with the most density is found between 7.75 to 9.5 Å interhelical distance and -30 to -60° crossingAngle. We refer to this as the right-handed design region (Right). Finally, the third region corresponds to GASright, which is characterized by very short interhelical distance (6.5-7.5 Å) and crossing angles of -25 to -55°. The GASright ­is a well characterized dimerization motif that is stabilized by a combination of van der Waals packing and the formation of interhelical weak hydrogen bonds between helices (Anderson et al., 2017; Mueller et al., 2014). We decided to design this region as a control to analyze differences in stability between proteins stabilized solely by van der Waals packing and GASright stabilized by both packing and interhelical hydrogen bonding.

To determine appropriate axialRotations and zShifts for protein design, I used MSL to set proteins at a variety of backbone templates. I created a grid of the geometric templates for each design region, resulting in ~10000 geometries that were input into MSL and assessed for clashing at the interface. I chose to use a poly-Ala sequence for each template, as clashing at the interface of the small AA Ala would suggest the geometry is unable to accommodate other AAs. I then plotted any geometries that did not clash on a density map and extracted the ranges of axialRotations and zShifts **(S1, S2, S3)**. Finally, I randomly generated 1000s of geometries for each design region, where the crossingAngle and xShift is chosen from the membrane protein density map, and the axialRotation and zShift is chosen from the identified ranges where clashing did not occur in poly-Ala sequences. These geometries were used as input backbone templates for protein design.

3.3.2 Defining the Interface

A video game screen with a black background

Description automatically generatedTo reduce heterogeneity in expression, I designed the interface of a standardized TM helix of 21 AAs consisting of a poly-Leu backbone, a strategy previously applied to study the association of GASrights (Anderson et al., 2017). In my first protein design run, I used Solved Accessible Surface Area (SASA) to identify the interfacial positions of a protein set at a geometry from my membrane protein analysis. The SASA was calculated for each position on the dimer and the interface defined as the 8 positions with the least amount of access to the solvent, or the most buried positions. After sorting these designs and looking at the fluorescence results, variations in the interface were seen to increase the complexity of the data **(Supplement?)**. Although we were successfully able to design sequences that associate, the population was small and our energy score showed little correlation to association.

For the subsequent design runs, I standardized the interface for each region. The GASright was previously studied using similarly standardized backbones, so we chose to apply the same 8 positions as the interface. Since Right designs have similar geometric parameters outside of a larger xShift, I used the same 8 interfacial positions as GASright. For the Left designs, I visually inspected a variety of sequences with unique crossing angles and defined 3 interfaces. These interfaces accounted for known left-handed association motifs such as the leucine zipper, knobs-into-holes, and coiled coils (Ash et al., 2004; Bornberg-Bauer et al., 1998; Walshaw & Woolfson, 2003). Standardizing the interfaces permitted me to come up with a consistent mutational strategy to assess my proteins for their association at the given interface that is discussed in detail in section 3.4.6.

3.3.3 Developing the energy terms

To predict the energy of my designed proteins, I applied the same set of energy terms used previously by CATM: CHARMM\_VDW, SCWRL4\_HBOND, and CHARMM\_IMM1. These terms are computed for each protein during the sequence search to find the best interfacial sequence for the input geometric structure. To determine the stability of our designed dimers, we predict the interaction energies of the dimer structure and two monomers, and then subtract the monomer energy from the dimer energy (Dimer-Monomer). However, computing the monomer energy for each sequence initially created a bottleneck in our algorithm, limiting the number of sequences we could design. To account for this, I developed an energy term that estimates the monomer energy of each sequence: BASELINE\_MONOMER.

A graph with a line graph

Description automatically generated with medium confidence

The BASELINE\_MONOMER term was created by measuring the energetic contributions from each individual amino acid on a monomeric helix. These energy contributions were measured using the terms from CATM. I computed the energetics of 10000 random sequences and measured the self and pair energies for each amino acid. The self-energy represents the energy contribution for an individual amino acid to the protein stability alone, while the pair energies represent the energy contribution between any two interacting amino acids. I measured the pair energies for all AA pairs on the sequence and found that there was no longer a pair interaction between amino acids more than 10 away from each other. From each iteration, I calculated the average of all self and pair energies and saw a strong correlation between the measured monomer energy and the BASELINE\_MONOMER term (figure). This BASELINE\_MONOMER term was made for the subset of amino acids used in design (G, A, L, I, M, S, T, W, Y, F) and would need to be rerun to establish the term for any additional amino acids. This term was helpful in decreasing computational time, enabling us to design 1000s of sequences within a week.

