CS612 - Algorithms in Bioinformatics

Protein Structure Detection Methods

April 7, 2025

Secondary Structure Prediction

- Assignment of secondary structure is a typical annotation problem that can be addressed with various machine learning techniques
- HMMs can be used to annotate an amino-acid sequence with secondary structure information – HMMs are an example of generative models
- There are other methods that rely on neural networks, SVMs, and other machine learning techniques
- The current state of the art achieves accuracy rates of 70%-80%
- All approaches capture key amino-acid level signals present in alpha-helices and beta-strands
- Since coils, loops, turns do not have such well-defined signals, they are usually predicted as "other" and are more difficult to pin down

Secondary Structure Prediction Assessment

- Most common approach to measure secondary structure prediction accuracy is the Q_3 score:

$$Q_3 = \frac{\text{Number of residues correctly predicted}}{\text{Total number of residues in protein}} * 100$$

- Random prediction that follows the observed frequency of alpha-helices (39%), beta-strands (23%), and coils (38%) will give an average Q_3 accuracy of around 35%
- History of improvements in accuracy:
  - First-generation methods: 50%-56% accuracy
  - Second-generation methods: 70% accuracy
  - State-of-the-art (current) methods: 70%-80% accuracy

Secondary Structure Prediction Methods

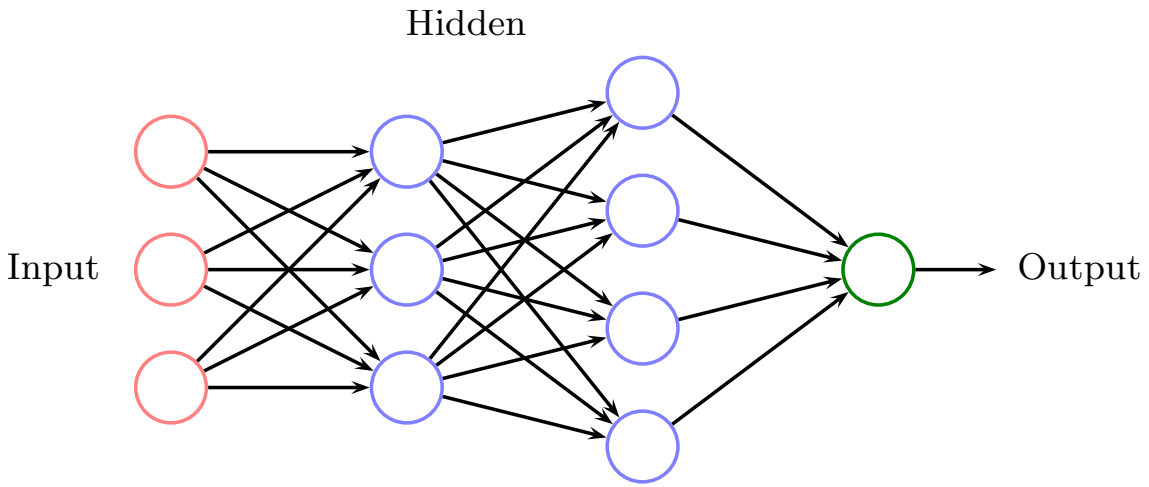
- First-generation (three representative methods) – 50%-56% accuracy:
  1. Rules manually derived from known native structures of proteins Example: Lim et al. with accuracy 50%
  2. Automated statistics on amino-acid composition and neighbor effects Example: Chou-Fasman et al. with accuracy 53%
  3. Statistics on composition taken on 17-residue windows [-8, ?, +8], using a statistical framework to predict the secondary structure of the middle '?' residue
  4. Example: GOR method with accuracy 56%

Secondary Structure Prediction Methods

- Second-generation helped by increase in deposited structures and the use of MSA to detect similar sequences with similar structures (two representative methods) 70% accuracy:
  1. MSA information combined with HMMs, neural networks, SVMs Example: PHD with accuracy 70.8%
  2. k-nearest information (k= 50-100 window) gathered from a database on a voting principle
  3. Example: NNSP with accuracy 72.2%
- Deep learning method using HMM and PSSM profiles – 80% and up.

Artificial Neural Networks

- Artificial Neural Networks (ANN) are computational models inspired by the biological neural networks that constitute animal brains.
- They are used for classification problems and pattern recognition.
- The network is represented as a weighted directed graph. The graph has a layered structure:
- An input layer, one or more "hidden layers" and an output layer.

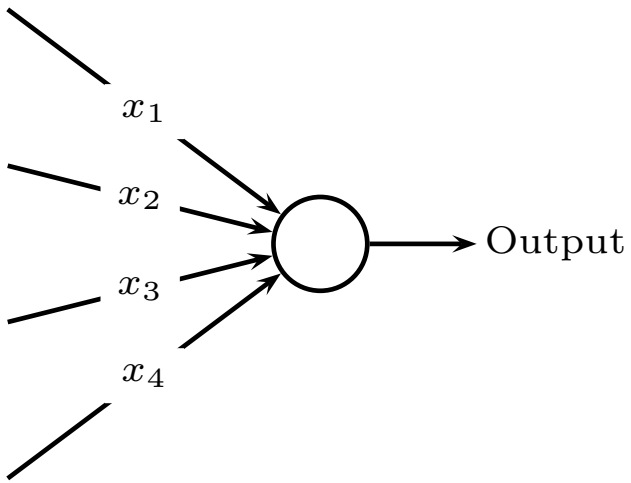


Artificial Neural Networks

- Every node represents a neuron and every edge represents a connection (synapse) between neurons.
- Different layers may perform different functions on their inputs.
- Signals travel from the input layer, to the last output layer, possibly after traversing the layers multiple times.
- The input layer receives input from the outside in the form of a vector. The output layer transmits output to the outside.
- The hidden layers are not connected directly to the outside, only to other layers.
- Each input is multiplied by its edge weight, representing the strength of the interconnection between neurons inside the network.

