Announcements

Python review this Friday from 3-5pm in room 300-300

If you have not filled out the course survey, please help! It is tremendously helpful for us to get your feedback so we can try to improve the course.

Thanks!
Previously....

Coevolution in sequence

Previously….

How’d we do?

Cytochrome bd oxidase (CYDA/CYDB/CYDX)
Science 2016

Fumarate hydratase (TTDA/TTDB)
PNAS 2016

Prolipoprotein diacylglyceryl transferase (LGT)
Nature 2016

Fluoride ion transporter (CRCB)
Nature 2015

lipoprotein signal peptidase II (LSPA) DMT superfamily transporter (YddG)
Science 2016

Nature 2016
Structure prediction games
FoldIt: Protein-folding game

Protein-folding is a computationally-intensive problem, let's crowd-source it.

- [https://fold.it/](https://fold.it/)
- Basic idea: allow players to optimize the Rosetta all-atom energy function
  - Game score is negative of the energy

In fold.it, players can modify protein and view progress/changes in their protein as an energy function. Players can compete with other players or the computer, to achieve the best (most optimal) energy score.
Go to Foldit
Example of human players out-perform automated methods

Each dot is a structure, structural comparison is plotted as Rosetta energy vs. RMSD to native structure. The green cloud is produced by human players, and they achieve lower energy and better RMSD.

Each dot indicates a structure conformation. Green dots indicate structures designed by human players, while yellow dots indicate computationally derived structure conformations. Human players outperform computer simulations in this case.

Common or strategic energy-minimizing moves are recorded by the developers of the folding game and built into existing protein prediction simulations.

Red: Computer model  
Blue: experimental NMR structure  
Green: best FoldIt solution
EteRNA: RNA design game

• Similar idea, but:
  – For RNA rather than protein.
  – Goal is RNA design. Users collective design RNA sequences, which are tested experimentally.
• From Rhiju Das (Stanford) and Adrien Treuille (CMU)
Problem definition

End-goal of protein design is to derive a sequence

Protein Design - deriving the sequence for a desired 3D structure and function

Structure Prediction - taking a sequence and solving structure and function
What are some (useful) examples?

- Designing enzymes (proteins that catalyze chemical reactions)
  - Useful for production of industrial chemicals and drugs
- Designing proteins that bind specifically to other proteins
  - Potential for HIV, cancer, Alzheimer’s treatment
  - Special case: antibody design
- Designing sensors (proteins that bind to and detect the presence of small molecules—for example, by lighting up or changing color)
  - Calcium sensors used to detect neuronal activity in imaging studies
  - Proteins that detect TNT or other explosives, for mine detection
- Making a more stable variant of an existing protein
  - Or a water-soluble variant of a membrane protein
Outline for today: Understanding the design work flow

Directly testing every sequence is not computationally tractable, so we simplify the design problem by considering the backbone and side chains separately.

Understand the structural context (for what purpose)
Starting from a known structure, or a topology if de novo

↓

Build a backbone

↓

Build the side chains

↓

Refine

↓

Predict the structure (if small enough)
Understand the structural context (for what purpose) Starting from a known structure, or a topology if de novo

If the goal is to bind a steroid drug, what kind of structure may we use?

How would we find this kind of structures?

How may we design it to bind?

In the context of finding a binding protein, we need to consider properties like the physical and electrochemical state of the protein that surrounds our target. For example, Digoxigen has many ring structures, which are hydrophobic (these groups are all non-polar). As a result, we would expect the binding pocket of our protein to be hydrophobic as well (like attracts like). We might also expect there to be functional groups along the protein backbone that can stabilize the -OH groups on Digoxigenin. Think functional groups (R-groups) that can form stabilizing H-bonds with the OH.

By contextualizing the purpose of our desired protein, we can constrain our design sequences by requiring certain amino acid groups (or specific residues) to reside at a given location(s) along the sequence.
Understand the structural context (for what purpose)
Starting from a known structure, or a topology if de novo

If you realize that your enzyme is not binding its substrate tightly enough...

What type of structure can one design to solve this problem?

Here, modifying the binding protein to form a more tightly-enclosed binding pocket can induce tighter/longer binding of the substrate molecule.
Understand the structural context (for what purpose)
Starting from a known structure, or a topology if \textit{de novo}

If the goal is to make a ruler for electron microscopy, how would we find or make this type of structure?

