Review

CS/BioE/Biophys/BMI/CME 279

Dec. 3 and 5, 2024

Ron Dror

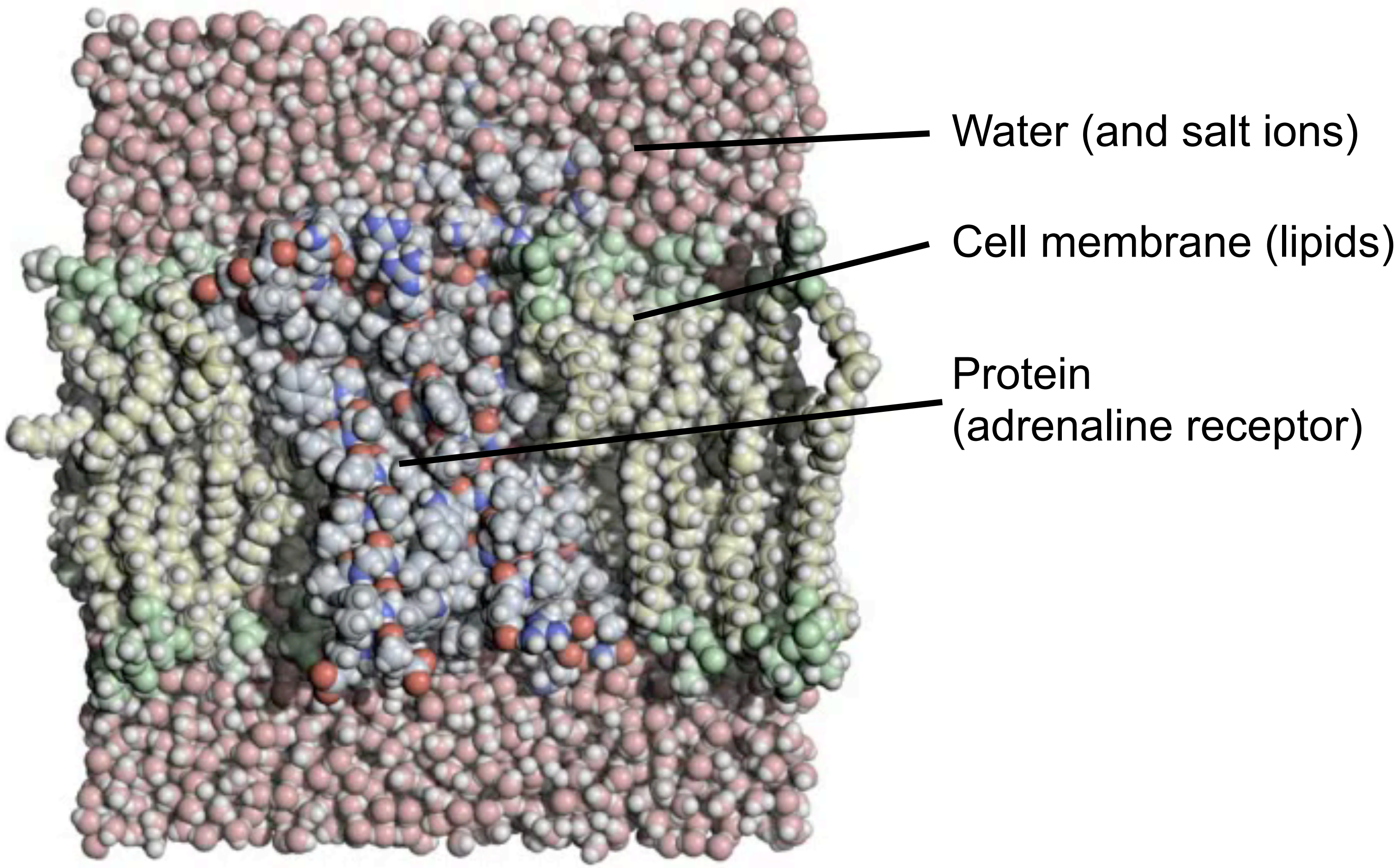
Outline

- Atomic-level modeling of biological macromolecules
  - Biomolecular structure (including protein structure)
  - Energy functions and their relationship to molecular conformation
  - Molecular dynamics simulation
  - Predicting structures of proteins and other biomolecules
  - Protein design
  - Ligand docking and virtual screening
- Coarser-level modeling and imaging-based methods
  - Fourier transforms and convolution
  - Image analysis
  - X-ray crystallography
  - Cryo-electron microscopy
  - Microscopy
  - Diffusion and cellular-level simulation
- Recurring themes

Atomic-level modeling of biological macromolecules

Atomic-level modeling

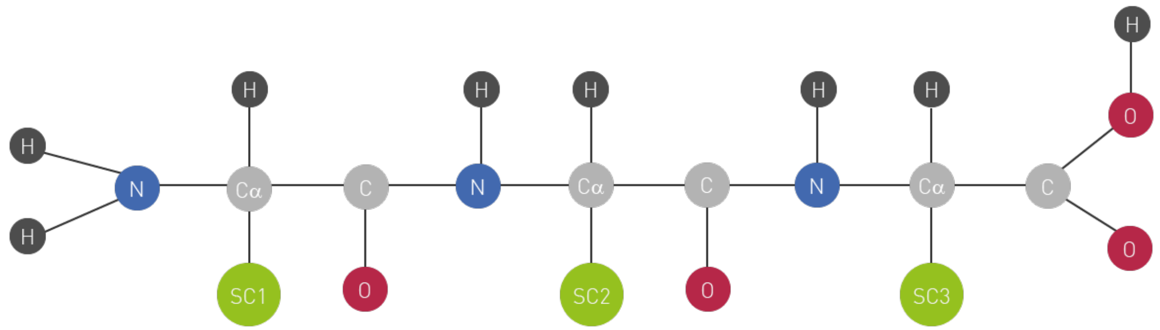
Biomolecular Structure



- Proteins are constantly jiggling around, as are the molecules that surround them (mostly water).
- Each protein—or other macromolecule—can in fact assume many structures, but they tend to be similar to one another. Usually we talk about the “average” structure, which is (roughly) what’s determined experimentally and deposited in the PDB.
- The surrounding molecules play a key role in determining protein structure.

Two-dimensional protein structure

- Proteins are chains of amino acids
- These amino acids are identical except for their side chains
  - Therefore proteins have regular (repeating) backbones with differing side chains
  - The different side chains have different chemical properties, and they ultimately determine the 3D structure of the protein.



From https://www.bmglabtech.com/en/blog/protein-assays/

What determines the three-dimensional structure of a protein or other biomolecule?

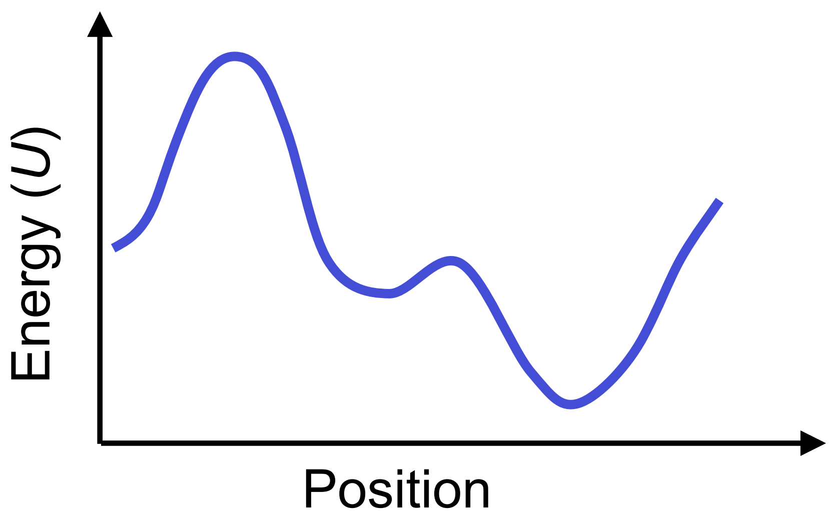
- Basic interactions
  - Bond length stretching
  - Bond angle bending
  - Torsional angle twisting
  - Electrostatic interaction
  - Van der Waals interaction (attractive and repulsive)
- Complex interactions
  - Hydrogen bonds
  - Hydrophobic effect
  - These “complex interactions,” which result from the basic interactions above, are particularly important to understanding why biomolecules adopt particular 3D structures

