## Course Outline

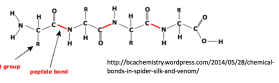
**Atomic-level modeling of proteins / macromolecules**
Protein structure; Energy functions + molecular conformation; MD simulation; protein structure prediction; protein design; ligand docking

**Coarser-level modeling and imaging-based methods**
Fourier transforms and convolution; Image analysis, X-ray crystallography; Single-particle electron microscopy, Microscopy, Diffusion and cellular-level simulation, Genome structure

### Atomic-level modeling

**Protein structure** Proteins are constantly jiggling around; as are molecules surrounding them (mostly water). There's also a cell membrane (made of lipids). Each protein can be made structures; but they tend to be similar. Usually we talk about the average structure (typically what is measured by X-ray crystallography in PDB). Surrounding molecules play a key role in determining structure.

**2D Protein structure.** Proteins are chains of amino acids. Amino acids are identical except for their sidechains. Proteins have reguar backbones with differing side chains.



http://biochemistry.wordpress.com/2014/05/28/chemical-bonds-in-spider-silk-and-venom/

**3D Protein structure - basic interactions** Bond length stretching, bond angle bending, torsional angle twisting, electrostatic interaction

**3D Protein structure - complex interactions** Hydrogen bonds, hydrophobic effet, particularly important.

### Energy functions + Molecular conformation

**Potential energy functions** Potential energy function $U(x)$ specifices total potential energy of a system of atoms; as a function of all their positions $\mathbf{x}$. For a system with $n$ atoms, $x$ is a vector of length $3n$ ($x_1$, $y_1$ and $z_1$ coordinates for every atom). More generally, include not only atoms in a pteotin but also surrounding atoms.

**Force calculation.** Take derivatives of potential energy to obtain force.

**Example energy function.**

$$U = \sum_{\text{bonds}} k_b \left(b - b_0\right)^2 + \sum_{\text{angles}} k_\theta \left(\theta - \theta_0\right)^2 + \sum_{\text{torsions}} \sum_n k_{\phi,n} \left[1 + \cos\left(n\phi - \phi_n\right)\right]$$

Bond lengths ("Stretch") / Bond angles ("Bend") / Torsional/dihedral angles — Bonded terms

$$+ \sum_i \sum_{j>i} \frac{q_i q_j}{r_{ij}} + \sum_i \sum_{j>i} \frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}}$$

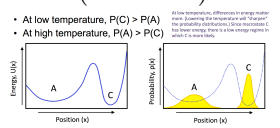
Electrostatic / Van der Waals — Non-bonded terms

**Boltzmann distribution.** Relates potential energy of an arrangemet of atoms to the probability of observing that arrangement of atoms (At equilibrium).

$$p(x) \propto \exp\left(-U(x)/(k_b T)\right)$$

- At low temperature, P(C) > P(A)
- At high temperature, P(A) > P(C)

At low temperatures, differences in energy matter more. (Lowering the temperature will "sharpen" the probability distribution.) Since movement is low, we know that there is a low-energy region in which C is more likely.




**Macrostates.** We typically care most about probability that protein atoms will be in some approximate arrangement, with any arrangement of surrounding atoms. Macrostates are "wells" of the potential energy function. To calculate the probability of a well, sum the probabilities of all specific atomics arrangements it contains.

$$P(A) = \int_{x \in A} P(\boldsymbol{x}) \propto \int_{x \in A} \exp\left(\frac{-U(\boldsymbol{x})}{k_B T}\right) d\boldsymbol{x}$$

**Free energies.** The free energy $G_A$ of a macrostate $A$ satisfies $P(A) = \exp(-G_A/(k_B T))$. Free energies are clearly defined only for macrostates. However, in protein structure prediction; design, and docking; often useful to define a "free energy function" that approximates a free energy function for some neighborhood of each arrangement of protein atoms. To predict protein structure - minimize free energy, not potential energy. The term energy function is used for both potential energy and free energy functions.

### Molecular dynamics

**MD overview.** An MD simulation predicts how atoms move around based on physical models of their interactions. Closest to the physics; aims to predict real dynamics of system. Can capture structural changes in proteins; protein-ligand binding, or protein folding.

**Basic MD algorithm.** Step through time (very short steps). At each step, calculate force acting on every atom using a molecular mechanics force field. Then update atom positions and velocities using Newton's second law. $\frac{dx}{dt} = v$; $\frac{dv}{dt} = \frac{F(x)}{m}$.
Note - this is an approximation, because we are using classical physics rather than quantum mechanics. But quantum mechanical calculations can be used to parametrize force fields.

**Why MD is computationally intensive.** One needs to take millions to trillions of timesteps to get to timescales on which events of interest take place. Computing the forces at each time step involves substantial computation. Particularly for non-bonded force terms! Which at between every pair of atoms.

**Sampling.** Given enough time, an MD simulation will sample the full Boltzmann distribution of the system. Hence - if one took a snapshot from the simulation after a long period of time; then the probability of the atoms being in a particular arrangement is given by the Boltzmann distribution.
One can also sample the Boltzmann distribution in other ways, including Monte Carlo sampling with the Metropolis criterion – Metropolis Monte Carlo: generate moves at random. Accept any move that decreases the energy. Accept moves that increase the energy by ΔU with probability exp$(-\Delta U/(k_B T))$. – If one decreases the temperature over time, this becomes a minimization method (simulated annealing)

### Protein structure prediction.

**Overview.** The goal: given the amino acid sequence of a protein, predict its average three-dimensional structure • In theory, one could do this by MD simulation, but that isn't practical • Practical methods for protein structure prediction take advantage of existing data on protein structure (and sequence)

**Two approaches.** Template-based modeling (homology modeling) – Used when one can identify one or more likely homologs of known structure (usually the case) • Ab initio structure prediction – Used when one cannot identify any likely homolog of known structure – Even ab initio approaches usually take advantage of available structural data, but in more subtle ways

**Template based modeling. Workflow.** • User provides a query sequence with unknown structure • Search the PDB for proteins with similar sequence and known structure. Pick the best match (the template). • Build a model based on that template – One can also build a model based on multiple templates, where different templates are used for different parts of the protein.

**Structure is more conserved than sequence.** Proteins with similar sequences tend to be homologs, meaning that they evolved from a common ancestor • The fold of the protein (i.e., its overall structure) tends to be conserved during evolution • This tendency is very strong. Even proteins with 15% sequence identity usually have similar structures. – During evolution, sequence changes more quickly than structure

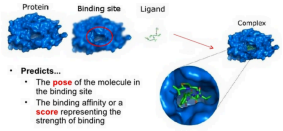
**Ab initio modeling - typically Rosetta.** Search for structure that minimizes an energy function – This energy function is knowledge-based (informed, in particular, by statistics of the PDB), and it approximates a free energy function • Use a knowledge-based search strategy – Rosetta uses a Monte Carlo search method involving "fragment assembly," in which it tries replacing structures of small fragments of the protein with fragment structures found in the PDB

**Protein design.** • Goal: given a desired approximate threedimensional structure (or structural characteristics), find an amino acid sequence that will fold to that structure • In principle, we could do this by searching over sequences and doing ab initio structure prediction for each possible sequence, but that's not practical

**Simplifying the problem.** • Instead of predicting the structure for each sequence considered, just focus on the desired structure, and find the sequence that minimizes its energy – Energy is generally measured by a knowledge-based free energy function • Consider a discrete set of rotamers for each amino acid side chain – Minimize simultaneously over identities and rotamers of amino acids • Assume the backbone is fixed – Or give it a bit of "wiggle room"