Another issue that I encountered was that many of my initial designs were often composed of only 2-3 different AAs. This result could impact our protein expression and insertion, as natural membrane protein sequences are often made of a diversity of AAs. To account for this sequence diversity, I developed a SEQUENCE\_ENTROPY term that outputs an energy based on how similar a sequence is to a natural membrane protein sequence. The frequency of each AA in membrane protein sequences was calculated from the sequences extracted from membrane protein PDBs. To convert the frequency of AAs in a sequence to an energy term, I utilized the following equation based on the Boltzmann entropy formula:

where R is the gas constant and T is temperature defaulted to 298K (RT = 0.592). To compute the sequence entropy, I needed to calculate the probability that the sequence occurs in membrane. First, the number of each AA (#AA) is counted within the sequence. Using these values, I then calculated the number of possible permutations for the sequence. This is determined using the following equation:

where n is the number of positions, which is divided by the factorial for #AA in the sequence multiplied, or the total number of combinations possible. The probability is computed using the frequency of each AA in membrane protein sequences (freq\_AA) to the power of the number of each AA in the sequence multiplied by the permutations:

This probability is inserted into the sequence entropy equation, returning a value that can be applied as an energy term for each sequence. My algorithm utilizes this term as an additional energy, predicting the stability of my designed homodimer sequences and adjusting the energy by the likelihood that they are found in natural membrane proteins sequences.

3.3.4 Sequence Search

For each input geometry, the algorithm first defines the best sequence using the Self Consistent Mean Field (SCMF) theory as implemented in MSL. This method estimates the conformational entropy of each design AA as a probability that it is present within the dimer interface (Koehl & Delarue, 1994). The sequence from the SCMF is then run through a MC sequence optimization: A random position on the interface is switched to a random AA, and the energy is calculated using the combination of the CATM energies and the energies defined in section 3.4.3.

Each energy term can be multiplied by an optional weight, meaning the total energy can be changed proportionately by the given weights. Previous research in our lab defaulted the weights of the CATM energy terms to 1, which was repeated in this study. To determine if SEQUENCE\_ENTROPY performed better at different weights, I ran a design test at weights of 1, 5, 10, 50, and 100 and compared the AA composition in these designs with the frequency found in membrane proteins. Weights greater than 10 were found to be optimal for mimicking the frequency of AAs in membrane proteins. We chose 10 as it resulted in the SEQUENCE\_ENTROPY term affecting the total energy at the same order of magnitude as the other terms. The equation for the total energy is as follows:

where the dimer energy is composed of CHARMM\_VDW, SCWRL4\_HBOND, and CHARMM\_IMM1.

The total energy is used for the acceptance criteria, and the MC searches for a multitude of sequences before reaching an energetic minimum. The sequences are saved in an output file to show the search trajectory of sequences along with their corresponding energy terms. The interface sequence with the best interaction energy is saved as the design undergoes further backbone refinement. **(figure here? Maybe should also add in my MC parameters somewhere)**

3.3.5 Backbone Refinement

Backbone refinement aims to minimize the energy of the sequence and structure. After initially starting with a specific geometric template, the newly designed sequence undergoes an MC based structural refinement procedure. The structure undergoes MC backbone perturbations, where one of the four inter-helical parameters (xShift, crossingAngle, axialRotation, zShift) is chosen and shifted during each cycle. The total energy for the refined structure is used as the acceptance criteria, with the BASELINE\_MONOMER energy being replaced by the computed monomer energy and the SEQUENCE\_ENTROPY term no longer applied:

The backbone refined geometry can be input into the sequence search to find other designable sequences.

3.3.6 Mutating the interface

For each protein, I made point mutations on the interface corresponding to interfacial positions expected to decrease association. I created a program that accepts the structure and interface as inputs, individually mutates each interfacial position to a given amino acid, and outputs the energy and structure. The first mutation is a clash mutant, where the interface was mutated to an Isoleucine, which often can kink into the opposing helix, often disrupting the ability for a protein to associate. The second mutation is a void mutant, where the interface was mutated to the small AA Alanine, aiming to decrease the amount of packing at the interface. We expect these mutants to enable us to determine if our proteins are **A graph showing the difference between a refinement and a new

Description automatically generated**associating by the designed interface.