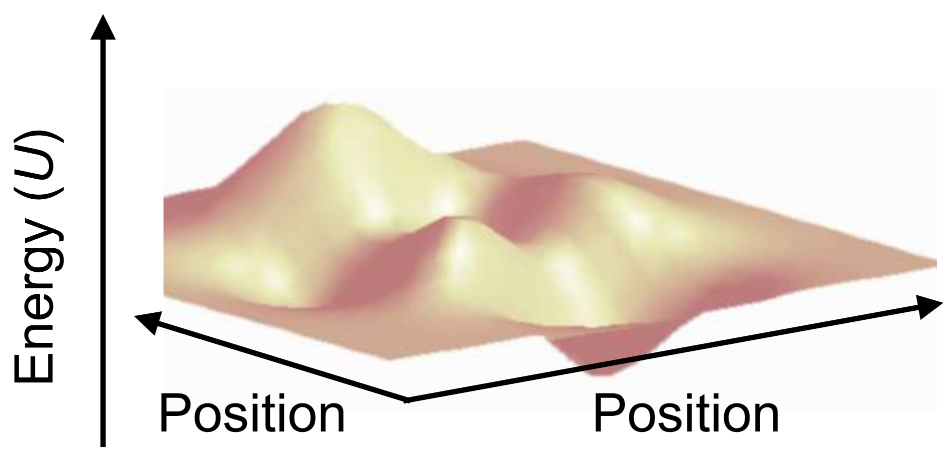
Atomic-level modeling

Energy functions and their relationship to molecular conformation

Potential energy functions

- A potential energy function $U(\boldsymbol{x})$ specifies the total potential energy of a system of atoms as a function of all their positions ($\boldsymbol{x}$)
  - For a system with n atoms, $\boldsymbol{x}$ is a vector of length $3n$ (x, y, and z coordinates for every atom)
  - In the general case, include not only atoms in the protein but also surrounding atoms (e.g., water)
- The force on each atom can be computed by taking derivatives of the potential energy function





Example of a potential energy function: molecular mechanics force field

$$U = \sum_{\text{bonds}} k_b \left(b - b_0\right)^2$$

Bond lengths ("Stretch")

$$+ \sum_{\text{angles}} k_\theta \left(\theta - \theta_0\right)^2$$

Bond angles ("Bend")

$$+ \sum_{\text{torsions}} \sum_{n} k_{\phi,n} \left[1 + \cos\left(n\phi - \phi_n\right)\right]$$

Torsional/dihedral angles

Bonded terms

$$+ \sum_{i} \sum_{j>i} \frac{q_i q_j}{r_{ij}}$$

Electrostatic

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

Van der Waals

Non-bonded terms

Above is the form of a typical molecular mechanics force field (as used in molecular dynamics simulations, for example). The terms correspond to the "basic interactions" discussed previously.

The Boltzmann Distribution

- The Boltzmann distribution relates the potential energy of a particular arrangement of atoms ("conformation") to the probability of observing that arrangement of atoms (at equilibrium):

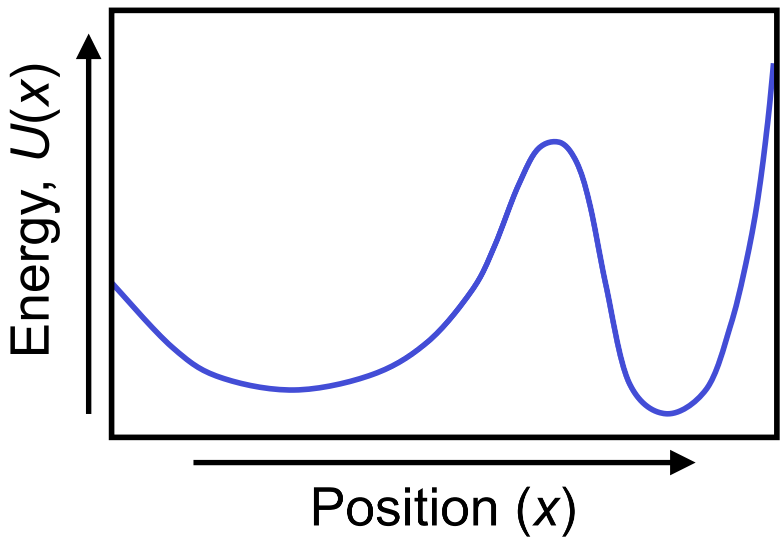
$$p(\boldsymbol{x}) \propto \exp\left({-U(\boldsymbol{x})} \middle/ {k_B T} \right)$$

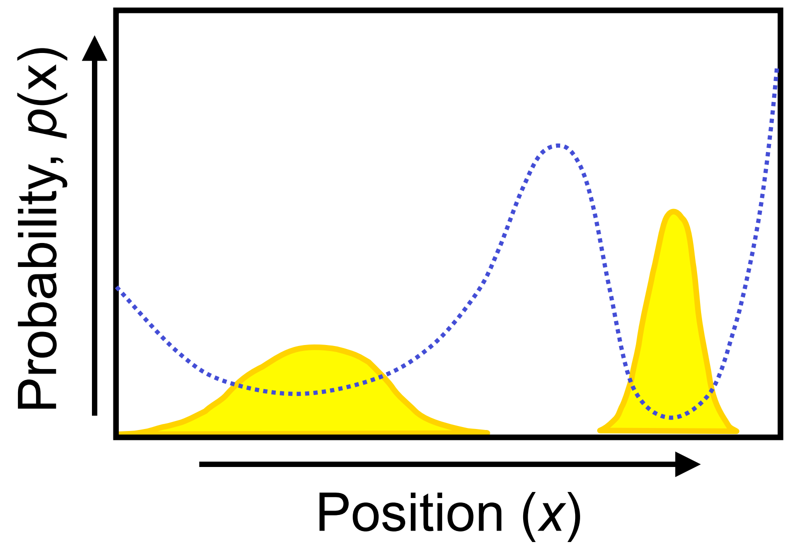
Equivalently,

$$p(\mathbf{x}) = \frac{1}{Z} \exp\left(\frac{-U(\mathbf{x})}{k_B T} \right)$$

where T is temperature and k_B is the Boltzmann constant

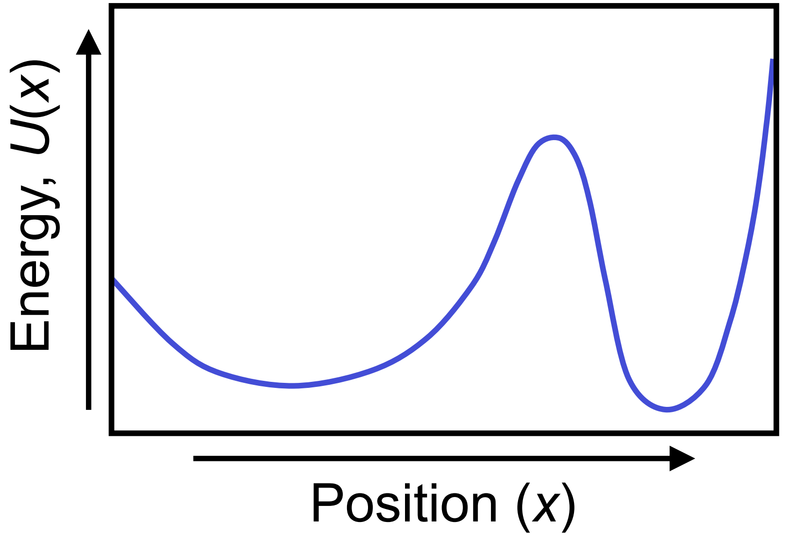
- Note: Z is chosen such that the probabilities sum to 1 across all arrangements of atoms. It depends on U and T but not on $\mathbf{x}$.