**Heuristic but effective.** These simplifications mean that practical protein design methodologies are highly heuristic, but they've proven surprisingly effective • The minimization problem itself is also usually solved with heuristic methods (e.g., Metropolis Monte Carlo)

**Ligand Docking.**
Goals: – Given a ligand known to bind a particular protein, determine its binding pose (i.e., location, orientation, and internal conformation of the bound ligand) – Determine how tightly a ligand binds a given protein




**Binding affinity** • Binding affinity quantifies the binding strength of a ligand to a protein (or other target) • Conceptual definition: if we mix the protein and the ligand (with no other ligands around), what fraction of the time will the protein have a ligand bound? • Affinity can be expressed as the difference $\Delta G$ in free energy of the bound state and the unbound state, or as the concentration of unbound ligand molecules at which half the protein molecules will have a ligand bound

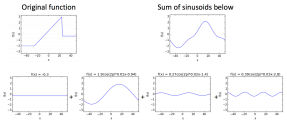
**Docking is heuristic.** In principle, we could estimate binding affinity by measuring the fraction of time the ligand is bound in an MD simulation, but this isn't practical

**Docking methodology.** Ligand docking is a fast, heuristic approach with two key components – A scoring function that very roughly approximates the binding affinity of a ligand to a protein given a binding pose – A search method that searches for the best-scoring binding pose for a given ligand • Most ligand docking methods assume that – The protein is rigid – The approximate binding site is known • That is, one is looking for ligands that will bind to a particular site on the target • In reality, ligand mobility, protein mobility, and water molecules all play a major role in determining binding affinity – Docking is approximate but useful – The term scoring function is used instead of energy function to emphasize the highly approximate nature of the scoring function

**Does this apply to RNA?** Some of these concepts apply with little or no modification to RNA and other biomolecules – Energy functions and their relationship to molecular conformation – Molecular dynamics simulation – Ligand docking • In other cases, the basic ideas apply, but the techniques are different – (RNA) structure prediction – (RNA) design

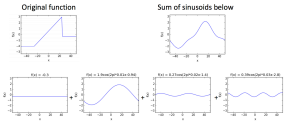
### Coarser modeling + imaging

**Fourier Transform.** Writing functions as sums of sinusoids




Given a function defined on an interval of length L, we can write it as a sum of sinusoids with the following frequencies/periods: – Frequencies: 0, 1/L, 2/L, 3/L, …. – Periods: constant term, L, L/2, L/3, …
Each of these sinusoidal terms has a magnitude (scale factor) and a phase (shift).
We can thus express the original function as a series of magnitude and phase coefficients – We can express each pair of magnitude and phase coefficients as a complex number • The Fourier transform maps the function to this set of complex numbers, providing an alternative representation of the function. • This also works for functions of 2 or 3 variables (e.g., images) • Fourier transforms can be computed efficiently using the Fast Fourier Transform (FFT) algorithm

**Convolution.** A weighted moving average. To convolve one function with another, we computed a weighted moving average of one function using the other function to specify the weights




Convolving two functions is equivalent to multiplying them in the frequency domain • In other words, we can perform a convolution by taking the Fourier transform of both functions, multiplying the results, and then performing an inverse Fourier transform • Why is this important? – It provides an efficient way to perform large convolutions (thanks to the FFT) – It allows us to interpret convolutions in terms of what they do to different frequency components (e.g., high-pass and lowpass filters)

**Image analysis.** Representations of an image: • We can think of a grayscale image as: – A two-dimensional array of brightness values – A function of two variables (x and y), which returns the brightness of the pixel at position (x, y) • A color image can be treated as: – Three separate images, one for each color channel (red, green, blue) – A function that returns three values (red, green, blue) for each (x, y) pair

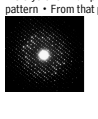
**Reducing image noise.** • We can reduce image noise using various filters (e.g., mean, median, Gaussian) – These all rely on the fact that nearby pixels in an image tend to be similar • The mean and Gaussian filters (and many others) are convolutions, and can thus be expressed as multiplications in the frequency domain – These denoising filters are low-pass filters. They reduce highfrequency components while preserving low-frequency components – These filters work because real images have mostly lowfrequency content, while noise tends to have a lot of highfrequency content

**High pass filters.** A high-pass filter reduces low-frequency components while preserving high-frequency components • We can sharpen images by adding a high-pass filtered version of the image to the original image • High-pass filtering can also be used to remove undesired background brightness that varies smoothly across the image

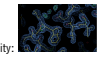
**Principcal Component Analysis** Basic idea: given a set of points in a multi-dimensional space, we wish to find the linear subspace (line, plane, etc.) that best fits those points.
• PCA provides a way to represent high-dimensional data sets (such as images) approximately in just a few dimensions • It is thus useful in summarization and classification of images

### X-ray crystallography

**Overview.** Get the molecule whose structure you want to determine to form a crystal – Can be very hard, especially for large structures and membrane proteins • Shine an intense beam of x-rays through the crystal, giving rise to a "diffraction pattern" (a pattern of spots of varying brightnesses) – Shine x-rays through the crystal at multiple angles to capture the full 3D diffraction pattern • From that pattern, infer the 3D structure of the molecule



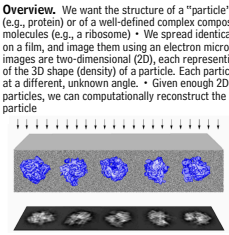
**Diffraction -> Structure.** The diffraction pattern is the Fourier transform of the electron density! – But only the magnitude of each Fourier coefficient is measured, not the phase • The lack of phase information makes solving the structure (i.e., going from the diffraction pattern to a set of 3D atomic coordinates) challenging


Contour map of electron density:

**Solving for molecular structure.** Step 1: Initial phasing – Come up with an approximate solution for the structure (and thus an approximate set of phases), often using a homologous protein as a model • Step 2: Phase refinement – Search for perturbations that improve the fit to the experimental data (the diffraction pattern), often using simulated annealing – Restrain the search to "realistic" molecular structures, usually using a molecular mechanics force field

### Single particle EM.

**Overview.** We want the structure of a "particle": a molecule (e.g., protein) or of a well-defined complex composed of many molecules (e.g., a ribosome) • We spread identical particles out on a film, and image them using an electron microscope • The images are two-dimensional (2D), each representing a projection of the 3D shape (density) of a particle. Each particle is positioned at a different, unknown angle. • Given enough 2D images of particles, we can computationally reconstruct the 3D shape of a particle




**2D -> 3D structure.** 2D image analysis: First, go from raw image data to high-resolution 2D projections – Image preprocessing – Particle picking – Image clustering and class averaging: reduce image noise by identifying images with similar view angles, aligning them, and averaging them

**3D reconstruction.** 3D reconstruction: Then use these high-resolution projections to build a 3D model – Reconstruction with known view angles is fairly straightforward. Standard algorithm is filtered back-projection. – Structure refinement with unknown view angles (the problem at hand) is harder. Iterate between improving estimates of view angles given a 3D model and building a better 3D model given those view angles. – Calculating an initial structure – Optional final step: Fitting atomic-resolution models to lower-resolution EM structures

### Microscopy

**Fluorescence Microscopy** • Suppose we want to know where a particular type of protein is located in the cell, or how these proteins move around • We can't do this by simply looking through a microscope, because: – We (usually) don't have sufficient resolution – The protein of interest doesn't look different from those around it • Solution: Make the molecules of interest glow by attachment of fluorophores (fluorescent molecules) – When you shine light of a particular wavelength on a fluorophore, it emits light of a different wavelength

**Single molecule tracking.** If the density of fluorescent molecules is sufficiently low, we can track individual molecules – Doing this well is a challenging computational problem

**Diffraction limit.** The image observed under a microscope is always slightly blurred due to fundamental limitations on how well a lens can focus light – The observed image is a low-pass filtered version of the ideal image • This leads to a limit on resolution known as the diffraction limit – The achievable resolution scales with wavelength of the radiation used (i.e., a shorter wavelength leads to a smaller minimum distance between resolvable points) – X-rays have shorter wavelength than visible light. Electrons have much shorter wavelengths.