The Simplest Artificial Neural Network

- Let us first look at what is perhaps the simplest ANN, a *perceptron*.
- A perceptron takes binary inputs, $x_1 \ldots x_n$ and produces a single binary output.
- The inputs are weighted by real numbers: $w_1 \ldots w_n$, scaling the importance of each input to the output.
- Overall, the input is a weighted sum $\sum_{1}^{n} w_i x_i$.



A perceptron

- The output is either 0 or 1, depending on whether $\sum_{1}^{n} w_i x_i$ is below or above a given threshold. respectively:

$$Output = \begin{cases} 0, & \text{if } \sum_{1}^{n} w_i x_i < threshold \\ 1, & \text{otherwise} \end{cases}$$

- This is called a step function. You can think about it as a very simple decision making device, weighing up evidence from the input neurons.

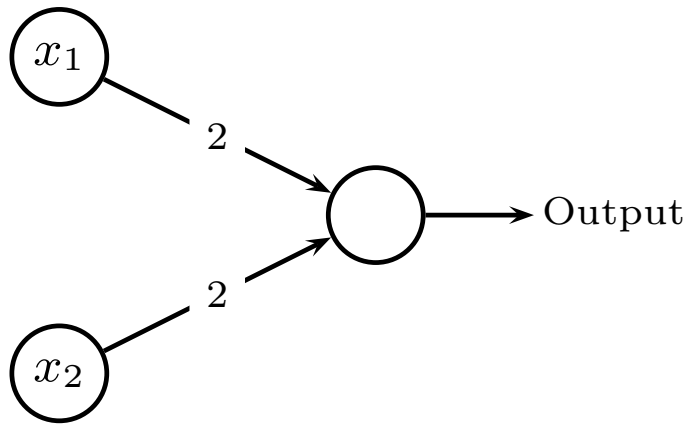
An or perceptron

Here is a simple example, a perceptron with two input neurons that calculates logical OR:

x_1	x_2	Output
0	0	0
0	1	1
1	0	1
1	1	1

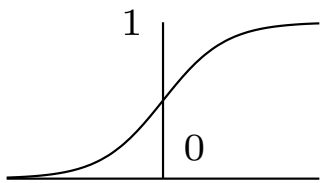
The activation function is:

$$Output = \begin{cases} 0, & \text{if } \sum_1^n w_i x_i < 2 \\ 1, & \text{otherwise} \end{cases}$$



A Perceptron

- The input neurons are binary in this case and the weights are 2.
- It is easy to see that the output is 2 if and only if both inputs are 0, and 1 otherwise.
- The step function from above is often replaced by a sigmoidal function with a smoother threshold: $Output = \frac{1}{1+e^{-x}}$.



Sigmoidal function

- We can use this simple perceptron model to build increasingly complex networks.
- Each of the perceptrons in a given layer makes a decision based on the input from the previous layer.
- This way, a many-layer network of perceptrons can engage in sophisticated decision making.

Types of Neural Networks

Definition

feed forward network A feed forward network is a network where the output is only propagated in one direction: From the input to the output.

- This is the simplest neural network model.
- There are no feedback connections in which outputs of the model are fed back into itself.

Types of Neural Networks

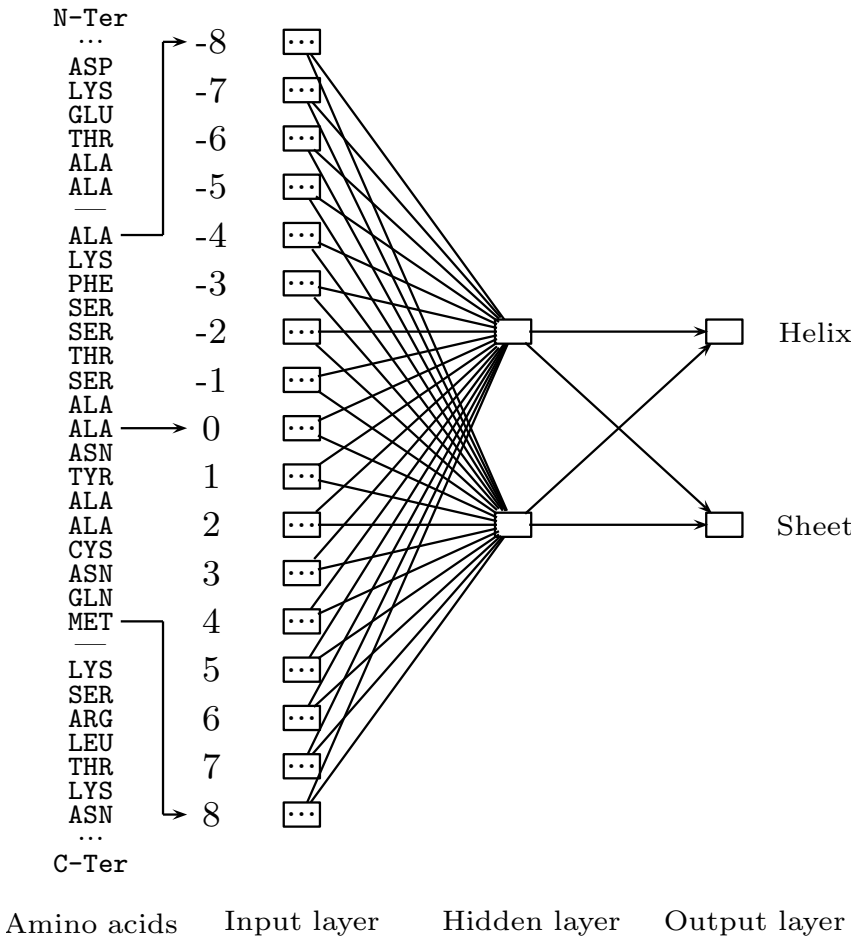
Definition

backpropagation network A backpropagation network is a network where an output can be fed back through the network in order to minimize the output error.