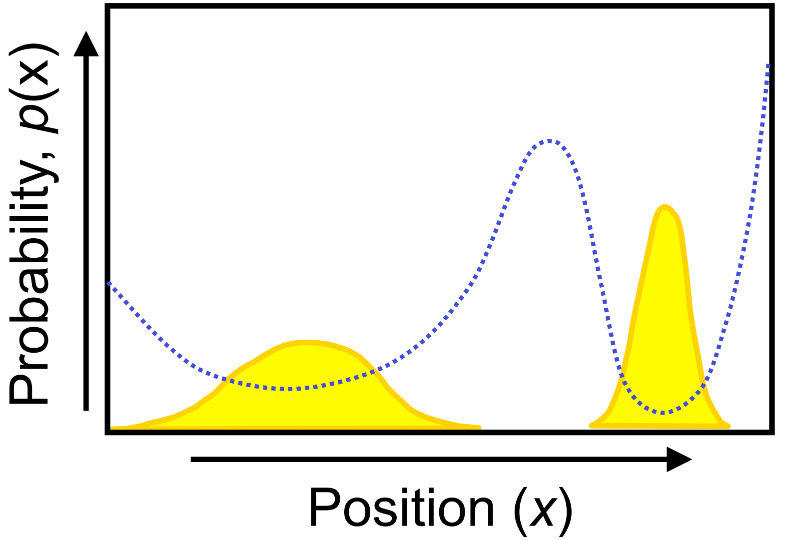




Conformational states (macrostates)

- We typically care most about the probability that protein atoms will be in some *approximate* arrangement, with *any* arrangement of surrounding atoms
- We thus care about the probability of sets of atomic arrangements, called conformational states
  - These correspond roughly to wells of the potential energy function
  - To calculate probability of a well, we sum the probabilities of all the specific atomic arrangements (conformations) it contains





Free energies

- The free energy G_A of a conformational state A satisfies:

$$P(A) = \exp\left({-G_A} \big/ {k_B T} \right)$$

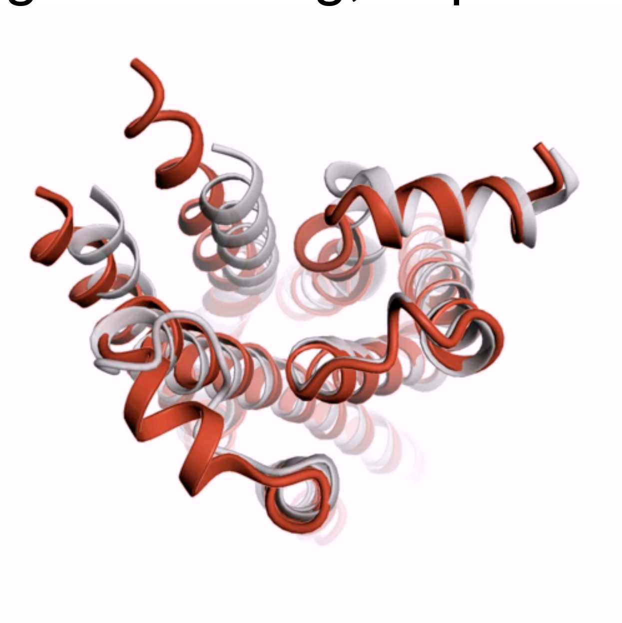
- Clarifications:
  – Free energies are clearly defined only for conformational states/macrostates
  – However, in protein structure prediction, protein design, and ligand docking, it's often useful to define a "free energy function" that approximates a free energy for some *neighborhood* of each arrangement of protein atoms
    - To predict protein structure, we minimize free energy, not potential energy
  – The term "energy function" is used for both potential energy and free energy functions

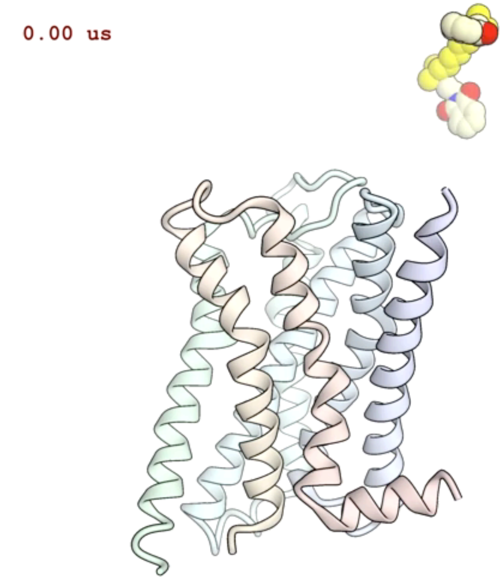
Atomic-level modeling

Molecular dynamics simulation

Molecular dynamics (MD) simulation

- An MD simulation predicts how atoms move around based on physical models of their interactions
- Of the atomic-level modeling techniques we covered, this is closest to the physics; it attempts to predict the real dynamics of the system
- It can thus be used to capture functionally important *processes*, including structural changes in proteins, protein-ligand binding, or protein folding





Basic MD algorithm

- Step through time (very short steps)
- At each time step, calculate force acting on every atom using a molecular mechanics force field
- Then update atom positions and velocities using Newton's second law

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{v}$$

$$\frac{d\boldsymbol{v}}{dt} = \frac{F(\boldsymbol{x})}{\boldsymbol{m}}$$

Note: this is inherently an approximation, because we're using classical physics rather than quantum mechanics. (Quantum mechanical calculations are used to determine force field parameters, however.)

Sampling

- Given enough time, an MD simulation will sample the full Boltzmann distribution of the system
  - This means that if one takes a snapshot from the simulation after a long period of time, 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

$$e^{-\Delta U / k_B T}$$

Atomic-level modeling

Predicting structures of proteins and other biomolecules

Protein structure prediction

- 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 structures and sequences

Approaches to protein structure prediction (i.e., what information can we leverage?)

- Template-based modeling (homology modeling)
  - Useful when one can identify one or more likely homologs of known structure (usually the case)
- Multiple sequence alignment (coevolution) methods
  - Rely on sequences of many homologs without requiring that their structures are known
- *Ab initio* structure prediction
  - Does not require identifying homologs of the query sequence
  - Even *ab initio* approaches usually take advantage of available structural data, but in more subtle ways

Template-based structure prediction: basic 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.

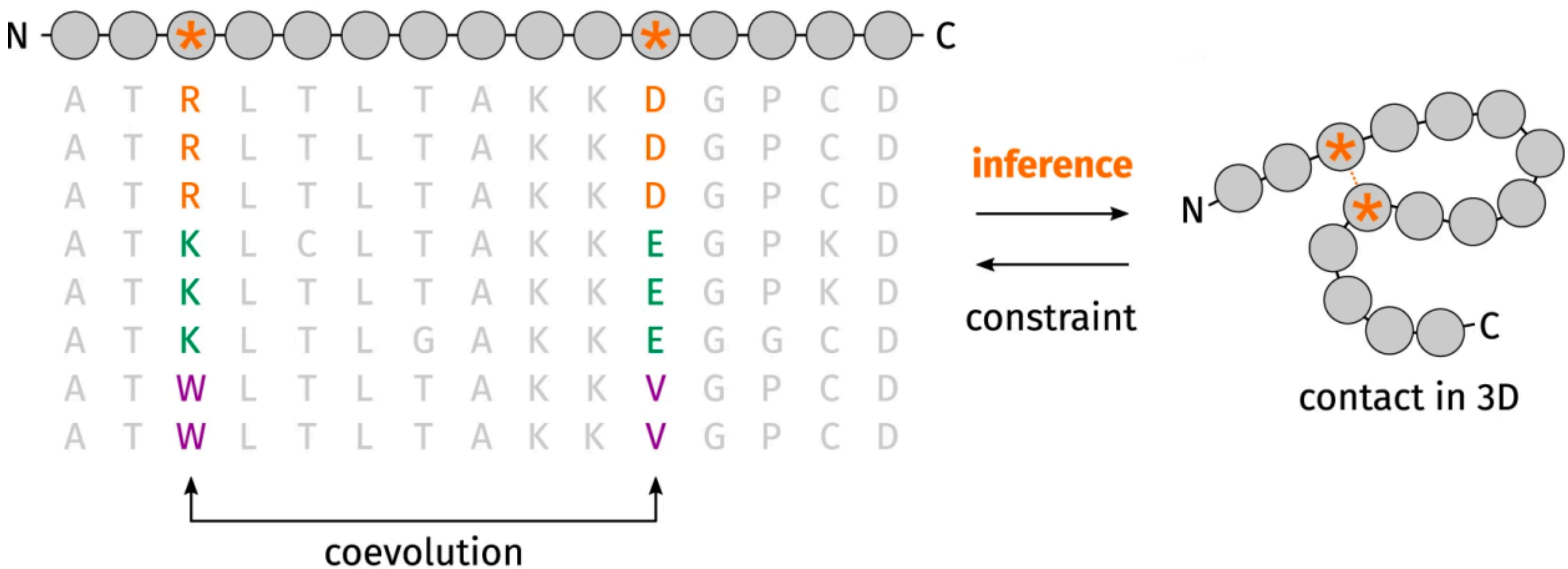
Principle underlying template-based modeling: structure is more conserved than sequence