I also created a program to predict the energetics of the chosen mutants for each of my structures. The program reads an input structure and sequence. To ensure that the mutated sequence is able to find alternate dimer interfaces, the structure undergoes a more detailed MC backbone refinement. In the backbone refinement designed in CATM, each geometric shift randomly alters the structure by choosing a value between 0 and an input upper limit. For example: When the geometric xShift is chosen, a random value from 0-0.5 Å is applied to the structure. In this version, the geometric term procedurally decreases to a lower limit. Each cycle, the chosen geometric term is decreased by multiplying it by the metropolis criteria until it reaches the lower limit. Once the lower limit is reached, this value will always be chosen whenever this term is shifted again. After initially testing this process on my designs, I found that the new backbone refinement resulted in more stable energies for my designed proteins (figure). The energetics from this refinement is utilized to evaluate my structures and mutants against their reconstructed fluorescence in sort-seq.

3.4 Analysis

3.4.1 Software

The following calculations, analyses, and graphing were implemented and performed using Python v. 2.7. Relevant packages include:

Pandas: (McKinney, 2011)

DNAChisel: (Zulkower & Rosser, 2020)

Numpy: (McKinney, 2012)

Matplotlib: (Tosi, 2009)

Seaborn: (Waskom, 2021)

Scipy: (Virtanen et al., 2020)

Sklearn: (Pedregosa et al., 2011)

Pymol: (DeLano, 2002)

All programs and code can be found on Github.

3.4.2 Design Analysis

During my first design run, I encountered an issue when visually inspecting my poly-Leucine standardized sequences: interfaces often included voids to accommodate Leucine at the termini, preventing clashing interactions. Although these designs had a considerable amount of van der Waals packing, we wanted to ensure that the interface was driving association. I chose to repeat my design process with the smaller AA Alanine at the termini. These structures were found to include less voids and a well packed interface (figure?). Although the termini are unchanged in our experiment, we included these Alanine termini designs in our dataset with the assumption that helices in the experiment would be more flexible to accommodate these interfaces than our rigid helices.

To analyze my designs, I wrote a script that compiles all design directories and outputs a variety of plots. These plots include energetics bar graphs separated by each design region and AA percent composition **(S4)**. These sequences alongside their structures can then be input into the mutation program to identify mutations expected to decrease association. Finally, these designed sequences, their respective mutants, and a variety of control sequences were ordered in an oligo pool library from Twist Bioscience and cloned into plasmids for TOXGREEN sort-seq.

3.4.3 Fluorescence Reconstruction

As detailed previously in (cite sort-seq), plasmids obtained from sorted populations of *E. coli* can be sent for Next Generation Sequencing (NGS) and the sequencing results reconstructed to fluorescence corresponding to dimerization propensity. Briefly, reconstructed fluorescence levels were calculated as a weighted average (Kosuri et al., 2013). This method normalizes the reads per protein per bin with the fraction of the population found in that bin. The normalized fractional contribution of each bin (j) for each protein (i), aij is calculated as:

A diagram of a function

Description automatically generatedA mathematical equation with black letters

Description automatically generated with medium confidence

where the normalized fractional contribution is then multiplied by the median fluorescence of that bin (mj­). Finally, the contributions for each sorted bin are summed to determine the reconstructed fluorescence. The sequencing data is run through a Python pipeline that computes the above reconstructed fluorescence for all sequences found in the NGS data. The reconstruction resulted in 853 of our 1075 designs (79%) present within each replicate of the NGS data.

3.4.4 TOXGREEN Conversion

Studies using TOXCAT systems typically normalize the dimerization propensity by comparing dimerization propensity to the well-studied strong dimer Glycophorin A (GpA), which is included as a control in each of our libraries. This normalization is calculated as a percentage of GpA fluorescence (%GpA). The reconstructed fluorescence is converted to %GpA using the following equation:

To calibrate our fluorescence properly to %GpA, we also include a variety of control sequences that we have previously studied using TOXGREEN. Upon initial inspection, the controls present in our experiment were reconstructed to a lower %GpA value than we’ve previously found in TOXGREEN.