### Diffusion / simulation.

How do molecules move in a cell? -Molecules jiggle about because other molecules keep bumping into them • Individual molecules thus follow a random walk • Diffusion = many random walks by many molecules – Substance goes from region of high concentration to region of lower concentration – Aggregate behavior is deterministic

**Particle perspective.** In the basic case of random, unconfined, undirected motion: – Individual molecules follow a random walk, leading to Brownian motion – Mean squared displacement is proportional to time – The proportionality constant is specified by the diffusion constant • Faster-moving molecules have larger diffusion constants

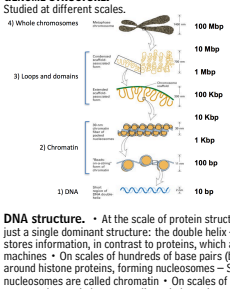
**Continuum view of diffusion.** If enough molecules are involved, we can predict their aggregate behavior.
• The rate at which the concentration of a molecule changes with time is given by the diffusion equation. – This rate is determined by the second derivatives of concentration with respect to each spatial coordinate

$$\frac{\partial c}{\partial t} = D\left(\frac{\partial^2 c}{\partial x^2} + \frac{\partial^2 c}{\partial y^2} + \frac{\partial^2 c}{\partial z^2}\right).$$

**Reaction-diffusion simulation.** • A common way to model how molecules move within the cell involves reaction-diffusion simulation • Basic rules: – Molecular move around by diffusion – When two molecules come close together, they have some probability of reacting to combine or modify one another • Two implementation strategies: – Particle-based models – Continuum models (based on the diffusion equation)

### Genome structure.

Studied at different scales.




**DNA structure.** • At the scale of protein structure, DNA forms just a single dominant structure: the double helix – DNA primarily stores information, in contrast to proteins, which act as molecular machines • On scales of hundreds of base pairs (bp), DNA wraps around histone proteins, forming nucleosomes – Strings of nucleosomes are called chromatin • On scales of 10 Kbp or greater, chromatin has a complicated, dynamic structure – This has a big effect on gene expression – It's a focus of current research

**Contact maps.** Chromosome conformation capture techniques identify frequencies of contacts between one part of a DNA strand and another. These contact frequencies can be used to infer structural features of chromatin, such as domains and loops.

**Simulating chromatin's spatial organization**
Simulations are used: – To reconstruct 3D genome structure, beginning with a contact map. Here, the contact map is used as a set of constraints. – To test mechanistic hypotheses (e.g., for how loops and domains are formed). One generally compares the results to experimental contact maps. • Multiple scales – The simulations above are coarse-grained, with one particle representing 1-500 Kbp (closer to 1 Kbp for simulation of loops/domains, closer to 500 Kbp for simulation of whole chromosomes) – Simulations are also used to study DNA structure on finer scales. These might be all-atom MD simulations (one particle represents one atom), or they might be somewhat coarser grained (e.g., one particle represents one base).

### Recurring themes

**Physics vs. data driven approaches** • Physics-based approaches: modeling based on first-principles physics • Data-driven approaches: inference/learning based on experimental data • Examples: – Physics-based vs. knowledge-based energy functions – Molecular dynamics vs. protein structure prediction • Most methods fall somewhere on the continuum between these two extremes – Examples: ligand docking or solving x-ray crystal structures

**Energy functions** • Energy functions (representing either potential energy or free energy) play a key role in many of the techniques we've covered, including: – Molecular dynamics – Protein structure prediction – Protein design – Ligand docking – X-ray crystallography

**Structural interpretation + predictions.** • Structural predictions – Protein structure prediction – Molecular dynamics simulation – Ligand docking – Reaction-diffusion simulations • Structural interpretation of experimental data – X-ray crystallography – Single-particle electron microscopy – Image analysis for fluorescence microscopy data – Analyzing chromosome conformation capture data

**Math ideas.** • Fourier transforms and convolution play important roles in: – Image analysis – X-ray crystallography – Single-particle electron microscopy – (And also in molecular dynamics and some docking methods, though you're not responsible for that) • Another recurring math concept: Monte Carlo methods

**Methods at different spatial scales** • Atomic-level modeling of single proteins vs. coarser-level modeling of complexes and cells • Experimental methods: x-ray crystallography vs. single-particle electron microscopy • Simulation methods: molecular dynamics vs. reaction-diffusion simulations

### End of review: Supplemental

**Details on FFT.**

Let $x_0, \ldots, x_{N-1}$ be complex numbers. The DFT is defined by the formula

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i 2\pi k n / N} \qquad k = 0, \ldots, N-1.$$

Evaluating this definition directly requires $O(N^2)$ operations: there are $N$ outputs $X_k$, and each output requires a sum of $N$ terms. An FFT is any method to compute the same results in $O(N \log N)$ operations. All known FFT algorithms require $\Omega(N \log N)$ operations, although there is no known proof that a lower complexity score is impossible.[17]

**Convolution.** $[f * g](n) = \sum_{m=-\infty}^{\infty} f(m) g(n - m)$.

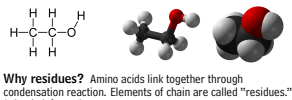
**Convolution theorem.**

$$f * g = \mathcal{F}^{-1}\{\mathcal{F}\{f\} \cdot \mathcal{F}\{g\}\}$$

Allows us to perform convolution faster. Allows allows us to characterize convolution operations in the frequency domains. e.g. convolution with a Gaussian will preserve low-frequency components while reducing high-frequency components.

**PDB.** A collection of (almost) all published experimental structures of biomacromolecules (e.g., proteins) • Each identified by 4-character code (e.g., 1rcx) • Currently 134,000 structures. 90determined by x-ray crystallography. • Browse it and look at some structures. Options: • 3D view in applet on PDB web pages • PyMol: fetch 1rcx • VMD: mol pdbload 1rcx

**2D vs 3D structure.** Two-dimensional (chemical) structure vs. three-dimensional structure • Two-dimensional (chemical) structure shows covalent bonds between atoms. Essentially a graph. • Three-dimensional structure shows relative positions of atoms.




**Why residues?** Amino acids link together through condensation reaction. Elements of chain are called "residues." (what is left over).

**Hydrophobic effect.** Hydrophilic molecules are polar and thus form
hydrogen bonds with water • Polar = contains charged atoms. Molecules containing oxygen or nitrogen are usually polar. • Hydrophobic molecules are apolar and don't form hydrogen bonds with water

**Hydrophobicity** Water molecules around a non-polar solute form a cage-like structure, which reduces the entropy. When two non-polar groups associate, water molecules are liberated from the solvation shell, increasing teh entropy.

**Free energy** So far we have assigned energies only to microstates, but it's useful to assign them to macrostates as well. • Define the free energy GA of a macrostate A such that:

$$P(A) = \exp\left(-G_A/(k_B T)\right)$$ • This is analogous to Boltzmann distribution formula: $P(X) \propto \exp\left(-U(x)/(k_B T)\right)$.