- 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

Multiple sequence alignment (co-evolution) methods

- Even if no *structure* of a related protein is available, one can frequently still find many *sequences* of related proteins
- One can use these sequences to infer information about the relative positions of amino acid residues in 3D

Amino acids in direct physical contact tend to covary or "coevolve" across related proteins



From: https://www.blopig.com/blog/2021/07/alphafold-2-is-here-whats-behind-the-structure-prediction-miracle/

- Given many sequences of related proteins, amino acids that coevolve are probably close together
- This approach again exploits the fact that structure is more conserved than sequence during evolution

Ab initio structure prediction (as exemplified by 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 being modeled with fragments of protein structures from the PDB

Recent deep learning methods for protein structure prediction leverage multiple sources of information

- First-generation deep learning methods extracted more information from multiple sequence alignments
- More recent deep learning methods (e.g., AlphaFold 2/3 and RoseTTAFold) do more:
  - Take both multiple sequence alignments and templates as inputs (that is, sequences and structures of related proteins)
  - Learn favorability of local arrangements of amino acid residues and their constituent atoms (i.e., side-chain packing) from very large numbers of available protein structures
  - Learn how to combine these sources of information effectively
  - Generalize to complexes of proteins and other biomolecules

Atomic-level modeling

Protein design

Overall goal

- Design a protein to serve a particular function or purpose
  - In particular, choose the appropriate amino acid sequence

Typical protein design workflow

1. Based on design goals (e.g., desired function), choose structural requirements (e.g., some small part of the protein must adopt a particular structure, with the rest holding that part in place)

   Human expert judgement

2. **Structure design**: select a backbone structure that is compatible with structural requirements and that is "designable" (i.e., "achievable" by actual proteins)

   Computation (or Human expert judgement)

3. **Sequence design**: select an amino acid sequence that will adopt (fold into) the desired backbone structure

   Computation

4. Perform wet-lab experiments to check which designs work, and (optionally) to improve them

   Wet-lab experiments

Structure design

- Goal: select a target structure for the protein backbone
- This can be challenging, because only certain backbone structures are achievable, and it’s not easy to tell which structures those are
- Traditional approaches include assembling secondary structure elements by hand, modifying the backbone structure of an existing protein, or assembling fragments of existing protein structure

Structure design

- Recent machine learning approaches such as RFDiffusion make this process more systematic and, often, more reliable
  - Use experimentally determined backbone structures of real proteins to learn a generative model that produces backbone structures with desired characteristics specified by the user
  - Roughly analogous to using DALL-E to generate images, but instead of providing a description in text, one usually provides a quantitative description of what's desired (e.g., desired local structure, symmetry, or binding to a particular molecule)

Sequence design

- Goal: given a desired approximate three-dimensional backbone structure, find an amino acid sequence that will fold to that structure
- In principle, we could accomplish this by doing structure prediction for every possible sequence, but that's not practical

Traditional approach to sequence design, exemplified by Rosetta

- Focus on the desired backbone structure and find the sequence that minimizes its energy (relative to the unfolded state)
  - Energy is generally estimated 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
- The minimization problem is usually solved with heuristic methods (e.g., Metropolis Monte Carlo)

Recent machine learning–based approaches to sequence design, exemplified by ProteinMPNN

- Train a machine learning method to predict amino acid sequences of real proteins given their experimentally backbone structures
  - Or, better, given approximate backbone coordinates: instead of using the exact experimentally determined coordinates, add some noise (small random numbers)
- Methods like ProteinMPNN can generate many candidate sequences for any given backbone structure
  - ProteinMPNN effectively learns to approximate the free energy of a sequence when adopting a target backbone structure, and then samples sequences that result in low values of this (very approximate) free energy

Computational protein design is heuristic but effective

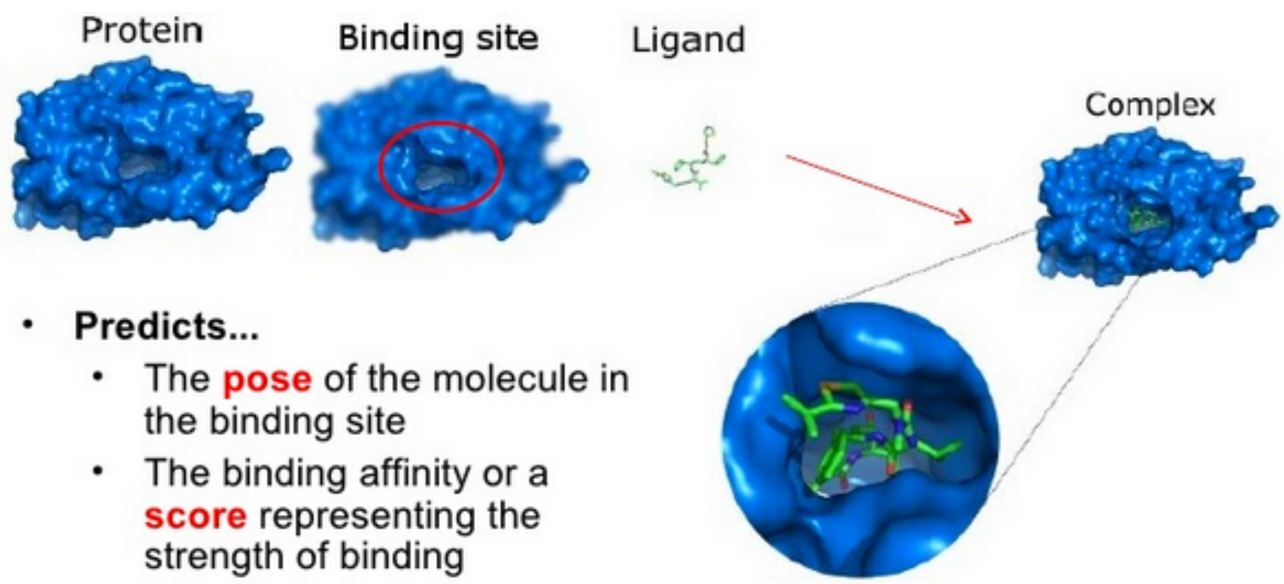
- Practical protein design methodologies involve multiple simplifying assumptions
- They’re heuristic (that is, based on algorithms that aren’t guaranteed to work but tend to work in practice)
- One generally produces many candidate designs and tests them experimentally in the hope that at least one will work
- Despite these limitations, computational protein design has proven effective and powerful for a wide variety of applications