I conducted a separate low-throughput TOXGREEN experiment on the control sequences and a subset of my designed sequences. When I compared the TOXGREEN to the reconstructed data, the reconstructed values were noticeably smaller for most sequences. However, when we plotted TOXGREEN versus the reconstruction, we found a clear correlation between the two datasets (figure). To properly calibrate our reconstruction to the results we see in TOXGREEN, we converted the reconstruction data to TOXGREEN. To adjust the reconstructed fluorescence value to TOXGREEN, we apply the equation found by the correlation line. The reconstructed fluorescence is multiplied by the slope and then subtracted by the y-intercept of the correlation. The values for each sequence are averaged with their corresponding replicates and normalized to the GpA sorted in each design population. This conversion let us differentiate between different levels of dimerization propensity as seen previously in TOXGREEN experiments: monomers (0-35%), weak dimers (35-60%), and strong dimers (>60%).

A group of graphs with lines

Description automatically generated with medium confidence

3.4.5 Determining proper membrane insertion

A liquid maltose growth assay was run in triplicate as in (cite sort-seq). Briefly, cultures composed of *E. coli* containing plasmids for the designs, mutants, and controls are grown overnight. These are normalized by OD600 in the morning and the normalized population added to flasks of liquid maltose media **(say what composed of?)** and grown for 36H, with timepoints taken every 6 hours. The populations for each timepoint are spun down, plasmids extracted through miniprep, and prepared for NGS. Within each population are control sequences that are known not to insert in the membrane, as shown by failure to grow on maltose plates. To assess whether the sequences properly insert into the membrane, we compare the relative abundance of our designs to these controls. The relative abundance from the overnight growth (0H) and the growth in liquid maltose at 30H to determine the ability to insert:

Sequences that are more abundant than these controls are considered properly inserted. 489 of the 853 designs present (57%) pass our insertion test. **(could create a supplemental figure here if needed similar to SMA maltose figures)**

3.4.6 Identifying proteins associating by designed interface

To determine if sequences dimerize along the designed interface, we sought to identify sequences where the mutant results in a significant decrease in association. First, we analyzed the fluorescence for the clash and void mutants against the fluorescence of our designed sequences. When comparing the clash mutations to the fluorescence of the design, we saw a significant decrease in their association. However, the void mutants did not show this same decrease, often resulting in similar fluorescence as the WT designs. This data suggests that mutating for loss of packing does not impact association, possibly as a result of dimerization by an alternate interface. We decided to move forward by trimming our data using the clashing mutants, which appear to disrupt association by our designed interface **(boxplot figure from paper)**. We trimmed our data for any designs where the clashing mutation was monomeric (< 35% GpA) or less than 50% of the design fluorescence in terms of %GpA. This resulted in 248 out of the 853 designs present (29%) associate by our designed interface.

3.4.7 Comparison to energetics

We plotted the energy score against the dimerization propensity in fluorescence for each protein and separated the data by design region. Unfortunately, our energy score does not correlate well to proteins outside of the GASright region **(figure from paper)**. This data suggests that although were able to design sequences that associate (>35% GpA), we are unable to predict their dimerization propensity using our energetics. However, a majority of our designs outside of the GASright associate as weak dimers (35-60% GpA). This suggests that our energetics may not be well tuned to predict weakly dimerizing proteins. I discuss how this can be addressed in future experiments in Chapter 4.

3.4.8 Hydrogen bond mutations

In our design procedure, we included AAs that had the potential to hydrogen bond due to how frequently they are found in membrane proteins. Our design energies predicted low levels of hydrogen bonding, suggesting that they wouldn’t impact association. However, because our energetics show little correlation with experimental dimerization propensity, we are unable to confidently state that our sequences associate solely by van der Waals packing without the influence of other forces like hydrogen bonding.

To ascertain if our proteins associate solely by packing, I conducted an experiment where we mutated our proteins to remove the potential for hydrogen bonding. To identify proteins with the potential for hydrogen bonding, I wrote a Python script that searches through my protein structures and identifies any oxygen atoms within 3 Å, a generous threshold for potential hydrogen bond formation. I identified 17 proteins that associated from mildly weak to strong dimers (>40% GpA) with the potential for at least 1 hydrogen bond. I mutated all hydrogen bonding AAs in these sequences to hydrophobic AAs with similar steric bulk: Thr->Val, Tyr->Phe, and Ser->Ala. I ordered the original design sequences, their respective clash mutants, and the hydrophobic mutants as gblocks from Twist Bioscience and successfully cloned 13/17 proteins into the TOXGREEN plasmid for experiments. **(explain either TOXGREEN and/or sort-seq after data comes back)**