Can solve for $G_A$ to get $G_A = -k_B T \log_e(P(A))$. Can also express free energy in terms of enthalpy (mean potential energy, H) and entropy (S): $G_A = H_A - TS_A$.
Protein will adopt conformational state with the minimum free energy! Note that at room temp; conformational state of minimum free energy is usually very different from the microstate with minimum potential energy.

**Potential energy vs. free energy.** Potential energy describes how often individual microstates will be occupied. Free energy describes how often macrostates, or groups of these microstates, will be occupied.
In particular, free energy takes into account both the potential energy of the individual microstates (roughly enthalpy) and the number of microstates (entropy) by integrating the probability of being in each individual microstate. Please review assignment 2 question 4 to see precisely how free energy is calculated from potential energy.

**X-ray crystallography vs. EM** Keep in mind that x-rays are photons: self-propagating packets of light energy. Electrons are charged particles: movement results because of the Lorentz force a.k.a. an electric field or magnetic field acting upon the particle. The wavelength of x-rays is given by the dispersion relation: $\lambda = c/f$ while the wavelength of electrons is given by the de Broglie wavelength: $\lambda = h/(mv)$.
With that out of the way, Abbe's diffraction limit $d = \lambda / (2 \mathrm{NA} \sin(\theta))$ is a result of imaging not crystallography. Imaging requires that you have the phase (and lenses do that for us). Crystallography does not recover the phase. In this way, you could say crystallography throws away the phase information in the hope of getting higher resolution on just the magnitude part of the Fourier Transform.
There are also some practical engineering considerations w.r.t. the combinations of x-rays/electrons and imaging/crystallography. X-ray and electron lenses are hard to make. Electron TEM/SEM imagers use a scanning point-by-point approach. I also speculate that electrons are hard to bombard through a crystal without damaging it, electron imaging is more used for thin sections not relatively thick crystals.

**Simulated annealing** Lower the temperature gradually so as to select for shallower, narrower wells.

**Force / energy** $F(x) = -\nabla U(x)$.

**Simulating solvent / water** Ignoring the solvent (the molecules surrounding the molecule of interest) leads to major artifacts – Water, salt ions (e.g., sodium, chloride), lipids of the cell membrane • Two options for taking solvent into account – Explicitly represent solvent molecules • High computational expense but more accurate • Usually assume periodic boundary conditions (a water molecule that goes off the left side of the simulation box will come back in the right side, like in PacMan) – Implicit solvent • Mathematical model to approximate average effects of solvent • Less accurate but faster

**MD timescales** Simulations require short time steps for numerical stability – 1 time step ≈ 2 fs (2×10–15 s) • Structural changes in proteins can take nanoseconds (10–9 s), microseconds (10–6 s), milliseconds (10–3 s), or longer – Millions to trillions of sequential time steps for nanosecond to millisecond events (and even more for slower ones) • Until recently, simulations of 1 microsecond were rare • Advances in computer power have enabled microsecond simulations, but simulation timescales remain a challenge • Enabling longer-timescale simulations is an active research area, involving: – Algorithmic improvements – Parallel computing – Hardware: GPUs, specialized hardware

**Limitation of MD: Covalent bonds** • Once a protein is created, most of its covalent bonds do not break or form during typical function. • A few covalent bonds do form and break more frequently (in real life): – Disulfide bonds between cysteines – Acidic or basic amino acid residues can lose or gain a hydrogen (i.e., a proton)

**Why is MD so intensive?** • Many time steps (millions to trillions) • Substantial amount of computation at every time step – Dominated by non-bonded interactions, as these act between every pair of atoms. • In a system of N atoms, the number of non-bonded terms is proportional to N2 – Can we ignore interactions beyond atoms separated by more than some fixed cutoff distance? • For van der Waals interactions, yes. These forces fall off quickly with distance. • For electrostatics, no. These forces fall off slowly with distance.

**How to speed up MD?** • Reduce the amount of computation per time step • Reduce the number of time steps required to simulate a certain amount of physical time • Reduce the amount of physical time that must be simulated • Parallelize the simulation across multiple computers • Redesign computer chips to make this computation run faster

**Monte Carlo / Metropolis Criterion** Randomly sample over move space but accept a move using Metropolis Criterion. Ensures that simulation samples the Boltzmann distribution.

- The Metropolis criterion for accepting a move is:
  - Compute the potential energy difference ($\Delta U$) between the pre-move and post-move position
    - $\Delta U < 0$ if the move would *decrease* the energy
  - If $\Delta U \leq 0$, accept the move
  - If $\Delta U > 0$, accept the move with probability $e^{-\Delta U / k_B T}$

If run for long enough, probability of observing a particular arrangement of atoms is given by Boltzmann distribution $p(x) \propto \exp\left(-U(x)/(k_B T)\right)$.

Gradually reduce temperature $T$ during simulation - becomes a minimization strategy.

**How are predicted structures used?** • Drug development – Computational screening of candidate drug compounds – Figuring out how to optimize a promising candidate compound – Figuring out which binding site to target • Predicting the function of a protein • Identifying the mechanism by which a protein functions, and how one might alter that protein's function (e.g., with a drug) • Interpreting experimental data – For example, a computationally predicted approximate structure can help in determining an accurate structure experimentally, as we'll see later in this course

**Why not solve structures experimentally?** • Some structures are very difficult to solve experimentally – Sometimes many labs work for decades to solve the structure of one protein • Sequence determination far outpaces experimental structure determination – We already have far more sequences than experimental structures, and this gap will likely grow

**Why not simulate protein folding using MD?** 1. Folding timescales are usually much longer than simulation timescales. 2. Current molecular mechanics force fields aren't sufficiently accurate. 3. Disulfide bonds form during the real folding process, but this is hard to mimic in simulation.

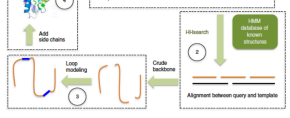
**What if we can't identify homolog in PDB?** We can still use information based on known structures – We can construct databases of observed structures of small fragments of a protein – We can use the PDB to build empirical, "knowledgebased" energy functions

**Basic sequence similarity measures** • Basic measure: count minimum number of amino acid residues one needs to change, add, or delete to get from one sequence to another – Sequence identity: amino acids that match exactly between the two sequences – Not trivial to compute for long sequences, but there are efficient dynamic programming algorithms to do so

**Better sequence similarity measures** • We can do better – Some amino acids are chemically similar to one another (example: glutamic acid and aspartic acid) • Sequence similarity is like sequence identity, but does not count changes between similar amino acids

**Even better sequence similarity measures** We can do even better – Once we've identified some homologs to a query sequence (i.e., similar sequences in the sequence database), we can create a profile describing the probability of mutation to each amino acid at each position – We can then use this profile to search for more homologs – Iterate between identification of homologs and profile construction – Measure similarity of two sequences by comparing their profiles – Often implemented using hidden Markov models (HMMs) (but you are not responsible for knowing about HMMs)