Atomic-level modeling

Ligand docking and virtual screening

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



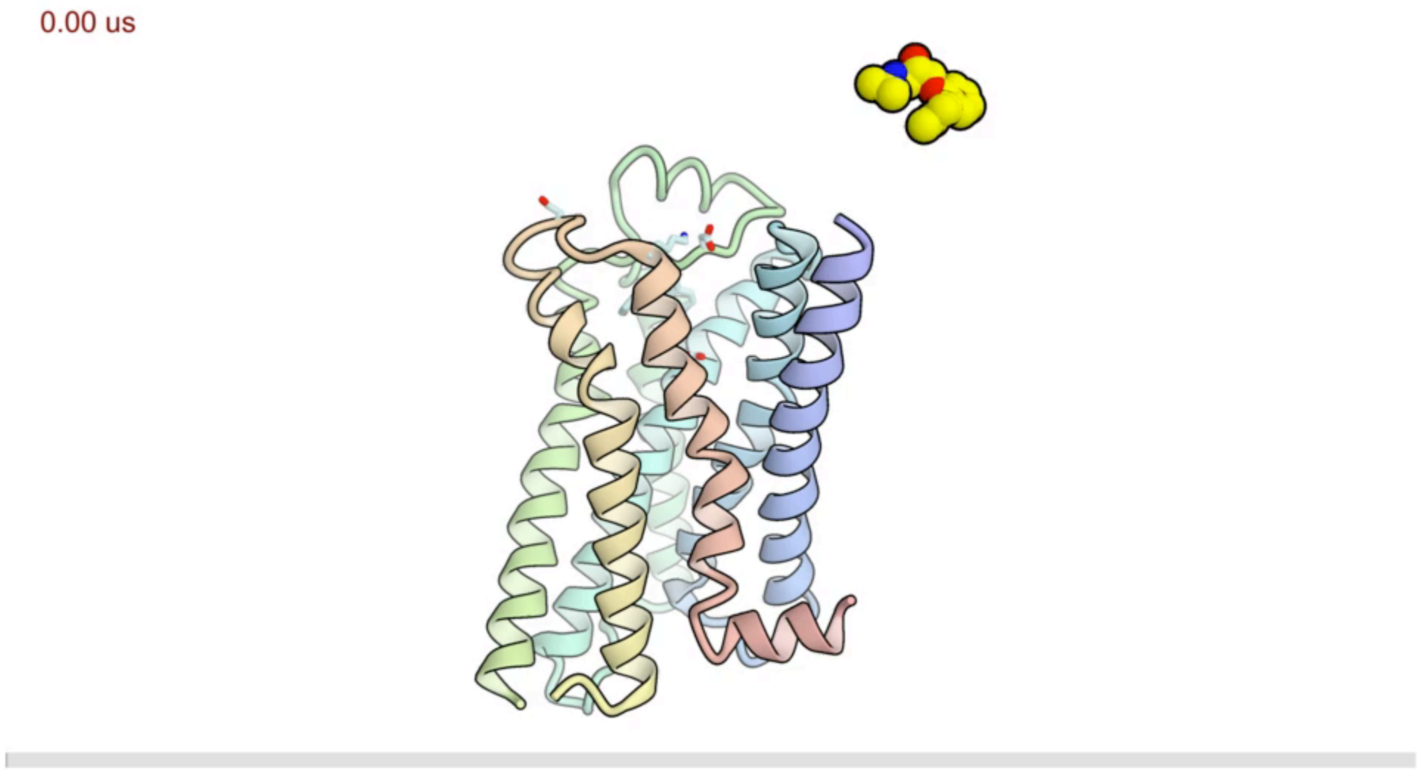
http://www.slideshare.net/baoilleach/proteinligand-docking-13581869

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 concentration of unbound ligand molecules at which half the protein molecules will have a ligand bound, or as the difference ΔG in free energy of the bound state and the unbound state

Binding affinity

- 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



Standard ligand docking methodology

- Ligand docking has two key components, both heuristic:
  - 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 (in a non-exhaustive manner)

Virtual screening

- Goal: identify ligands that bind to a target—particularly ligands that are very different from any known binder
- Typical process
  - Select a virtual library of chemical compounds
  - Use docking to roughly estimate the affinity of each
  - Buy or make a chemically diverse subset of the compounds with the best predicted affinities
  - Do experiments to test how well these compounds bind
  - Optional: optimization of experimentally validated binders by testing related chemical compounds

Coarser-level modeling and imaging-based methods

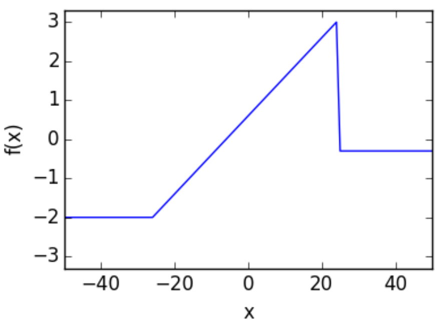
Coarser-level modeling and imaging-based methods

Fourier transforms and convolution

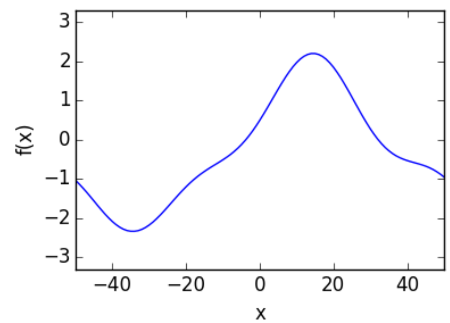
Writing functions as sums of sinusoids

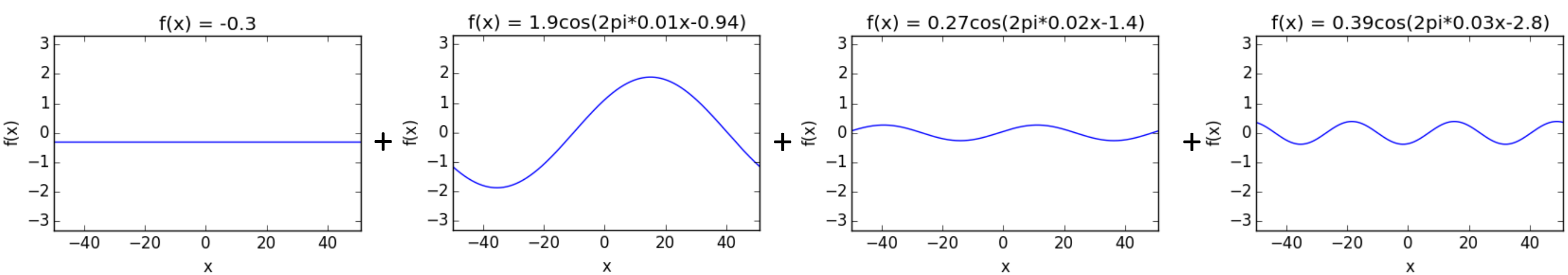
- Given a function defined on an interval of length *L*, we can write it as a sum of sinusoids with the following frequencies/periods:
  - Frequencies: 0, 1/*L*, 2/*L*, 3/*L*, ….
  - Periods: constant term, *L, L/2, L/3, …*

Original function



Sum of sinusoids below





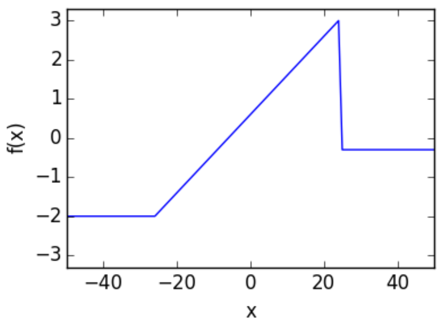
Decreasing period

Increasing frequency

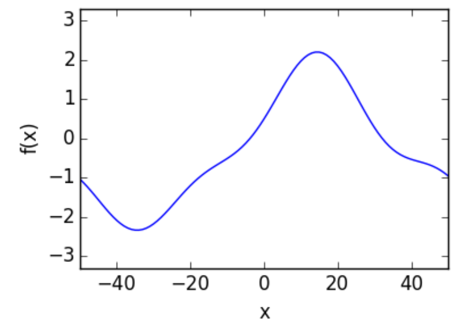
Writing functions as sums of sinusoids

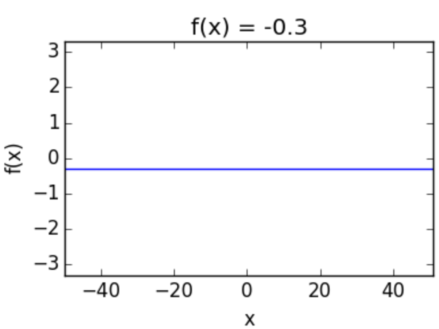
- Each of these sinusoidal terms has a magnitude (scale factor) and a phase (shift).