- Arbitrary weights are initially assigned and the output values are compared with the correct answer (target output) to compute the value of some predefined error-function.
- The error is then fed back through the network. Using this information, the algorithm adjusts the weights of each connection in order to reduce the value of the error function by some small amount.
- The process continues for a pre-defined number of rounds or until the output converges to a good enough value.

ANN for Secondary Structure Prediction

- A representative example goes back to 1989.
- The input was a set of 62 proteins. 48 were used as the training set, and the remaining 14 were the test set.
- The network consisted of one input layer, a single hidden layer and an output layer.



ANN for Secondary Structure Prediction

- The input layer was a sliding window of size 17 on the amino acid sequence.
- The prediction is made for the central residue in the window.
- Each amino acid at each window position is encoded by a group of 21 inputs, one for each possible amino acid type and one is a null input when the window overlaps with the N- or C- terminus.
- In each group of 21 inputs, the input corresponding to the amino acid type at that window position is set to 1 and all other inputs are set to 0.
- Thus, the input layer consists of 17 groups of 21 inputs each, and for any given 17 amino acid window, 17 network inputs are set to 1 and the rest are set to 0.

ANN for Secondary Structure Prediction

- The hidden layer consists of two units. The output layer also consists of two units.
- Secondary structure is encoded in these output units as follows: $(1, 0)$ = helix, $(0, 1)$ = sheet, and $(0, 0)$ = coil.
- Actual computed output values are in the range $0.0 - 1.0$ and are converted to predictions with the use of a threshold t.
- Helix is assigned to any group of four or more contiguous residues having helix output values greater than sheet outputs and greater than t.
- β-Strand is assigned to any group of two or more contiguous residues, having sheet output values greater than helix outputs and greater than t.
- Residues not assigned to helices or sheets are assigned to coil.

PSIPRED – A Newer ANN Based Method

- Use PSSM (Position-specific scoring matrices) based on sequence profiles.
- The PSSM is obtained from PSI-BLAST on a custom-made sequence databank and used as input to the neural network.
- The matrix has $20 \times M$ entries (M is the size of the target sequence)
- Each entry is the log-likelihood of this particular substitution in the template.
- Feed-forward network with a single hidden layer.
- A window of 15 amino acids was found to be optimal.

Hidden Markov Models (HMM)

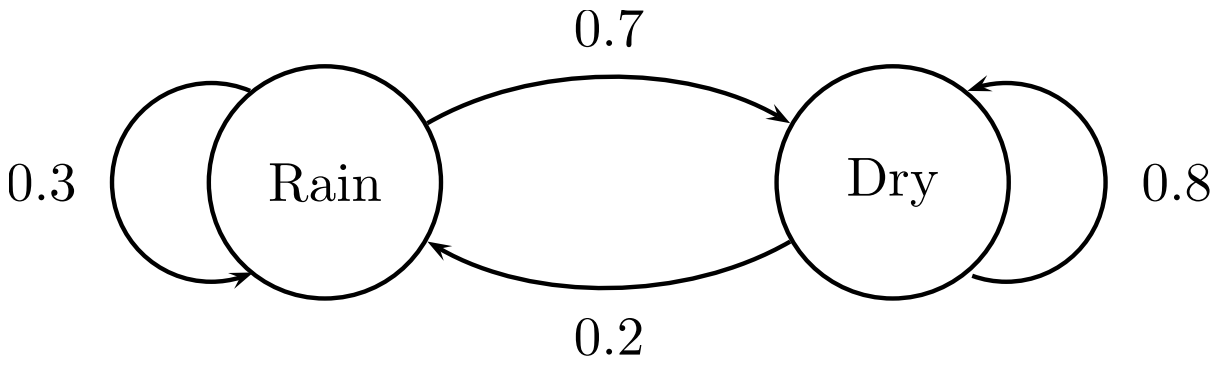
- A statistical model used to model randomly changing systems.
- It assumes that future states of the system depend only on the current one, and not any past ones.
- A `Markov Chain` is a model that describes the system using a random variable that changes over time, and its distribution depends only on the state preceding it.
- More formally, we are given the following:
  - A set of states $\{S_1, S_2, ..., S_N\}$
  - A set of transition probabilities $a_{ij} = P(S_i|S_j)$
  - A set of initial probabilities $\pi_i = P(S_i)$ for every i
- In addition, we assume that every state depends only on the previous state: $P(S_{ik}|S_{i1}, S_{i2}, ..., S_{ik-1}) = P(S_{ik}|S_{ik-1})$

Hidden Markov Models Example

- Weather prediction:
- Given two states, Rain and Dry, with the following transition probabilities:

From \ To	Rain	Dry
Rain	0.3	0.7
Dry	0.2	0.8

- Plus the following transitional probabilities: $P(Rain) = 0.4$ and $P(Dry) = 0.6$.