Huang & Parmeggiani
Understand the structural context (for what purpose)
Starting from a known structure, or a topology if *de novo*

If the goal is to make membrane channels of different sizes (maybe to control ion flow, perhaps to sequence DNA), how do we build this type of structures?

There are specific properties of membrane proteins, such as having hydrophobic residues along the outer mid-section of the protein, that we can encode into our sequence. One might look at existing proteins with similar function to derive these segments of the sequence.
Each of these problems is solved with a different strategy, based on how the backbone is generated.

Having provided the context for our protein, how do we actually go about building it?

- **Fixed-backbone design**: Involves searching through pdb and finding compatible, binding proteins.
- **Loop modeling**: Hold parts of the protein constant and modifying local fragments.
- **De novo design with fragment assembly**: Use fragments of proteins to piece together candidate proteins.
- **De novo design with parametric equations**: Guide protein design with math.
For the discussions today, we will again focus on Rosetta based strategies

Rosetta is by far the most power design tool, with most accumulated successful designs created to date

There are many other methods available for fixed-backbone design scenarios that only involve computations that optimizes the side chains.

But in terms of direct manipulation and construction of the backbones, Rosetta has a clear advantage — not just because of its large collection of protocols, but because of the robustness of the Rosetta energy function (a tightly designed folding funnel makes discrimination of poor structures relatively easily)
Outline for today: Understanding the design work flow

Understand the structural context (for what purpose)
Starting from a known structure, or a topology if *de novo*

Build a backbone
(fragment based or parametric eq. based)

Build the side chains

Refine

Predict the structure (if small enough)

We are going to leave out the Fixed-bb case for now, since that’s just side chain design
Thinking about protein structures (before solving structures)
(first x-ray structure of myoglobin in 1958)

THE STRUCTURE OF PROTEINS: TWO HYDROGEN-BONDED HELICAL CONFIGURATIONS OF THE POLYPEPTIDE CHAIN

By Linus Pauling, Robert B. Corey, and H. R. Branson*

Gates and Crellin Laboratories of Chemistry,
California Institute of Technology, Pasadena, California†

PNAS April 1, 1951 37 (4) 205-211
geometric requirements for amino acid backbones

\[ \omega \text{ angle} = 180^\circ \]

The electrons from the double-bonded C=O form a “resonance structure”, meaning that the electrons are also shared between the N-C bond. This means that the N-C bond has some degree of double-bond character. Double bonds do not allow for rotation, which is why the \( \omega \) angle, the rotation around the N-C bond is constrained to 180 degrees (and rarely 0 degrees).

“I was thunderstruck by Pauling and Corey's paper. In contrast to Kendrew's and my helices, theirs was \textbf{free of strain}; all of the amide groups were planar and every carbonyl group formed a perfect hydrogen bond with an imino group four residues further along the chain.”

-Max Perutz

This “free of strain” observation is the key to all current protein design methods.
Fragment based construction (this includes loop modeling and de novo design)

• An extension of the ab initio setup used for structure prediction
  How do we find these structures? They are generally provided by the user, who might, for example, specify the number of helices as a parameter.

• Secondary structure type and range is manually specified

• When fragments are only applied to a local region of a structure, it is called a “loop modeling” problem

• When fragments are applied throughout, it is a “de novo” design (e.g. an entire protein is built from scratch)

• More of an art than science
De novo protein design in action

topology: 4 x (5 strand + 3 loop + 13 helix + 3 loop + 5 strand + 3 loop + 10 helix + 3 loop)
Alternatively: making regular backbones by parametric equations, which works for helices

How can we describe backbones with parametric equations?
Ex. two alpha helices can form a coiled coil (C), where the two side chains interact uniformly when each forms 7-amino acid sequence repeats (A). Because this pattern is so regular, you can describe them using a parametric equation. Ie there are many regularly occurring patterns in proteins, that can be defined using a mathematical expression.

simple sequence rules for coiled-coils: heptad repeats

- on average 3.6 residues per turn
- This number can vary depending on the superhelical twist
- periodic heptads maintain “knobs into holes” packing pattern to form a coiled-coil structure

https://doi.org/10.1385/1-59745-187-8:35
Alternatively: making regular backbones by parametric equations, which works for helices