Original function

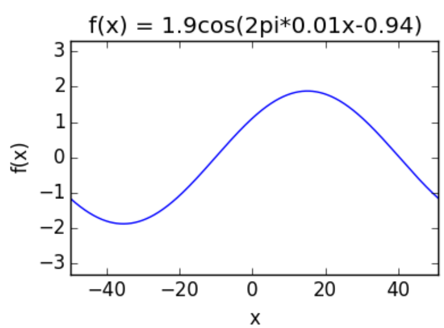


Sum of sinusoids below

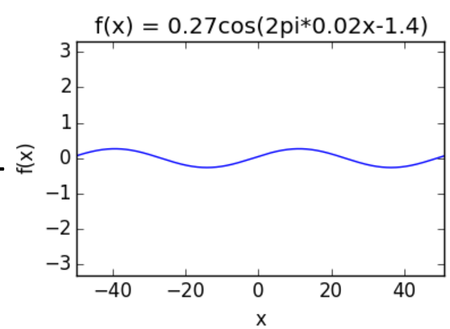




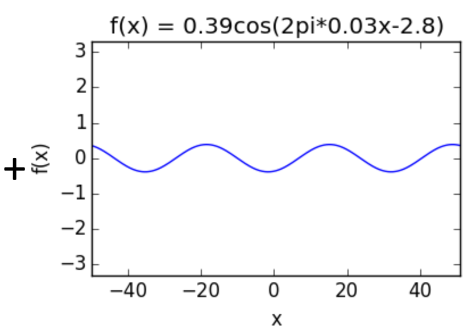
+



+



+



Magnitude: -0.3
Phase: 0 (arbitrary)

Magnitude: 1.9
Phase: -.94

Magnitude: 0.27
Phase: -1.4

Magnitude: 0.39
Phase: -2.8

The Fourier Transform: Expressing a function as a set of sinusoidal term coefficients

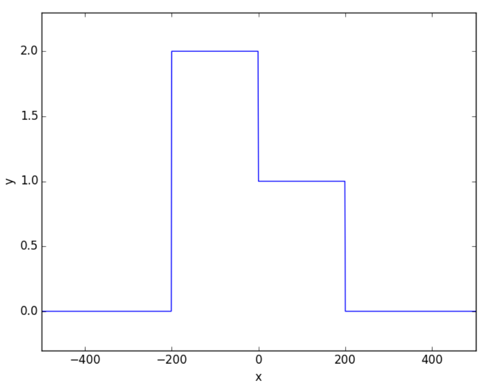
- We can thus express the original function as a series of magnitude and phase coefficients
  - We can express each pair of magnitude and phase coefficients as a complex number
- The Fourier transform maps the function to this set of complex numbers, providing an alternative representation of the function.
- 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

Constant term (frequency 0)	Sinusoid 1 (period *L*, frequency 1/*L*)	Sinusoid 2 (period *L*/2, frequency 2/*L*)	Sinusoid 3 (period *L*/3, frequency 3/*L*)
Magnitude: -0.3	Magnitude: 1.9	Magnitude: 0.27	Magnitude: 0.39
Phase: 0 (arbitrary)	Phase: -.94	Phase: -1.4	Phase: -2.8

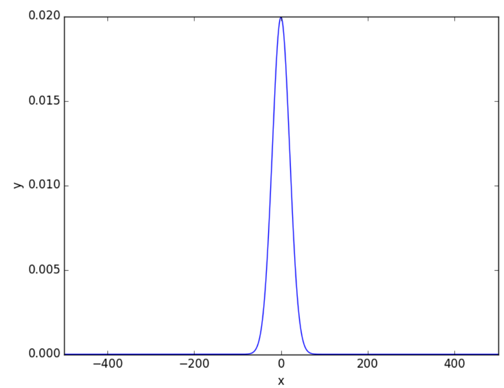
Convolution

- Convolution is a weighted moving average
  - To convolve one function with another, we compute a weighted moving average of one function using the other function to specify the weights

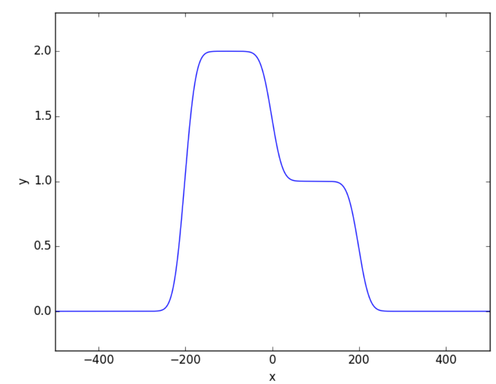
f



g



f convolved with g



Convolution = multiplication in frequency domain

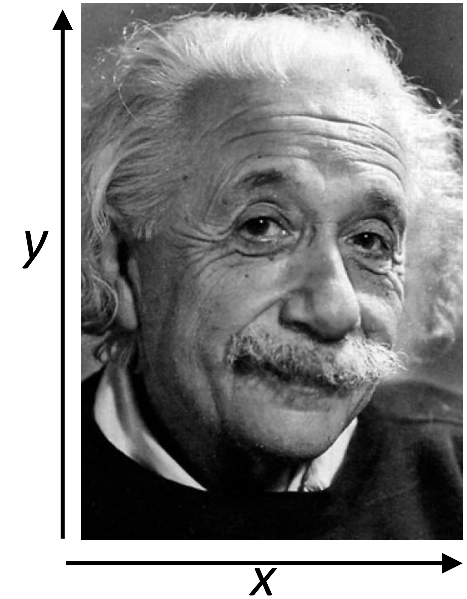
- One can compute the convolution of two functions by taking the Fourier transform of both functions, multiplying the resulting coefficients at each frequency, and then performing an inverse Fourier transform
- Why is this important?
  - It provides an efficient way to perform large convolutions (thanks to the FFT)
  - It allows us to interpret convolutions in terms of what they do to different frequency components (e.g., high-pass and low-pass filters)

Coarser-level modeling and imaging-based methods

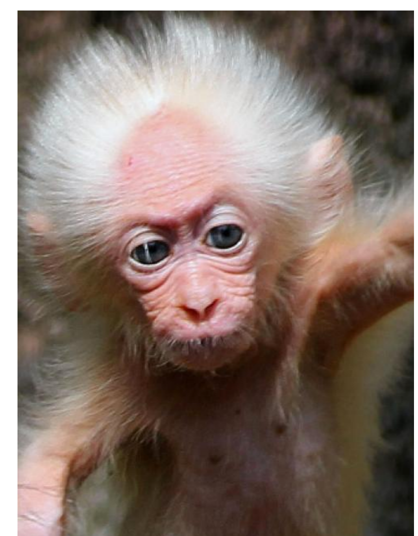
Image analysis

Representations of an image

- We can think of a gray-scale image as:
  - A two-dimensional array of brightness values
  - A function of two variables (*x* and *y*), that returns the brightness of the pixel at position (*x*, *y*)



- A color image can be treated as three separate images (red, green, blue), or as a function that returns three values (red, green, blue) for each (*x*, *y*) pair