**Phyre2**




- Identify similar sequences in protein sequence database
- Choose a template structure by: (1) comparing sequence profiles and (2) predicting secondary structure for each residue in the query sequence and comparing to candidate template structures. Secondary structure (alpha helix, beta sheet, or neither) is predicted for segments of query sequence using a neural network trained on known structures.
- Compute optimal alignment of query sequence to template structure
- Build a crude backbone model (no side chains) by simply superimposing corresponding amino acids. Some of the query residues will not be modeled, because they don't have corresponding residues in the template (insertions). There will be some physical gaps in the modeled backbone, because some template residues don't have corresponding query residues (deletions).
- Use loop modeling to patch up defects in the crude model due to insertions and deletions. For each insertion or deletion, search a large library of fragments (2-15 residues) of PDB structures for ones that match local sequence and fit the geometry best. Tweak backbone dihedrals within these fragments to make them fit better
- Add side chains. Use a database of commonly observed structures for each side chain (these structures are called rotamers). Search for combinations of rotamers that will avoid steric clashes (i.e., atoms ending up on top of one another).
- • In "intensive mode," Phyre 2 will use multiple templates that cover (i.e., match well to) different parts of the query sequence. – Build a crude backbone model for each template – Extract distances between residues for "reliable" parts of each model – Perform a simplified protein folding simulation in which these distances are used as constraints. Additional constraints enforce predicted secondary structure – Fill in the side chains, as for single-template models

**Ab initio structure prediction** Recall two main approaches for protein structure prediction: Two main approaches to protein structure prediction • Template-based modeling (homology modeling) – Used when one can identify one or more likely homologs of known structure • Ab initio structure prediction – Used when one cannot identify any likely homologs of known structure – Even ab initio approaches usually take advantage of available structural data, but in more subtle ways

**Rosetta** Knowledge-based energy function – In fact, two of them: • The "Rosetta energy function," which is coarse-grained (i.e., does not represent all atoms in the protein), is used in early stages of protein structure prediction • The "Rosetta all-atom energy function," which depends on the position of every atom, is used in late stages • A knowledge-based strategy for searching conformational space (i.e., the space of possible structures for a protein) – Fragment assembly forms the core of this method

**Rosetta energy function** At first this was the only energy function used by Rosetta (hence the name) • Based on a simplified representation of protein structure: – Do not explicitly represent solvent (e.g., water) – Assume all bond lengths and bond angles are fixed – Represent the protein backbone using torsion angles (three per amino acid: $\phi$, $\psi$, $\omega$) – Represent side chain position using a single "centroid," located at the side chain's center of mass • Centroid position determined by averaging over all structures of that side chain in the PDB

**Take away from Rosetta energy functions** The (coarse-grained) Rosetta energy function is essentially entirely knowledge-based – Based on statistics computed from the PDB • Many of the terms are of the form –log[ P(A) ], where P(A) is the probability of some event A – This is essentially the free energy of event A. Recall definition of free energy:

$$G_A = -k_B T \log_e(P(A)) \qquad P(A) = \exp\left(-G_A / k_B T\right)$$

**Simplifying assumptions of Rosetta** • Still makes simplifying assumptions: – Do not explicitly represent solvent (e.g., water) – Assume all bond lengths and bond angles are fixed • Functional forms are a hybrid between molecular mechanics force fields and the (coarse-grained) Rosetta energy function – Partly physics-based, partly knowledge-based

**PE functions or FE functions?** • The energy functions of previous lectures were potential energy functions • One can also attempt to construct a free energy function, where the energy associated with a conformation is the free energy of the set of "similar" conformations (for some definition of "similar") • The Rosetta energy functions are sometimes described as potential energy functions, but they are closer to approximate free energy functions – This means that searching for the "minimum" energy is more valid – Nevertheless, typical protocol is to repeat the search process many times, cluster the results, and report the largest cluster as the solution. This rewards wider and deeper wells.

**Rosetta fragment assembly** • Uses a large database of 3-residue and 9-residue fragments, taken from structures in the PDB • Monte Carlo sampling algorithm proceeds as follows: 1. Start with the protein in an extended conformation 2. Randomly select a 3-residue or 9-residue section 3. Find a fragment in the library whose sequence resembles it 4. Consider a move in which the backbone dihedrals of the selected section are replaced by those of the fragment. Calculate the effect on the entire protein structure. 5. Evaluate the Rosetta energy function before and after the move. 6. Use the Metropolis criterion to accept or reject the move. 7. Return to step 2 • The real search algorithm adds some bells and whistles

**Rosetta refinement** • Refinement is performed using the Rosetta allatom energy function, after building in side chains • Refinement involves a combination of Monte Carlo moves and energy minimization • The Monte Carlo moves are designed to perturb the structure much more gently than those used in the coarse search – Many still involve the use of fragments

**Best structure prediction method** • Currently, it's probably I-TASSER – http://zhanglab.ccmb.med.umich.edu/I-TASSER/ • I-TASSER is template-based, but it uses threading, meaning that when selecting a template it maps the query sequence onto the template structure and evaluates the quality of the fit – This allows detection of very remote homology • I-TASSER combines many algorithms – It incorporates a surprisingly large number of different components and strategies, including an ab initio prediction module – It runs many algorithms in parallel and then looks for a consensus between the results • Example: at least seven different threading algorithms – Inelegant but effective

**Structure prediction game** FoldIt, Eterna (RNA design), allow players to fold / with the goal of achieving minimum energy conformations.

**Comparing structures of a protein** Comparing structures of a protein • The most common measure of similarity between two structures for a given protein is root mean squared distance/deviation (RMSD), defined as

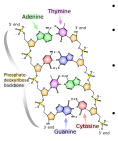
$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - w_i)^2}.$$

where x gives the coordinates for one structure and w the coordinates for the other • We generally want to align the structures, which can be done by finding the rigid-body rotation and translation of one structure that will minimize its RMSD from the other – The relevant measure of similarity is RMSD after alignment.

## Genome structure

**Epigenetics** • Epigenetics is the study of how the same genome produces different functions. – DNA can be chemically marked – DNA can be left accessible or packaged away – DNA can form loops that bring together important sites – RNA is also regulated in many different ways a Gene regulation is closely tied to genome organization.

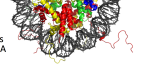
**DNA is stiff**




- DNA has repeating negative charge – it is a *polyelectrolyte*
- The negative charges repel, making DNA quite inflexible at the scale of 10s of base pairs
- Ions in solution affect DNA flexibility
  - Monovalent ions screen charge
  - Divalent cations (Mg2+) promote compaction
  - Important for simulation
- How do you compact DNA? By wrapping it around something with positive charges.

**Nucleosome**

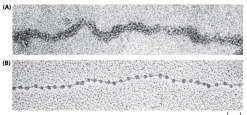
- Around 150bp of DNA winds around an octamer of eight *histone proteins*
- Together, the DNA and the histone octamer are called a *nucleosome*
- Histones have many positively charged residues
- This wrapping compacts the genome linearly by about 7x
- Stringing multiple nucleosomes together gives *chromatin* – DNA and its associated proteins.



**Chromatin fiber**

The chromatin fiber

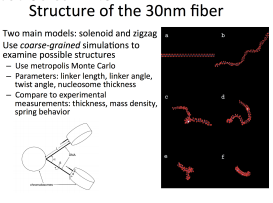
- Repeating nucleosomes form the 10nm fiber, like beads on a string
- In a test tube, chromatin condenses to form a thick fiber about 30nm in diameter




**Structure of 30nm fiber**

Structure of the 30nm fiber

- Two main models: solenoid and zigzag
- Use *coarse-grained* simulations to examine possible structures
  - Use metropolis Monte Carlo
  - Parameters: linker length, linker angle, twist angle, nucleosome thickness
  - Compare to experimental measurements: thickness, mass density, spring behavior



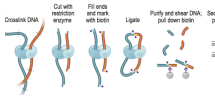
**Nucleosomes / gene regulation**

Nucleosomes affect gene expression in two important ways:

1. Presence of a nucleosome at the beginning of a gene inhibits its transcription
2. Chromatin can be *accessible* or *compacted*. Compaction inhibits transcription
   - This is determined by chemical modifications on histone tails

**Chromosome conformation capture**

- Under a microscope, all DNA sequences "look the same," so you can't match your image to genomic position
  - There are ways to mark specific DNA sequences (fluorescence in situ hybridization), but very low throughput
- We can use DNA sequencing to map self-contacts of the folded 3D genome
- This method is called *chromosome conformation capture* (3C)




**3D genome reconstruction**

Given a map of self-contacts, how can you reconstruct the structures that produced it?