3.5 Conclusion

A group of blue squares

Description automatically generated3.6 Supplementary Info

A screenshot of a computer screen

Description automatically generated

A grid of blue squares

Description automatically generated

A screenshot of a graph

Description automatically generated

3.7 References

Anderson, S. M., Mueller, B. K., Lange, E. J., & Senes, A. (2017). Combination of Cα-H Hydrogen Bonds and van der Waals Packing Modulates the Stability of GxxxG-Mediated Dimers in Membranes. *J Am Chem Soc*, *139*(44), 15774-15783. <https://doi.org/10.1021/jacs.7b07505>

Ash, W. L., Stockner, T., MacCallum, J. L., & Tieleman, D. P. (2004). Computer modeling of polyleucine-based coiled coil dimers in a realistic membrane environment: insight into helix-helix interactions in membrane proteins. *Biochemistry*, *43*(28), 9050-9060. <https://doi.org/10.1021/bi0494572>

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H.,…Bourne, P. E. (2000). The protein data bank. *Nucleic acids research*, *28*(1), 235-242.

Bornberg-Bauer, E., Rivals, E., & Vingron, M. (1998). Computational approaches to identify leucine zippers. *Nucleic Acids Research*, *26*(11), 2740-2746. <https://doi.org/10.1093/nar/26.11.2740>

DeLano, W. L. (2002). Pymol: An open-source molecular graphics tool. *CCP4 Newsl. Protein Crystallogr*, *40*(1), 82-92.

Elofsson, A., & von Heijne, G. (2007). Membrane protein structure: prediction versus reality. *Annu Rev Biochem*, *76*, 125-140. <https://doi.org/10.1146/annurev.biochem.76.052705.163539>

Ghirlanda, G. (2009). Design of membrane proteins: toward functional systems. *Current Opinion in Chemical Biology*, *13*(5), 643-651. <https://doi.org/https://doi.org/10.1016/j.cbpa.2009.09.017>

Greener, J. G., Kandathil, S. M., Moffat, L., & Jones, D. T. (2022). A guide to machine learning for biologists. *Nat Rev Mol Cell Biol*, *23*(1), 40-55. <https://doi.org/10.1038/s41580-021-00407-0>

Januliene, D., & Moeller, A. (2021). Single-Particle Cryo-EM of Membrane Proteins. *Methods Mol Biol*, *2302*, 153-178. <https://doi.org/10.1007/978-1-0716-1394-8_9>

Joh, N. H., Wang, T., Bhate, M. P., Acharya, R., Wu, Y., Grabe, M.,…DeGrado, W. F. (2014). De novo design of a transmembrane Zn²⁺-transporting four-helix bundle. *Science*, *346*(6216), 1520-1524. <https://doi.org/10.1126/science.1261172>

Karplus, M., & Petsko, G. A. (1990). Molecular dynamics simulations in biology. *Nature*, *347*(6294), 631-639. <https://doi.org/10.1038/347631a0>

Kermani, A. A. (2021). A guide to membrane protein X-ray crystallography. *FEBS J*, *288*(20), 5788-5804. <https://doi.org/10.1111/febs.15676>

Koehl, P., & Delarue, M. (1994). Application of a Self-consistent Mean Field Theory to Predict Protein Side-chains Conformation and Estimate Their Conformational Entropy. *239*(2), 249-275.

Korendovych, I. V., Senes, A., Kim, Y. H., Lear, J. D., Fry, H. C., Therien, M. J.,…Degrado, W. F. (2010). De novo design and molecular assembly of a transmembrane diporphyrin-binding protein complex. *J Am Chem Soc*, *132*(44), 15516-15518. <https://doi.org/10.1021/ja107487b>

Kosuri, S., Goodman, D. B., Cambray, G., Mutalik, V. K., Gao, Y., Arkin, A. P.,…Church, G. M. (2013). Composability of regulatory sequences controlling transcription and translation in Escherichia coli. *Proc Natl Acad Sci U S A*, *110*(34), 14024-14029. <https://doi.org/10.1073/pnas.1301301110>

Kulp, D. W., Subramaniam, S., Donald, J. E., Hannigan, B. T., Mueller, B. K., Grigoryan, G., & Senes, A. (2012). Structural informatics, modeling, and design with an open-source Molecular Software Library (MSL). *J Comput Chem*, *33*(20), 1645-1661. <https://doi.org/10.1002/jcc.22968>

Liang, B., & Tamm, L. K. (2016). NMR as a tool to investigate the structure, dynamics and function of membrane proteins. *Nat Struct Mol Biol*, *23*(6), 468-474. <https://doi.org/10.1038/nsmb.3226>

Lomize, M. A., Lomize, A. L., Pogozheva, I. D., & Mosberg, H. I. (2006). OPM: orientations of proteins in membranes database. *Bioinformatics*, *22*(5), 623-625.