Reducing image noise

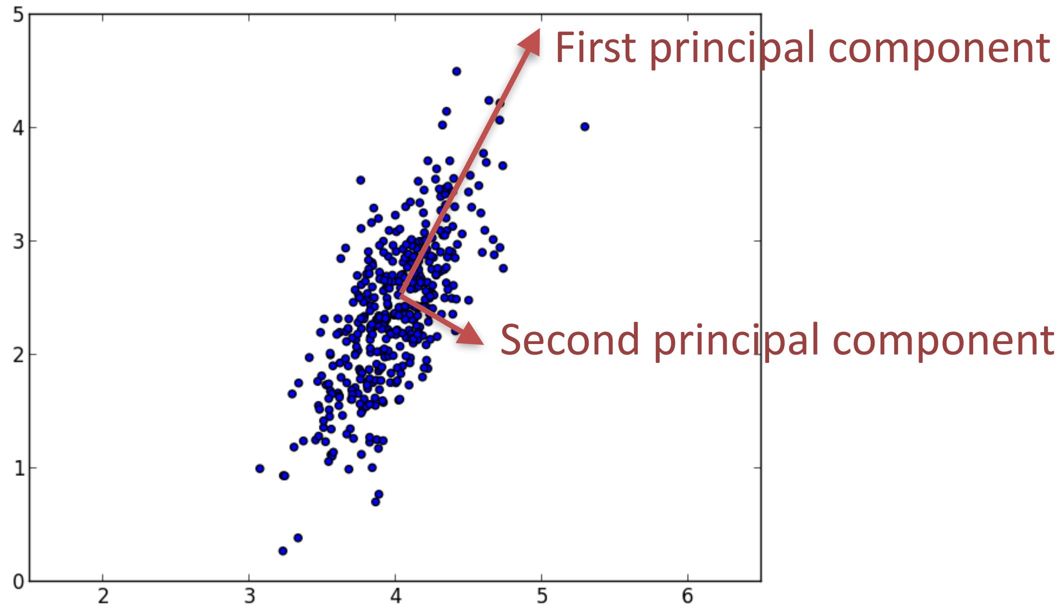
- We can reduce image noise using various filters (e.g., mean, median, Gaussian)
  - These all rely on the fact that nearby pixels in an image tend to be similar
- The mean and Gaussian filters (and many others) are convolutions, and can thus be expressed as multiplications in the frequency domain
 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

High-pass filters

- A high-pass filter reduces magnitudes of low-frequency coefficients 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

Principal component analysis (PCA)

- 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.



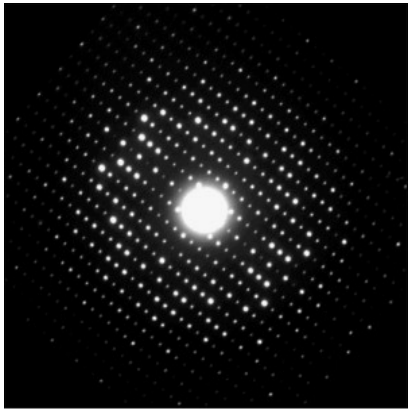
- PCA provides a way to represent high-dimensional data sets (such as images) approximately in just a few dimensions
- It is thus useful in summarization and classification of images

Coarser-level modeling and imaging-based methods

X-ray crystallography

The basic idea

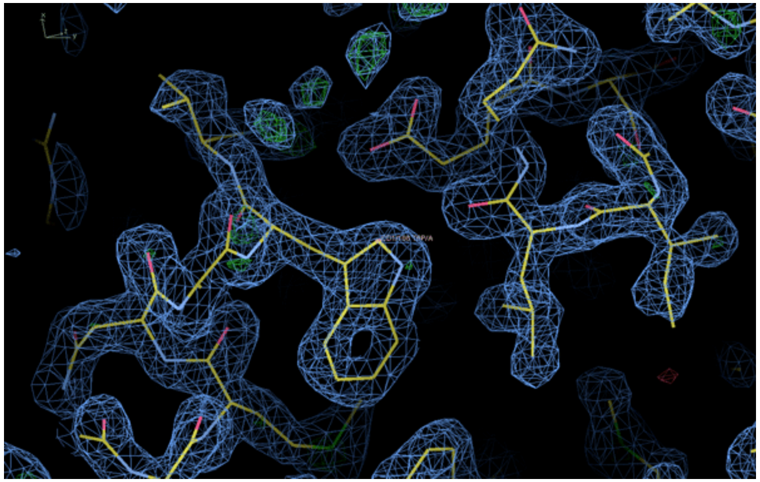
- Get the molecule whose structure you want to determine to form a crystal
- Shine an intense beam of x-rays through the crystal, giving rise to a “diffraction pattern” (a pattern of spots of varying brightnesses)
  - Shine x-rays through the crystal at multiple angles to capture the full 3D diffraction pattern
- From that pattern, infer the 3D structure of the molecule



http://lacasadeloscristales.trianatech.com/wp-content/uploads/2014/09/image005-300x300.jpg

How does the diffraction pattern relate to the molecular 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

http://www.lynceantech.com/images/electron_density_map.png

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 predicted structure as a model
- Step 2: *Phase refinement*
  - Search for perturbations that improve the fit to the experimental data (the diffraction pattern)
  - Restrain the search to “realistic” molecular structures, usually using a molecular mechanics force field

Coarser-level modeling and imaging-based methods

Cryo-electron microscopy

The basic idea

- We want the structure of a “particle”: a molecule (e.g., a protein) or 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 (atomic 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

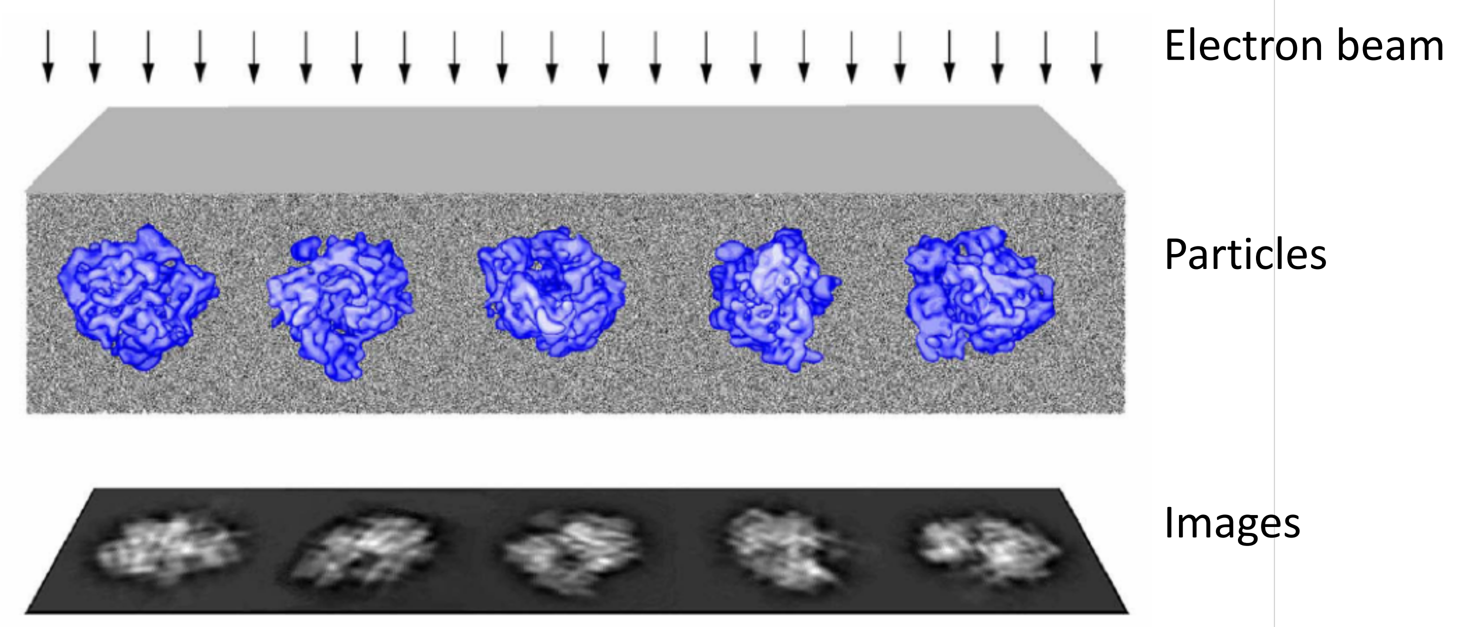
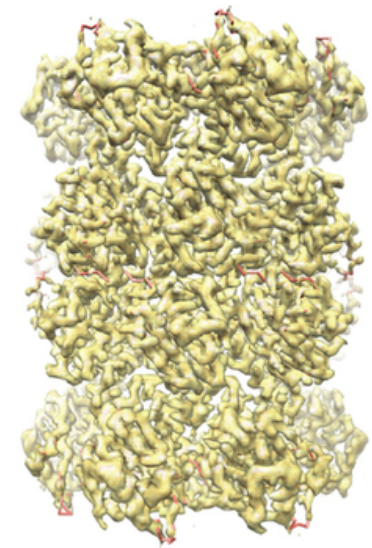


Image from Joachim Frank
http://biomachina.org/courses/structures/091.pdf

Determining the 3D structure: key steps

- **2D image analysis**: Go from raw image data to 2D projections that are somewhat cleaner, but still noisy
- **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.