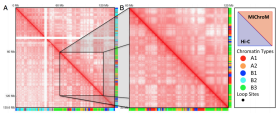
1. Knowledge-based: 3C or other data can be used as constraints
2. Mechanism-based: propose a mechanism that organizes the genome and simulate it

Mechanisms that organize the genome are still quite mysterious. Need computation to tackle this problem.

**Minimal Chromatin Model**

"**Minimal Chromatin Model**" (MiChroM)

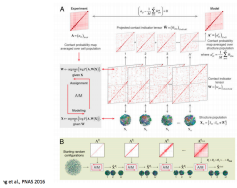
- Knowledge-based: use 1D compartment info
- Beads in the same compartment prefer to interact
- Train energy function on one chromosome and apply it to the whole genome



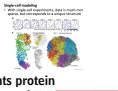

**Population modeling**

**Population-based modeling** – use contacts as constraints

- Assignment step: decide which contacts are most important for each individual structure
- Modeling step: optimize the structure based on those constraints

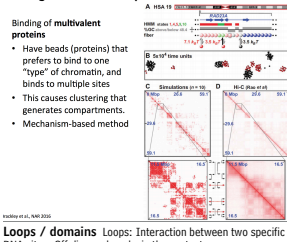



ig et al., PNAS 2016




**Single cell modeling**

**Binding of multivalents protein**




Binding of **multivalent proteins**

- Have beads (proteins) that prefers to bind to one "type" of chromatin, and binds to multiple sites
- This causes clustering that generates compartments.
- Mechanism-based method

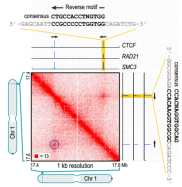
Hackley et al., NAR 2016

**Loops / domains** Loops: Interaction between two specific DNA sites. Off diagonal peaks in the contact map. Domain: continguous region of DNA that interacts much more with itself than with its neighbors. Appears as a square on the contact map diagonal.

Loop formation correlates with nearby gene activation.

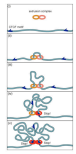
**Anchored at convergent CTCF sites**

Loops are anchored at convergent CTCF sites




**Looping via diffusion should not produce convergent sites.**





**Loops and domain form via extrusion**

**Extrusion simulations. Simulate extrusion.**

$$U_n = \sum_{i=m-1,n+1} K_{\text{bond}}(R_{\text{bond}} - d_{n,i})^2 \quad \text{Polymer backbone}$$

$$+ \sum_{i \neq n} 4\epsilon_{\text{LJ}}\left[\left(\frac{\sigma_{\text{LJ}}}{d_{n,i}}\right)^{12} - \left(\frac{\sigma_{\text{LJ}}}{d_{n,i}}\right)^{6}\right] \quad \text{Lennard-Jones forces}$$

$$+ \sum_{i \in L(n)} K_{\text{loop}}(R_{\text{loop}} - d_{n,i})^2 \quad \text{Loop bonds}$$

$$d_{n,i} = \|x_n - x_i\|$$

Coarse-grained simulations:
- DNA is a polymer of identical repeating units
- Local forces drive compaction
- Certain pairs of sites are bonded in a loop
- Extrusion causes the looped sites to move over time

## Protein design

**Overview** • Why design proteins? • Overall approach: Simplifying the protein design problem • Protein design methodology – Designing the backbone – Select sidechain rotamers: the core optimization problem – Optional: giving the backbone wiggle room – Optional: negative design – Optional: complementary experimental methods • Examples of recent designs • How well does protein design work?

**Applications of protein design** Sample applications • 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

**Direct approach to protein design** • Computationally intractable – We'd need to use ab initio structure prediction – Ab initio structure prediction for even one sequence is computationally intensive – 20N possible sequences with N residues • May not be good enough! – Ab initio structure prediction is far from perfect, in part because energy functions are imperfect – Given an energy function, what we really want is to maximize the probability of the desired structure (compared to all other possible folded and unfolded structures) – We could do this by sampling the full Boltzmann distribution for each candidate sequence … but that's even more computationally intensive!

**How to simplify protein design** We can dramatically simplify this problem by making a few assumptions 1. Assume the backbone geometry is fixed 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

**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).
Want to minimize total free energy:

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

**Designing backbone** • The first step of most protein design protocols is to select one or more target backbone structures. – This is as much art as science. – Often multiple target structures are selected, because some won't work. (Apparently proteins can only adopt a limited set of backbone structures, but we don't have a great description of what that set is.) • Methods to do this: – Use an experimentally determined backbone structure – Assemble secondary structural elements by hand – Use a fragment assembly program like Rosetta, selecting fragment combinations that fit some approximate desired structure

**Optimization methods for sidechain rotamers** • 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)

**Flexible backbone design** - better to have "wiggle room" in backbone. • This requires optimizing simultaneously over rotamers and backbone geometry. – Often addressed through a Monte Carlo search procedure that alternates between local tweaks to backbone dihedrals and changes to side-chain rotamers

**Negative design** • 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 can't directly avoid all possible incorrect structures without making the problem much more complicated. So the overall approach is still heuristic.

**Experimental methods** Computational protein design is often combined with experimental protein engineering methods • For example, computational designs can often be improved by directed evolution – Directed evolution involves introducing random mutations to proteins and picking out the best ones – Usually this is done in living cells, with the fittest cells (i.e., those containing the "best" version of the protein) selected by some measure

**Mean filter**

- The simplest way is to use a "mean filter"
  - Replace each pixel by the average of a set of pixels surrounding it
  - For example, a 3x3 mean filter replaces each pixel with the average of a 3x3 square of pixels
  - This is equivalent to convolving the image with a 3x3 matrix:

$$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = 1/9 \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Why 1/9?
Values should sum to 1, so that overall brightness of the image remains constant

**Median filter** A median filter ranks the pixels in a square surrounding the pixel of interest, and picks the middle value • This is particularly effective at eliminating noise that corrupts only a few pixel values (e.g., saltand-pepper noise) • This filter is not a convolution

**High pass filter** • A high-pass filter removes (or reduces) low-frequency components of an image, but not high-frequency ones • The simplest way to create a high-pass filter is to subtract a lowpass filtered image from the original image – This removes the low frequencies but preserves the high ones – The filter matrix itself can be computed by subtracting a low-pass filter matrix (that sums to 1) from an "identity" filter matrix (all zeros except for a 1 in the central pixel)

**Applications of high pass filter** • To highlight edges in the image • To remove any "background" brightness that varies smoothly across the image • Image sharpening – To sharpen the image, one can add a high-pass filtered version of the image (multiplied by a fractional scaling factor) to the original image – This increases the high-frequency content relative to low-frequency content – In photography, this is called "unsharp masking"

## Microscopy

**Light microscopy** • Basic idea: – Shine light on a biological sample (i.e., one or more cells) – Measure the light that is reflected or transmitted – Use lenses • Why do we need lenses in a microscope?