Derivation of coiled-coil parametric equations

The Fourier Transform of a Coiled-Coil

By F. H. C. Crick

x, y equations are essentially the polar form of the circle equation

Update z while drawing the circle, thus forming a helix

\[ x = r_0 \cos \omega_0 t, \]
\[ y = r_0 \sin \omega_0 t, \]
\[ z = P(\omega_0 t/2\pi), \]

\[ x = r_1 \cos \omega_1 t, \]
\[ y = r_1 \sin \omega_1 t, \]
\[ z = 0. \]

\[
\begin{align*}
x &= r_0 \cos \omega_0 t + r_1 \cos \omega_1 t \cdot \cos \omega_0 t - r_1 \cos \alpha \cdot \sin \omega_0 t \cdot \sin \omega_1 t, \\
y &= r_0 \sin \omega_0 t + r_1 \sin \omega_1 t \cdot \cos \omega_0 t + r_1 \cos \alpha \cdot \cos \omega_0 t \cdot \sin \omega_1 t, \\
z &= P(\omega_0 t/2\pi) - r_1 \sin \alpha \cdot \sin \omega_1 t.
\end{align*}
\]

You are not responsible for the derivations
Helical bundles can be built from a few parameters

Tweaking parametric equation parameters will result in different helix structures

α-helix parameters

$R_1, \omega_1, \varphi_1$ and $\Delta z$

Superhelical parameters

$R_0, \omega_0, \varphi_0', d$

For unstrained helices

$(\omega_0 + \omega_1) = 100^\circ$

Grigoryan et al., JMB 2011

Two cool examples of proteins created by parametric equation design

Programmable specificity

Scott Boyken, Zibo Chen, Ben Groves, Georg Seelig

You can cluster proteins based on amino acid binding partners, thus we can define what neighboring amino acids may be by drawing from a distribution of likely binding partners.

Heat-map of binding (Yeast-2-hybrid)
Outline for today: Understanding the design work flow

Understand the structural context (for what purpose)
Starting from a known structure, or a topology if *de novo*

Build a backbone

Build the side chains

Refine

Predict the structure (if small enough)
The sequence design problem is actually really hard

- $20^N$ possible sequences with $N$ residues
- Given an energy function, what we really want is to find the sequence that maximize the probability of the desired structure compared to all other possible folded and unfolded structures

We would also like the sequence to minimize the energy function, such that the resulting structure is optimally-stable, ie does not fold into other states

Given the computationally-intensive nature of the problem, stochastic methods, specifically Monte-Carlo, are used to approximate the solution to this problem

For those who are interested ->

Protein Engineering vol.15 no.10 pp.779–782, 2002

Protein Design is NP-hard
Niles A.Pierce$^{1,2}$ and Erik Winfree$^3$

$^1$Applied and Computational Mathematics and $^3$Computer Science and Computation and Neural Systems, California Institute of Technology, Pasadena, CA 91125, USA

$^2$To whom correspondence should be addressed.
E-mail: niles@caltech.edu

Biologists working in the area of computational protein design have never doubted the seriousness of the algorithmic challenges that face them in attempting in silico sequence selection. It turns out that in the language of the computer science community, this discrete optimization problem is NP-hard. The purpose of this paper is to explain the context of this observation, to provide a simple illustrative proof and to discuss the implications for future progress on algorithms for computational protein design.

Keywords: complexity/design/NP-complete/np-hard/proteins
We can dramatically simplify this problem by making a few assumptions

Again, we are collapsing our solution space to make finding an optimal sequence easier.

1. Assume the backbone geometry is fixed when new sequences are introduced
2. Assume each amino acid can only take on a finite number of geometries (rotamers)
3. Assume that what we want to do is to maximize the energy drop from the completely unfolded state to the target geometry
   - In other words, simply ignore all the other possible folded structures that we want to avoid

We might define rotamers based on existing knowledge of similar protein structure
The simplified problem