Reconstructed 3D density map

Li et al., Nature Methods 10:584 (2013)

Coarser-level modeling and imaging-based methods

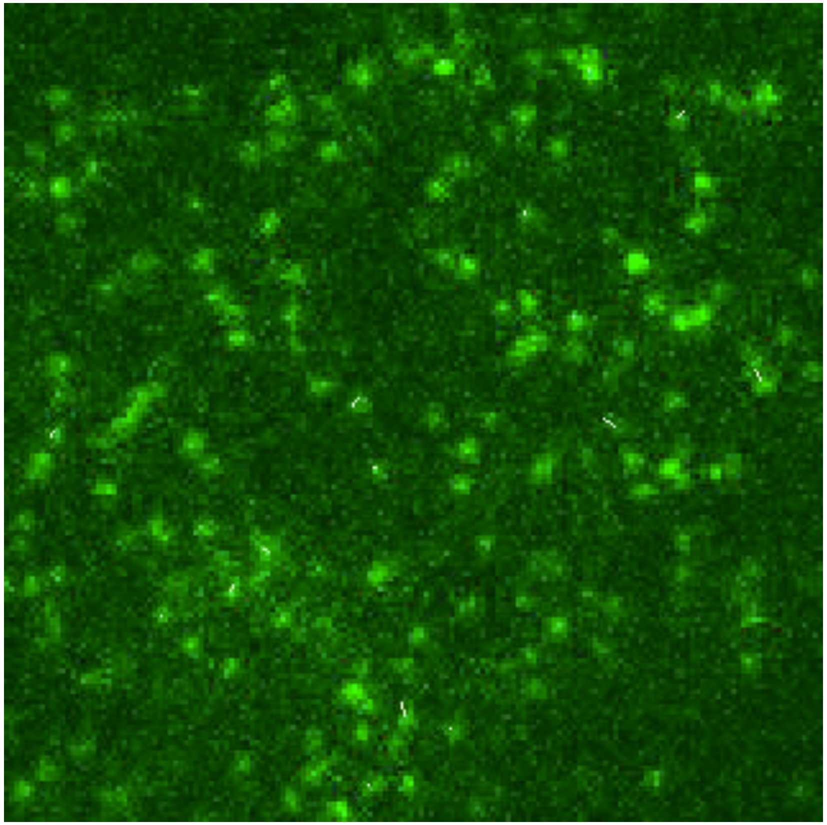
Microscopy

Fluorescence microscopy: basic idea

- 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 attaching 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



Data: Bettina van Lengerich, Natalia Jura
Tracking and movie: Robin Jia

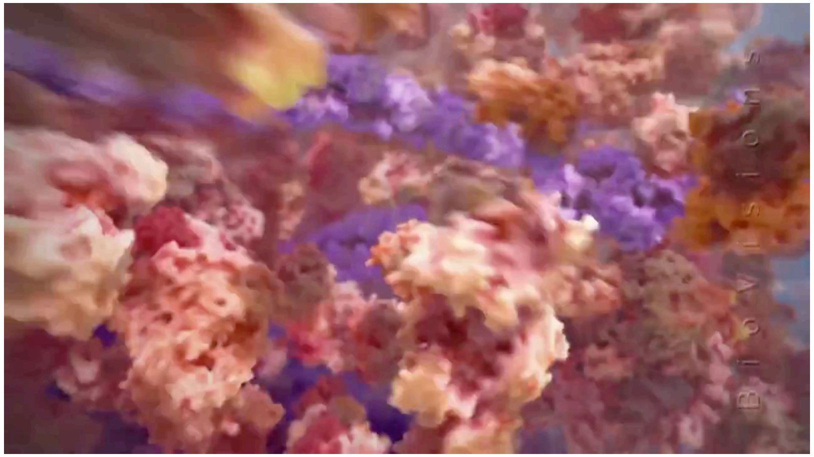
The diffraction limit

- 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.

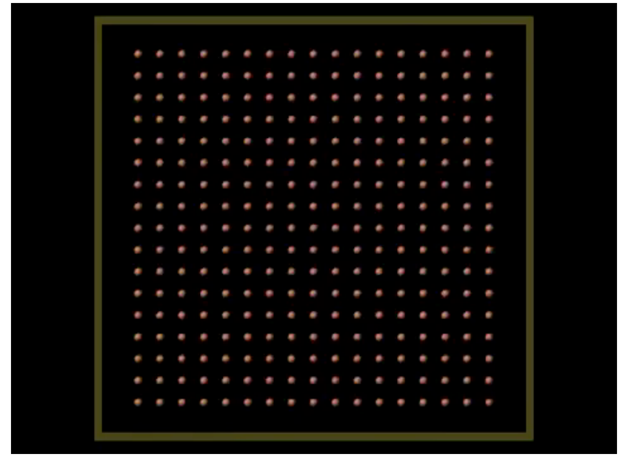
Coarser-level modeling and imaging-based methods

Diffusion and cellular-level simulation

How do molecules move within a cell?



From *Inner Life of the Cell* | *Protein Packing*, XVIVO and Biovisions @ Harvard



https://www.youtube.com/watch?v=1jYabtziQZo

- 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-based 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 derivative of concentration with respect to each spatial coordinate

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

D is the diffusion constant

Reaction-diffusion simulation

- A common method to model cellular processes is *reaction-diffusion simulation*
- Basic rules:
  - Molecules 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 (each particle represents one molecule or complex)
  - Continuum models (based on the diffusion equation)

Recurring Themes

Physics-based 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. knowledge-based protein structure prediction
- Most methods fall somewhere on the continuum between these two extremes
  - Examples: ligand docking or solving x-ray crystal structures
  - Machine learning is playing an ever-greater role in this field, but the best machine learning methods usually incorporate some physics-based information

Energy functions

- Energy functions (approximate representations of 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

Computation is required not only for prediction of structure and dynamics but also for structural interpretation of experimental data

- Computational prediction
  - Biomolecular structure prediction
  - Protein design
  - Molecular dynamics simulation
  - Ligand docking
  - Reaction-diffusion simulations
- Structural interpretation of experimental data
  - X-ray crystallography
  - Cryo-electron microscopy
  - Image analysis for fluorescence microscopy data (including super-resolution imaging)

Recurring math concepts

- **Fourier transforms** and **convolution** play important roles in:
  - Image analysis
  - X-ray crystallography
  - Cryo-electron microscopy
  - Some methods for docking and molecular dynamics simulation
- Another recurring math concept: **Monte Carlo methods** for sampling a probability distribution and for optimization

Similarities and differences in methods employed at different spatial scales

- Atomic-level modeling of single proteins vs. coarser-level modeling of cells
- Experimental methods: x-ray crystallography vs. single-particle electron microscopy
- Simulation methods: molecular dynamics vs. reaction-diffusion simulations

Although I’ve mostly focused on proteins as examples, the concepts and methods we’ve covered apply to other biomolecules as well (RNA, DNA, carbohydrates, small molecules, etc.)

How did people do these things before they had powerful computers?

Large-scale simulation in 1971

- Excerpt from “Protein synthesis: an epic on the cellular level”
- Performed at Stanford!

https://www.youtube.com/watch?v=u9dhO0iCLww

Course evaluations

- Please fill them out, as this helps me continue to improve the course!