**Lenses in microscopy** • The lenses in a microscope do two things: – Magnify the image – Focus the image, so that much of the light coming from a particular point in the sample ends up focusing on a particular point on either your retina or a sensor (e.g., CCD) • You need a lens to form a clear image, even if you have a very high-resolution sensor

**Fluorescence microscopy** • Make the molecules of interest glow • Attach a fluorophore (fluorescent molecule) to the molecule of interest • When you shine light of a particular wavelength on a fluorophore, it emits light of a different wavelength – Additional advantage: not only does the molecule glow, the light it emits has a different wavelength than the incident illumination, making it easier to isolate

**Diffraction limit** The physics of light—in particular, the fact that it is a wave—impose a fundamental limit on how well a lens can focus it • The light from a single point in space will not focus to a single point • Instead, it will focus to a disk-like pattern called an "Airy pattern" – This means the observed image will be slightly blurred – In fact, we can think of the observed image as the true image convolved with the Airy pattern. This constitutes a low-pass filter.
• Resolution limit of a light microscope: – The wavelength of visible light is 400–700 nm – A light microscope can't distinguish points that are closer than 200 nm • Many cellular structures are smaller than this. A protein is just a few nm across

**Beating diffraction limit. Decrease wavelength** • Higher-frequency radiation (e.g., x-rays) has shorter wavelengths and thus allows higher resolution – It also damages the sample more • It's possible to image with electrons, which have a much shorter wavelength (.1 nm) – Electron microscopy can thus achieve much higher resolution – Disadvantages: can't use living cells, and molecules of interest won't glow

**Super resolution microscopy** • A number of recently developed techniques achieve resolution well beyond the diffraction limit – This requires violating some of the assumptions of that limit • I'll briefly describe the most popular of these techniques, known alternately as STORM (stochastic optical reconstruction microscopy) or PALM (photoactivation localization microscopy)

**STORM** If we have only a few fluorophores in an image, we can localize them very accurately • Thus by getting only a few fluorophores to turn on at a time, identifying their locations in each image, and combining that information (computationally) across many images, we can build a composite image of very high resolution

**Fick's law** • Suppose that particles are uniformly distributed in the y and z dimensions, and vary only in x • Let c represent concentration (a function of x) • Define the flux J as the rate at which particles diffuse across a boundary - Fick's 1st law:

$$J = -D\frac{\partial c}{\partial x}.$$

**Concentration**

- Now think of concentration and flux as a function of position *x* and time *t*
- The concentration at a particular position goes down with time if there is more flux away from that position than there is coming in to that position (in other words, if the flux at that position is increasing as one moves in the positive *x* direction)

$$\frac{\partial c}{\partial t} = -\frac{\partial J}{\partial x}$$

**Diffusion equation**

- Combining these formulae gives us:

$$\frac{\partial c}{\partial t} = -\frac{\partial J}{\partial x} = -\frac{\partial}{\partial x}\left(-D\frac{\partial c}{\partial x}\right) = D\frac{\partial^2 c}{\partial x^2}$$

$$\frac{\partial c}{\partial t} = D\frac{\partial^2 c}{\partial x^2}$$

(Can be generalized to multiple dimensions).

**Continuum approach** • Advantage: faster • Disadvantage: less accurate for small numbers of molecules • Unlike the particle-based approach, the continuum approach is deterministic

**Why is docking useful** • Virtual screening: Identifying drug candidates by considering large numbers of possible ligands • Lead optimization: Modifying a drug candidate to improve its properties – If the binding pose of the candidate is unknown, docking can help identify it (which helps envision how modifying the ligand would affect its binding) – Docking can predict binding strengths of related compounds

**MD direct approach for docking** • It is so computationally intensive that we usually cannot do it for even a single ligand – Drug molecules usually take seconds to hours to unbind from their targets. – Microsecond-timescale molecular dynamics simulations usually take days.

**Scoring functions for docking** Scoring functions used for docking tend to be empirical – Capture chemists' intuition about what makes a a ligand—receptor interaction energetically favorable (e.g., hydrogen bonding, or displacement of water from a hydrophobic binding pocket) – Parameters are often optimized based on known binding affinities of many ligands for many receptors – Some scoring functions borrow terms from molecular mechanics force fields, but a molecular mechanics force field is rarely used directly as a scoring function for docking • The scoring function is an (extremely rough) attempt to approximate the binding free energy. Molecular mechanics force fields give potential energy associated with a particular arrangement of atoms.

**How well does docking work** • The best docking protocols: – Predict a reasonably accurate pose (for ligands that do in fact bind the target protein) about half the time for rigid targets (the "easy" cases) • In these cases, one of the highly-ranked poses is usually close to the correct one – Provide useful, but far from perfect results, when ranking ligands • Tend to work best when comparing closely related ligands – Are not particularly useful when it comes to quantitatively estimating binding free energies

**Crystal** Under certain conditions, molecules line up into a regular grid (a "lattice"). e.g. table salt.
• Under certain conditions, entire proteins will pack into a regular grid (a lattice) (e.g. insulin crystals)

**Electron density** • The electron density corresponding to the 3D structure of a molecule gives the probability of finding an electron at each point in space • X-rays bounce off electrons they hit

**What causes diffraction patterns?** • Short answer: interference of light – The bright spots are places where light interferes constructively. Elsewhere it tends to interfere destructively (cancel out).

**Diffraction pattern; electron density** • It turns out that the diffraction pattern is the Fourier transform of the electron density – Both the electron density and the diffraction pattern are functions of three dimensions (i.e., defined at every point in a 3D volume) – Each bright spot in the diffraction pattern corresponds to one sinusoidal component of the electron density – The Fourier transform gives a magnitude and a phase for each sinusoid, but it's only practical to measure the magnitude, not the phase • Brightness of the spot gives the magnitude

**Initial phasing** • The most common method for initial phasing is molecular replacement – Start with a computational model of the protein structure (often the structure of a homologous protein) – Search over the possible ways that a protein with this structure could be packed into a crystal, and find the one that gives the best fit to the data

**Phase refinement** • Once we have an initial model, we can search for perturbations to that model that improve the fit to the experimental data – This is usually done through a Monte Carlo search (via simulated annealing) – One usually restrains the search to "realistic" molecular structures using a molecular mechanics force field • This dramatically improves the accuracy of the results • The idea was introduced by Axel Brunger, now on the Stanford faculty
- Typically done with cross validations.

**Misc notes on crystallography** • Protein crystals contain water – Often half the crystal is water (filling all the empty spaces between copies of the protein) – Usually only a few water molecules are visible in the structure, because the rest are too mobile • One usually can't determine hydrogen positions by x-ray crystallography – But one can model them in computationally • Some high-profile, published crystal structures have turned out to be completely incorrect, due to computational problems/errors

**Single particle EM (cryo electron microscopy)** • Single-particle EM has been around for decades, but it has improved dramatically in the last five years due to: – Invention of better cameras • Until recently, electrons were detected either by photographic film, or by scintillator-based digital cameras which converted electrons to photons for detection • New "direct-electron detectors" can detect electrons directly, substantially improving image resolution and quality – Better computational reconstruction techniques • Single-particle EM is thus coming into much wider use, and may challenge crystallography as the dominant way to determine experimental structures

**Single particle EM vs. X ray crystallography** • Single-particle EM's major advantage over crystallography is that it does not require formation of a crystal – Particularly advantageous for large complexes, which are usually difficult to crystallize – Also avoids structural artifacts due to packing in a crystal lattice. In EM, particles are in a more natural environment.
• On the other hand: – Single-particle EM's resolution is (typically) lower than that of crystallography – Reconstructing structures of small proteins from EM images is difficult, because images from different orientations look similar (i.e., "a blob") • Bottom line: single-particle EM is particularly advantageous for large complexes, because: – Large complexes tend to be harder to crystallize – The computational reconstruction problem in single-particle EM is usually easier to solve for large particles than for small ones