Markov Models Example

- Let us calculate the probability of a state sequence as follows:
  $P(S_{i1}, S_{i2}, ..., S_{ik-1}, S_{ik}) =$
  $P(S_{ik}|S_{i1}, S_{i2}, ..., Sik-1)P(S_{i1}, S_{i2}, ..., S_{ik-1})$.
- The Markov chain property states that the above is equal to:
  $P(S_{ik}|S_{ik-1})P(S_{i1}, S_{i2}, ..., S_{ik-1}) =$
  $P(S_{ik}|S_{ik-1})P(S_{ik-1}|S_{ik-2})...P(S_{i2}|S_{i2})P(S_{i1})$.
- The sequence $\{Dry, Rain, Rain, Dry\}$ corresponds to:
  $P(\{Dry, Rain, Rain, Dry\}) =$
  $P(Dry|Rain)P(Rain|Rain)P(Rain|Dry)P(Dry) =$
  $0.7 * 0.3 * 0.2 * 0.6 = 0.0252$.

Hidden Markov Models

- A *Hidden Markov Model* (HMM) is a Markov model the rules that produce the Markov chains are not known or "hidden".
- The rules include the probability for a certain observation for a certain state transition, given the state of the model at a certain time.
- We have the following properties, just like before:
  - A set of states $\{S_1, S_2, ..., S_N\}$
  - A set of transition probabilities $a_{ij} = P(S_i|S_j)$
  - A set of initial probabilities $\pi_i = P(S_i)$ for every i
- However, states are not visible, so in addition we get the following:
  - Each state randomly generates one of M observations (or visible states) $\{V_1, V_2, ..., V_M\}$
  - A set of observation probabilities: $b_i(V_m) = P(V_m|S_i)$. These are also called *emissions*.

Hidden Markov Models

The (HMM) method aims to solve the following problems:

1. given the model, find the probability of the observations.
2. given the model and the observations, find the most likely state transition trajectory.
3. maximize either 1 or 2 by adjusting the model's parameters.

- Say we now have two *observations*, {Rain, Dry} and two (invisible) *states*, {Low, High} (atmospheric pressure).
- In other words, we can observe whether it is rainy or dry, but we cannot directly tell what the atmospheric pressure is.
- The only way to recover the most likely atmospheric pressure is through the observations and the set of probabilities.
- We assume that the data observed is not the actual state of the model, but is instead generated by the underlying hidden states.

Hidden Markov Models

The transition probabilities are:

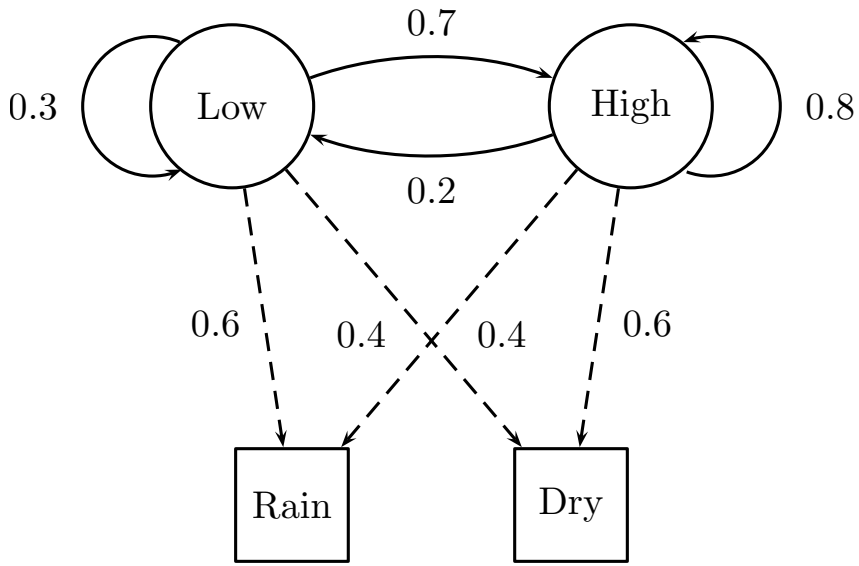
From \ To	Low	High
Low	0.3	0.7
High	0.2	0.8

The observation probabilities are:

From \ To	Rain	Dry
Low	0.6	0.4
High	0.4	0.6

The initial probabilities are:
$P(Low) = 0.4$ and
$P(High) = 0.6$.

The graphical representation of the HMM is as follows:



Hidden Markov Models

- Given a sequence of observations, say {Dry, Rain}, and want to calculate its probability, we account for all the possible hidden state sequences:

  P({Dry,Rain}) = P({Dry,Rain} , {Low,Low}) + P({Dry,Rain} , {Low,High}) + P({Dry,Rain} , {High,Low}) + P({Dry,Rain} , {High,High})

- Every term can be calculated given the transition probabilities and the Markov chain properties:

  $P(\{Dry, Rain\}, \{Low, Low\}) = P(\{Dry, Rain\}|\{Low, Low\})P(\{Low, Low\})$

- Now, based on the Markov chain property for the hidden states, the last term can be expressed as:

  $P(\{Low, Low\}) = P(Low|Low)P(Low)$

- Additionally, the value of the observed variable depends only on the value of the hidden state at that time. Therefore, the entire formula can be expressed as:

  $P(\{Dry, Rain\}, \{Low, Low\}) =$
  $P(Dry|Low)P(Rain|Low)P(Low)P(Low|Low) =$
  $0.4 * 0.4 * 0.6 * 0.4 * 0.3 = 0.01152$

- The other probabilities can be calculated in a similar manner.