• At each position on the backbone, choose a rotamer (an amino acid type and a side-chain geometry) to minimize overall energy
  – We assume the energy is a free energy. The Rosetta all-atom force field (physics-based/knowledge-based hybrid) is a common choice.
  – For each amino acid sequence, energy is measured relative to the unfolded state.
    • In practice a “reference energy” for each amino acid is subtracted off, corresponding roughly to how much that amino acid favors folded states
    • You’re not responsible for this
  – Assume that energy can be expressed as a sum of terms that depend on one rotamer or two rotamers each. This is the case for the Rosetta force fields (and for most molecular mechanics force fields as well).
• Thus, we wish to minimize total energy \( E_T \), where

\[
E_T = \sum_i \left[ E_i(r_i) + \sum_{i \neq j} E_{ij}(r_i, r_j) \right]
\]

Note that \( r_i \) specifies both the amino acid at position \( i \) and its side-chain geometry

Often use monte-carlo methods to select residue rotamers, after many iterations, derive a minimal energy conformation
Optimization methods

• Heuristic methods
  – Not guaranteed to find optimal solution, but faster
  – Most common is Metropolis Monte Carlo
    • Moves may be as simple as randomly choosing a position, then randomly choosing a new rotamer at that position
    • May decrease temperature over time (simulated annealing)

• Exact methods
  – Guaranteed to find optimal solution, but slow for larger proteins
  – Most common is likely Dead-End Elimination Method, which prunes branches of the exhaustive search tree by proving that certain rotamers are incompatible with the global optimum
  – The A* optimization algorithm (originally developed at Stanford, for robot path-finding) is also used
  – You’re not responsible for the details of how these exact methods work.
First evidence that automated computational protein (sequence) design is possible

Starting with the backbone of a zinc finger domain, which normally binds zinc (often found in proteins that bind DNA), scientists showed that they could derive the backbone residue conformations that would allow the protein to bind zinc.

A zinc-finger structure without the zinc ion (a zinc finger is a structural motif often found in proteins that interact with DNA)

But notice how the backbone is shifted

Backbone is shifted, because certain rotamer changes break the static backbone assumption

This was done with an exact optimization method
A simplifying assumption was that we simply minimize the energy of the desired structure—
in other words, we explore a local minima, but there is no guarantee that our local minima is the global minimum as well.

We do not consider all other possible structures. It’s possible that their energy ends up even lower.

In negative design, we identify a few structures that we want to avoid, and we try to keep their energies high during the design process.

This can help, but we cannot explicitly avoid all possible incorrect structures without making the problem much more complicated. So the overall approach is still heuristic.
Fixed backbone design in action
Outline for today: Understanding the design work flow

Understand the structural context (for what purpose)
Starting from a known structure, or a topology if *de novo*

- Build a backbone
- Build the side chains
- Refine

Predict the structure (if small enough)

wiggle the atoms to create more sequence diversity and potentially explore better local structures
Refinement protocols are very common among structural modeling programs. In the design process, the initial guess is almost never perfect, and this creates an opportunity for the backbones to drift in a refinement step (gradient energy minimization). The sequence is then redesigned to fit the new structure, and the steps are repeated. The sequence may or may not converge but should explore lower energy structures.

Alternate between fixing backbone and updating rotamers. We can save on computational demand on each iteration.
You can actually use Foldit (standalone version) for these three central steps

1. Build a backbone
2. Build the side chains
3. Refine
Outline for today: Understanding the design work flow

If the sequence is small (~ <150 amino acids), you can just run it through the structure prediction pipeline (sequence -> structure, remember that structure computation can take a long time, even for smaller proteins)

Understand the structural context (for what purpose)
Starting from a known structure, or a topology if *de novo*

The problem at the end of a design cycle is usually that you have too many “good looking” sequences. Need to know how to pick solutions to test experimentally

Build a backbone

Build the side chains

Refine

Predict the structure (if small enough)
When the protein is small enough (<150 amino acid, in general), the resulting sequence can be run through the ab initio prediction pipeline to see if the prediction converge to the designed structure. A few thousand sequences can be screened this way.

Even though the algorithm is reinforcing it’s own solutions energetically. Think of this step as a re-sampling process. The sequence should yield the right fragment types, and the sampling should confirm that the amino acids can lock into their target location. If the structures came out wrong, the models can inform negative designs to correct the sequences.

“Forward Folding” — predict the designed sequences

Analyzing sequences with poor funnel plots can inform negative design, informing which sequences/motifs to avoid

Screen candidates based on if the funnel plot provides a distribution that results in the target structure (shown in the bottom left by green dots)