**Single particle EM uses transmission electron microscopy** • In transmission electron microscopy, a beam of electrons pass through a thin sample before forming an image

**Sample preparation** • To survive in the electron microscope (in a vacuum, under electron bombardment), the particles are usually prepared in one of two ways: – Negative staining • Coat particles with heavy metal salt crystals • This increases contrast (particles are easy to pick out from background) • It limits resolution to 20 Å and can introduce artifacts – Vitrification • Particles are embedded in ice (vitreous ice: flash frozen, not crystalline) • This gives less contrast, but enables much higher resolution (below 4 Å) • High-resolution single-particle EM relies on vitrification and is thus referred to as cryo-electron microscopy (cryo-EM)

**Reconstruction methods** • 2D image analysis: First, go from raw image data to higher-resolution 2D projections – Image preprocessing (averaging; high pass filtering) – Particle picking – Image clustering and class averaging (K means clustering) • 3D reconstruction: Then use these higherresolution projections to build a 3D model – Background: Reconstruction with known view angles – Structure refinement with unknown view angles – Calculating an initial structure – Fitting atomic-resolution models to lower-resolution EM structures

**K means clustering** • Pick k random images as class exemplars • Then iterate the following: – Assign each image to the closest exemplar – Average all the images in each class to determine a new class exemplar

**Back projection** • How would you reconstruct the 3D density map from 2D projections? – Same problem is encountered in medical imaging (e.g. in CT scans, which are basically 3D x-rays) • The simplest approach would be back-projection: reverse the projection process by "smearing" each projection back across the reconstructed image

**Filtered back projection** Apply a specific high pass filter to each image before back projection.
Projection slice theorem (2D version): The 1D Fourier transform of the 1D projection of a 2D density is equal to the central section—perpendicular to the direction of projection—of the 2D Fourier transform of the density This theorem holds because each of the 2D sinusoids used in the 2D Fourier transform is constant in one direction

**Structure refinement** If we're not given the view angles for each particle, but we have a decent initial 3D model, then iterate the following steps to improve the model: – For each projection (i.e., each class average), find the view angle that best matches the 3D model – Given the newly estimated view angles, reconstruct a better 3D model (e.g., using filtered back-projection) • This is called 3D projection matching

**Avoiding overfitting** Structure refinement methods are prone to overfitting – Converged model can show features that don't really exist and just reflect noise in the images (analogous to the issue with image clustering) – A variety of methods have been developed recently to deal with this issue • Some use Bayesian approaches (e.g., RELION software) • Some of the most important recent algorithmic developments in single-particle EM are in this area.

**Atomic resolution models from lower resolution EM** Often we have high-resolution xray crystallography structures of each individual protein in a complex whose low-resolution structure was determined by single-particle EM. • We can fit the high-resolution structures into the EM density.

**Image preprocessing, problem 3** Problem 3: The optics cause the recorded image to be a blurred version of the ideal image – This blurring is a convolution, and can thus be expressed as a multiplication in the frequency domain, where the ideal image is multiplied by the "contrast transfer function" • Solution: Estimate parameters of the contrast transfer function, then correct for it – Some of the parameters are known (from the optics), while others are estimated from the images – Correction is generally done in the frequency domain

**How to get an initial structural model** Multiple options: – We might have one from prior experimental work – Conduct specialized experiments, often at lower resolution • Example: random canonical tilt approach, which requires collecting each image twice, from different angles – Direct computational solution • Common lines method: relies on the fact that Fourier transforms of different 2D projections share a common line • Stochastic hill climbing: a robust projection matching (refinement) approach that often allows random initialization

**Faster STORM using compressed sensing** Zhu et al. 2012, Nature methods. In this research, Zhu et al.[6] present a compressed sensing technique allows for sparse-signal re- covery for efficient STORM (stochastic optical reconstruction microscopy). Live cell imaging is particularly challenging because high temporal resolution is required[3][1]. STORM is a technique in which multiple samples are combined to retrieve single-molecule position data. Previ- ous approaches have applied maximum likelihood estimation or Bayesian statistics to perform this inference[5][4]. The authors apply a novel compressed-sensing based approach, which is well-suited for problems when the signal of interest is sparse. In particular, compressed sensing presents a framework to recover signal from highly noisy measurements. Applying their method, the re- searchers demonstrate the capability to handle activated fluorophore density an order of magnitude greater than previous approaches.

Consider a linear measurement **b** of the original signal **x**:

$$\mathbf{b} = \mathbf{A}\mathbf{x}.$$

where the matrix **A** is a known measurement function. Compressed sensing recovers the sparse signal **x** by minimizing its L1 norm:

$$\text{minimize } \|\mathbf{x}\|_1 \text{ subject to } \mathbf{b} = \mathbf{A}\mathbf{x}.$$

This study considers a variant of this traditional formulation, in which the following minimization is performed:

$$\text{Minimize } c^T\mathbf{x}$$
$$\text{Subject to } \mathbf{x} \geq 0 \text{ and } \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2 \leq \epsilon \cdot (\textstyle\sum_i b_i)^{1/2}.$$

The researchers apply a grid discretization to model the molecule positions, and the camera image is modeled as a convolution of the fluorophore distribution with a point-spread function of the imaging system. This global optimization problem can solved with classical linear programming strategies with quadratic constraints. The authors use a streamlined implementation that performs the optimization at local patches of the original image. This was found to produn little impact to reconstruction performance, due to the fact that the point-spread function is very narrow. The performance of the algorithm is evaluated by simulating images so that the analysis can be compared to ground truth molecule positions (see Figure 1). The simulation is fairly detailed, accounting for a) the variation of emitted photos from each molecule, b) the experimentally mea- sured point spread function, and c) the background and photon counting noise. During these simulated experiments, the authors set the peak of the log-normal photon number distribution at 3,000 photons. The paper presents fairly impressive results in position recovery precision, measured in FWHM (full-width at maximum) of nanometers. At a molecule density of 6 $\mu$m$^{-2}$, the localization preci- sion FWHM of compressed sensing is approximately 70nm, while that of single-molecule fitting is approximately 110nm. One drawback of the method is that compressed sensing has worse precision than a conventional single-molecule fitting approach for samples of low particle densities (less than 1.5 $\mu$m$^{-2}$).
The authors also test their algorithm on microtubules in Drosophila S2 cells, which demonstrates that their method can capture dynamics on a time resolution of 3s (see Figure 2). The compressed sensing algorithm was benchmarked against a single-molecule fitting algorithm used in other studies. The single-molecule fitting approach uses elliptical Gaussian functions to fit a local maximum in the input image, and then performs a thresholding in peak height, peak width, and ellipticity to prune overlapped molecules. In particular, the authors demonstrate that compressed sensing enables an increase in imaging speed by 6 to 15-fold compared to a conventional fitting method and 2 to 3-fold compared to DAOSTORM [2]. For future applications, the researchers note that compressed sensing can enable STORM to im- age processes with sub-second-scale resolution if a fast EMCCD or scientific CMOS camera is used. Furthermore, the methodology presented herein can be adapted to 3D super resolution microscopy.
In conclusion, the authors present a compressed-sensing based approach to STORM that achieves high localization precision and detailed time resolution. One shortcoming of the paper is simulations may not accurately capture the nuances of the error distribution observed experimentally. A future extension could be to develop experimental techniques (possibly using certain classes of well- controlled cell-types) that can produce accurate ground-truth data for performance analysis.