Using HMM for Secondary Structure Prediction

- HMM is an example of a *generative model*: we learn a model based on specific signals and see how well the model explains (classifies/annotates) the input queries.
- In secondary structure prediction the observations are the amino acids and the hidden process is the secondary structure.
- The assumption is that secondary structure can be modeled by Markov chain and the observed states (amino acids) are independent of each other, conditionally to the hidden process (the secondary structure composition).
- The simplest HMM models every secondary structure by a single state.
- The parameters are the transition and emission probabilities.
- More complex models assign several hidden states per secondary structure.
- Model parameters are then estimated from available data.

OSS-HMM (Optimal Secondary Structure prediction HMM)

- Calculates secondary structure elements with 75.5% accuracy.
- Let n_H, n_b and n_c be the number of hidden states that model α-helices, β-strands and coils, respectively.
- The optimal model selection is done in three steps:
  1. $n_H = n_b = n_c = n$, estimate models with n running from 1 to 75. Eventually, $n = 14$ was selected.
  2. Models were thus estimated with:
     1. $n_H = 1$ to 20 and $n_b = n_c = 1$,
     2. $n_b = 1$ to 15 and $n_H = n_c = 1$
     3. $n_c = 1$ to 15 and $n_H = n_b = 1$.

     $n_H = 15$, $n_b = 8$ $n_c = 9$ were selected
  3. Optimal model was selected as having 36 states with $n_H = 15$, $n_b = 9$ and $n_c = 12$.

Model Estimation:

Three criteria are used for the selection of the optimal model:

1. Q_3 as described above
2. The Bayesian Information Criterion (BIC) is defined as: $BIC = \log L - 0.5 \times k \times \log(N)$, where $\log L$ is the log-likelihood of the learning data under the trained model, k is the number of independent model parameters and N is the size of the training set.
3. The statistical distance between two models. The distance D_s between models M_1 and M_2 is given by: $D_s(M_1, M_2) = \frac{D(M_1,M_2)+D(M_2,M_1)}{2}$, and $D(M_1, M_2) = \frac{1}{T}|\log L(O^{(2)}|M_1) - \log L(O^{(2)}|M_2)$, where $O^{(2)}$ is a sequence of length T generated by model M_2 and $\log L(O^{(2)}|M_i)$ is the log-likelihood of $O^{(2)}$ under model M_i.

Some Details

- The selection process first considers models with equal numbers of hidden states.
- Then, models are considered where the number of states are set to one for two of the secondary structure classes, and increase for the remaining classes.
- This defines the model size range that needs to be explored for each structural class: 12–16 states for helices, 6–10 for strands and 5–13 for coil.
- All transitions between hidden states are initially allowed.
- However, many transitions in the final model are estimated to have probability zero.
- In fact, only 36% of potential transitions remain within the helix box, 57% within the strand box and 68% in the coil box.
- The final model has 448 non-null transitions (out of the possible $36^2 = 1296$), of which 89 have a probability greater than 0.1, for a total of 1096 free parameters.

Tertiary Structure Prediction – Comparative Modeling

- Comparative modeling is modeling of the unknown based on comparison to what is known
- In the context of modeling or computing the structure s_x assumed by a sequence x of amino acids:
- Structure is a function of sequence: So, $s_x = f(x)$?
- The function f encodes how the sequence x determines the structure s_x
- Given another protein of sequence y and known structure s_y, we can infer: IF $x \approx y$ THEN $s_x \approx s_y$
- It is important that x and y be similar enough
- An important question: how similar?

Comparative Modeling – Some Terminology

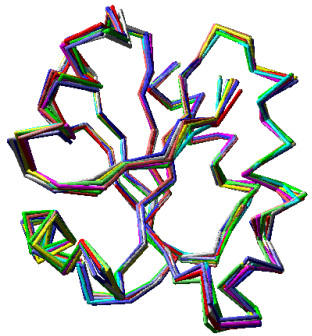
- The protein of unknown structure is the query or the target
- The protein of known structure whose sequence is similar to that of the target is the template
- The process of inferring the coordinates for the target is called model building
- Comparative modeling builds the model, completes it, refines it, and then evaluates it

Why Use Comparative Modeling?

- Structures of proteins in a given functional family are more conserved than their sequences
- About a third of all sequences assume known structures
- The number of unique protein folds is limited
- If not applicable to yield a high-resolution structure, comparative modeling can at least yield the fold for a sequence
- Currently, comparative modeling is both faster and more accurate (as long as the sequence identity is high) than ab initio or de novo methods for structure prediction

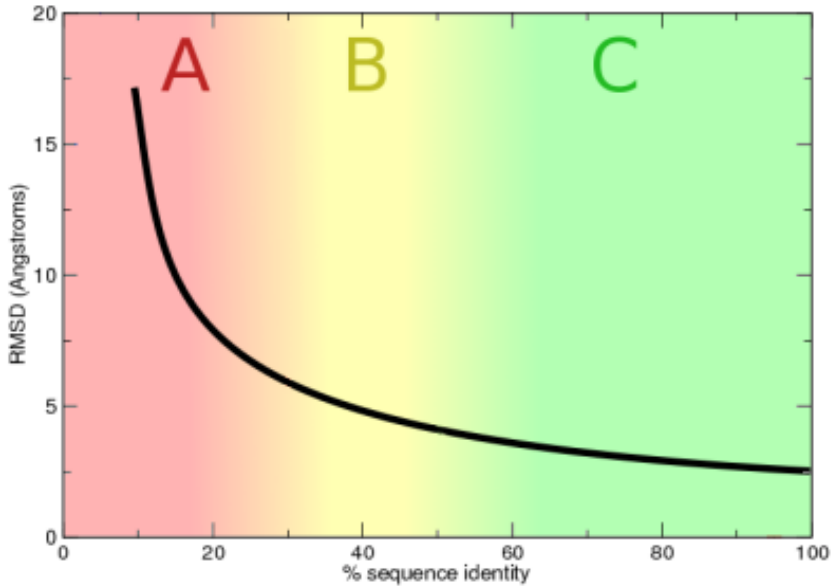
When to Use Comparative Modeling?