MacKerell, A. D., Bashford, D., Bellott, M., Dunbrack, R. L., Evanseck, J. D., Field, M. J.,…Karplus, M. (1998). All-atom empirical potential for molecular modeling and dynamics studies of proteins. *J Phys Chem B*, *102*(18), 3586-3616. <https://doi.org/10.1021/jp973084f>

McKinney, W. (2011). pandas: a Foundational Python Library for Data Analysis and Statistics.

McKinney, W. (2012). *Python for data analysis: Data wrangling with Pandas, NumPy, and IPython*. " O'Reilly Media, Inc.".

Mougeot, G., Dubos, T., Chausse, F., Péry, E., Graumann, K., Tatout, C.,…Desset, S. (2022). Deep learning -- promises for 3D nuclear imaging: a guide for biologists. *J Cell Sci*, *135*(7). <https://doi.org/10.1242/jcs.258986>

Mravic, M., He, L., Kratochvil, H. T., Hu, H., Nick, S. E., Bai, W.,…DeGrado, W. F. (2024). De novo-designed transmembrane proteins bind and regulate a cytokine receptor. *Nat Chem Biol*. <https://doi.org/10.1038/s41589-024-01562-z>

Mravic, M., Thomaston, J. L., Tucker, M., Solomon, P. E., Liu, L., & DeGrado, W. F. (2019). Packing of apolar side chains enables accurate design of highly stable membrane proteins. *Science*, *363*(6434), 1418-1423. <https://doi.org/10.1126/science.aav7541>

Mueller, B. K., Subramaniam, S., & Senes, A. (2014). A frequent, GxxxG-mediated, transmembrane association motif is optimized for the formation of interhelical Cα-H hydrogen bonds. *Proc Natl Acad Sci U S A*, *111*(10), E888-895. <https://doi.org/10.1073/pnas.1319944111>

Na, D. (2020). User guides for biologists to learn computational methods. *J Microbiol*, *58*(3), 173-175. <https://doi.org/10.1007/s12275-020-9723-1>

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O.,…Dubourg, V. (2011). Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, *12*, 2825-2830.

Shandler, S. J., Korendovych, I. V., Moore, D. T., Smith-Dupont, K. B., Streu, C. N., Litvinov, R. I.,…DeGrado, W. F. (2011). Computational design of a β-peptide that targets transmembrane helices. *J Am Chem Soc*, *133*(32), 12378-12381. <https://doi.org/10.1021/ja204215f>

Steinegger, M., & Söding, J. (2017). MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, *35*(11), 1026-1028.

Tosi, S. (2009). *Matplotlib for Python developers*. Packt Publishing Ltd.

van Iterson, M., van Haagen, H. H., & Goeman, J. J. (2012). Resolving confusion of tongues in statistics and machine learning: a primer for biologists and bioinformaticians. *Proteomics*, *12*(4-5), 543-549. <https://doi.org/10.1002/pmic.201100395>

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D.,…Contributors, S. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat Methods*, *17*(3), 261-272. <https://doi.org/10.1038/s41592-019-0686-2>

Walshaw, J., & Woolfson, D. N. (2003). Extended knobs-into-holes packing in classical and complex coiled-coil assemblies. *144*(3), 349-361.

Waskom, M. L. (2021). Seaborn: statistical data visualization. *Journal of Open Source Software*, *6*(60), 3021.

Yin, H., Slusky, J. S., Berger, B. W., Walters, R. S., Vilaire, G., Litvinov, R. I.,…DeGrado, W. F. (2007). Computational design of peptides that target transmembrane helices. *Science*, *315*(5820), 1817-1822. <https://doi.org/10.1126/science.1136782>

Zulkower, V., & Rosser, S. (2020). DNA Chisel, a versatile sequence optimizer. *Bioinformatics*, *36*(16), 4508-4509.