- How similar do x and y have to be to infer that the structure assumed by the sequence x is similar to that assumed by the sequence y?
- Statistical analysis of sequences with known structure reveals:
- Sequences with no less than 50% sequence identity assume very similar structures
- Minimum sequence identity for structural similarity: 25-30%

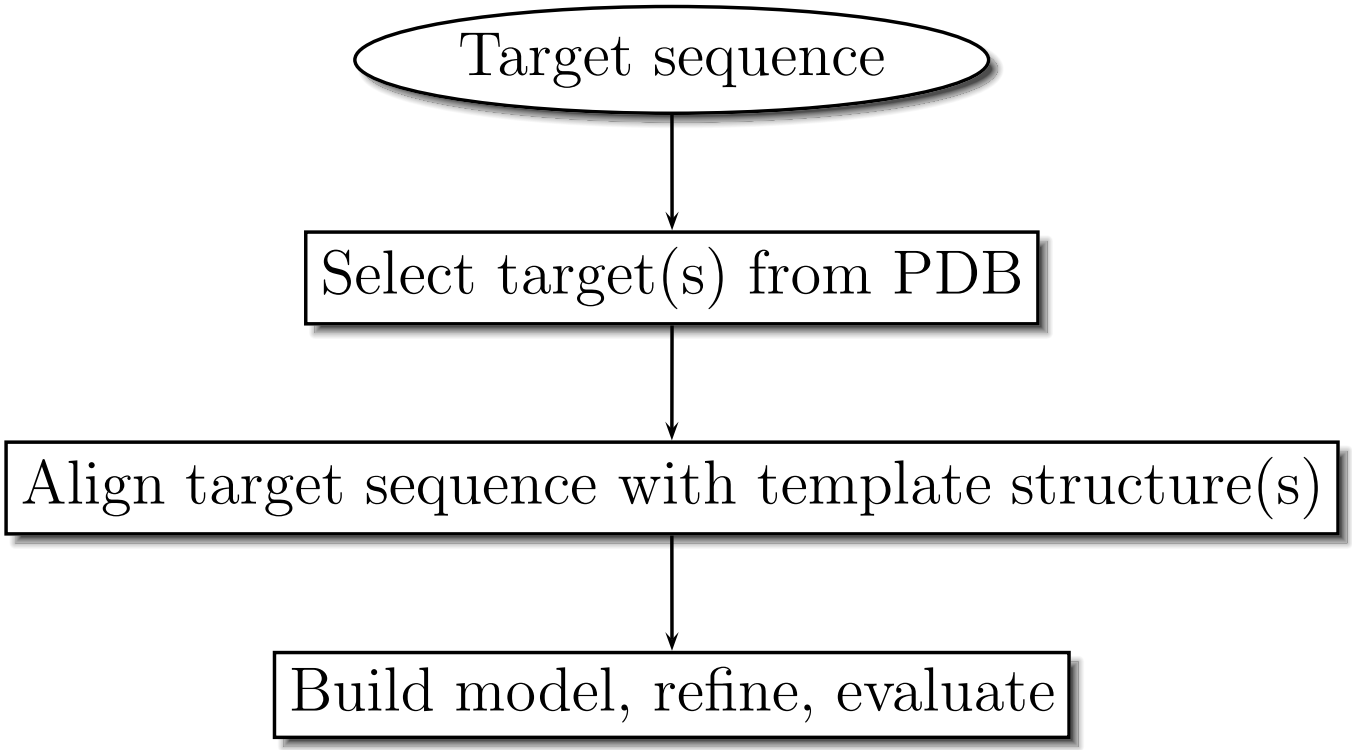


Higher than 30% sequence identity often results in very similar structures

Sequence-structure Relationship



A Simplistic View of Comparative Modeling



Alignment is the most critical step. comparative modeling cannot recover from a bad alignment.

Basic Steps of Homology Modeling

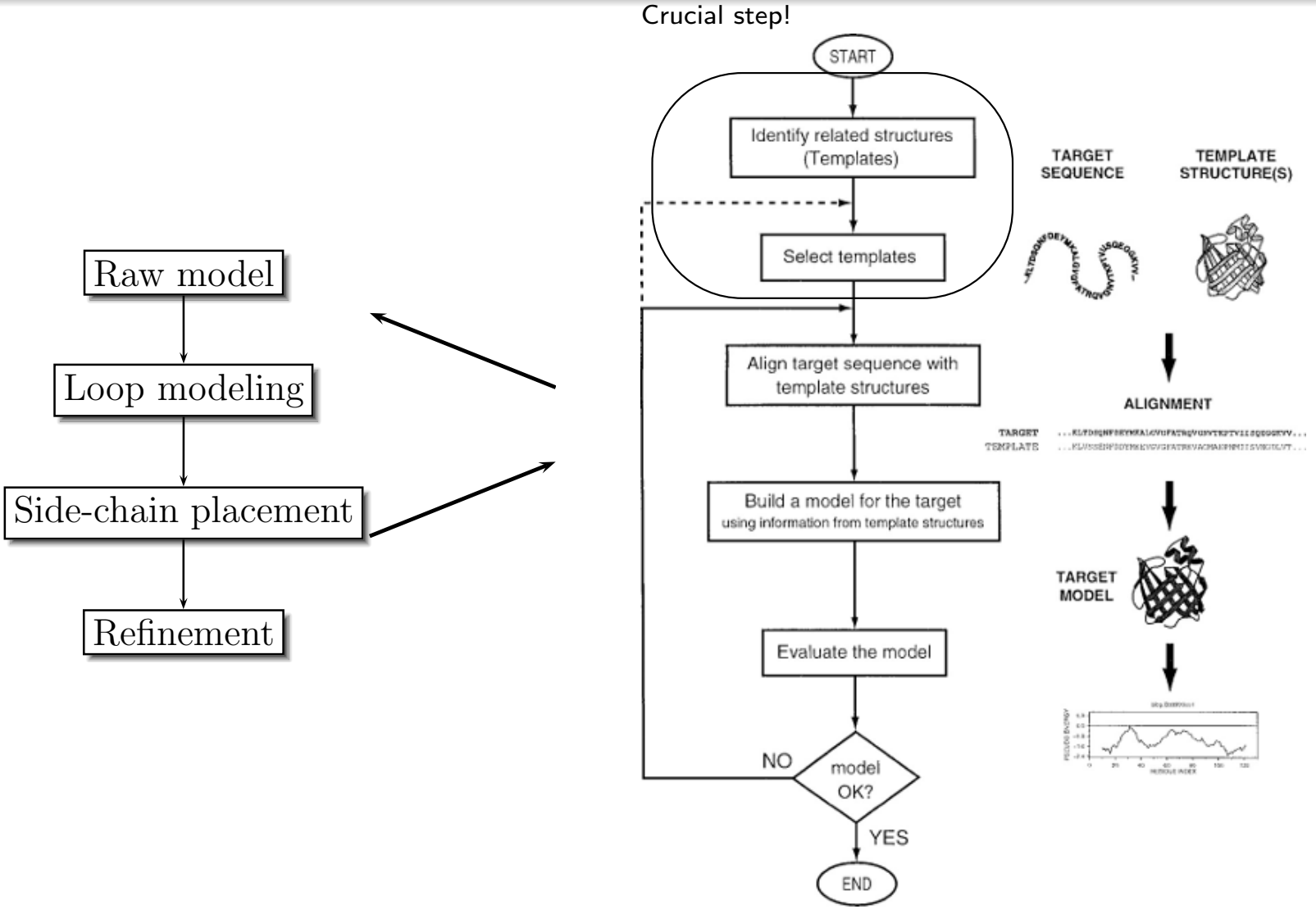


Figure 5.1.1 from M. A. Marti-Renom and A. Sali "Modeling Protein Structure from Its Sequence" Current Prototocols in Bioinformatics (2003). 5.1.1-5.1.32

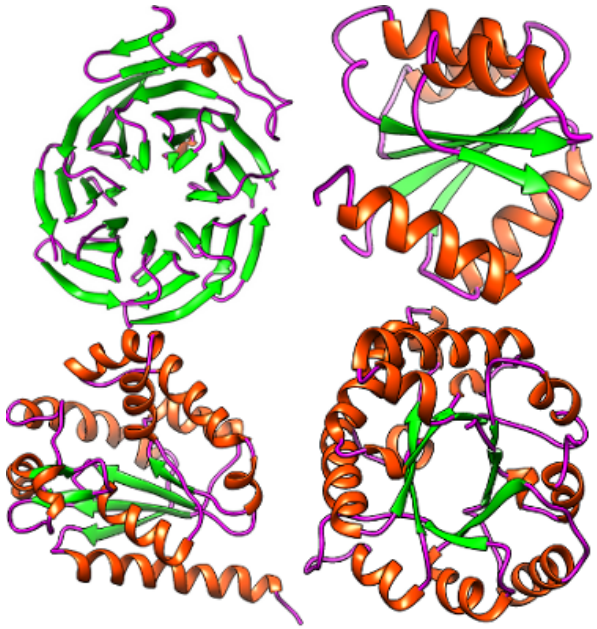
Basic Steps of Homology Modeling

1. Query a database of protein sequences with known structures with the target sequence, focusing on those with $\geq 30\%$ seq. identity to the target sequence
2. Align obtained sequences to target to choose templates
3. Identify structurally conserved (SC) and variable (SV) regions
4. Generate coordinates for the core region of the target
5. Complete the structure of the target
   - generate coordinates for loop regions
   - generate coordinates for side-chains
6. Refine the completed structure using energy minimization
7. Validate/evaluate completed structure

Step 1 – Query PDB

PRTEINSEQENCEPRTEINSEQUENC
EPRTEINSEQNCEQWERYTRASDFHG
TREWQIYPASDFGHKLMCNASQERWW
PRETWQLKHGFDSADAMNCVCNQWER
GFDHSDASFWERQWK

Query Sequence



PDB

Step 1 – Query PDB

PRTEINSEQENCEPRTEINSEQUENC
EPRTEINSEQNCEQWERYTRASDFHG
TREWQIYPASDFGHKLMCNASQERWW
PRETWQLKHGFDSADAMNCVCNQWER
GFDHSDASFWERQWK

Query Sequence

PRTEINSEQENCEPRTEINSEQUENC
EPRTEINSEQNCEQWERYTRASDFHG
TREWQIYPASDFGHKLMCNASQERWW
PRETWQLKHGFDSADAMNCVCNQWER
GFDHSDASFWERQWK

PRTEINSEQENCEPRTEINSEQUENC
EPRTEINSEQQWEWEWQWEWEQWEWE
WQRYEYEWQWNCEQWERYTRASDFHG
TREWQIYPASDWERWEREWRFDSFG

PRTEINSEQENCEPRTEINSEQUENC
EPRTEINSEQNCEQWERYTRASDFHG
TREWQIYPASDFGHKLMCNASQERWW
PRETWQLKHGFDSADAMNCVCNQWER
GFDHSDASFWERQWK

Hit #1

PRTEINSEQENCEPRTEINSEQUENC
EPRTEINSEQQWEWEWQWEWEQWEWE
WQRYEYEWQWNCEQWERYTRASDFHG
TR

PRTEINSEQENCEPRTEINSEQUENC
EPRTEINSEQNCEQWERYTRASDFHG
TREWQIYPASDFG

Hit #2

PRTEINSEQENCEPRTEINSEQUENC
EPRTEINSEQNCEQWERYTRASDFHG
TREWQIYPASDFGPRTEINSEQENCE
PRTEINSEQUENCEPRTEINSEQNCE
QWERYTRASDFHGTREWQIYPASDFG
TREWQIYPASDFGPRTEINSEQENCE
PRTEINSEQUENCEPRTEINSEQNCE
QWERYTRASDFHGTREWQ

PRTEINSEQENCEPRTEINSEQUENC
EPRTEINSEQNCEQWERYTRASDFHG
TREWQIYPASDFG

PRTEINSEQENCEPRTEINSEQUENC
EPRTEINSEQNCEQWERYTRASDFHG
TREWQIYPASDFGPRTEINSEQENC

PDB

Step 2 – Alignment

	G	E	N	E	T	I	C	S
G	10	0	0	0	0	0	0	0
E	0	10	0	10	0	0	0	0
N	0	0	10	0	0	0	0	0
E	0	0	0	10	0	0	0	0
S	0	0	0	0	0	0	0	10
I	0	0	0	0	0	10	0	0
S	0	0	0	0	0	0	0	10

	G	E	N	E	T	I	C	S
G	**60**	40	30	20	20	0	10	0
E	40	**50**	30	30	20	0	10	0
N	30	20	**40**	0	0	0	10	0
E	20	20	20	**30**	20	10	10	0
S	20	20	20	20	**20**	0	10	10
I	10	10	10	10	10	**20**	10	0
S	0	0	0	0	0	0	0	**10**

Dynamic programming

Step 2 – Alignment

Goal: Find a template or templates

pairwise sequence alignment
finds high homology sequences BLAST
http://www.ncbi.nlm.nih.gov/BLAST/

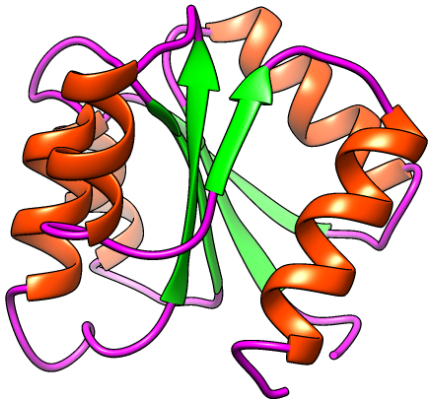
Improved multiple sequence alignment methods
improves sensitivity – remote homologs
PSI-BLAST, CLUSTAL

Step 2 – Alignment

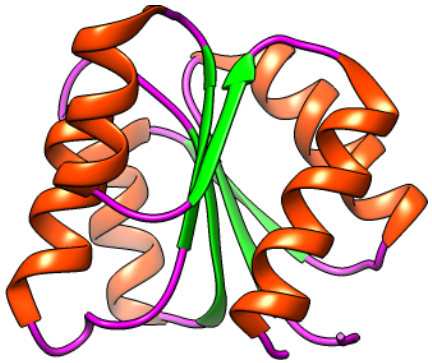
- Pairwise sequence alignment: BLAST, FASTA, WU-BLAST, SSEARCH, and more
- Available as web servers and standalone software
- Basic functionality needed: compare target sequence with sequences in the PDB (or any other comprehensive structural database)?
- BLAST scans the sequence for 3-letter words (wmers, where w = 3) and expands alignments from 3-mers
- Statistically significant alignments are hits
- Templates are hits with no lower than 30% sequence identity

Step 2 – Alignment

```
Query     ACDEFGHIKLMNPQRST--FGHQWERT-----TYREWYEG
Hit #1    ASDEYAHLRILDPQRSTVAYAYE--KSFAPPGSFKWEYEA
Hit #2    MCDEYAHIRLMNPERSTVAGGHQWERT----GSFKEWYAA
```



Hit #1



Hit #2

Step 2 – Alignment

- Global (Needleman-Wunsch) alignment can be used
- Alignment is the most crucial step, as comparative modeling can never recover from a bad alignment
- A small error in the alignment can translate to a significant error in the reconstructed model
- Multiple sequence alignments (that also align the templates to one another) is often better than pairwise alignment

Step 2 – Alignment

- A good template is closest to the target in terms of subfamilies
- This means that high overall sequence similarity is needed
- The template environment like pH, ligands, etc., should be the same as that of the target
- The quality of the experimentally-available template structure - the resolution, R-factor, etc. - should be high
- When choosing a template for a protein-ligand model, it is preferred that the template have the same ligand
- When modeling an active site – a high resolution template structure with ligand is important

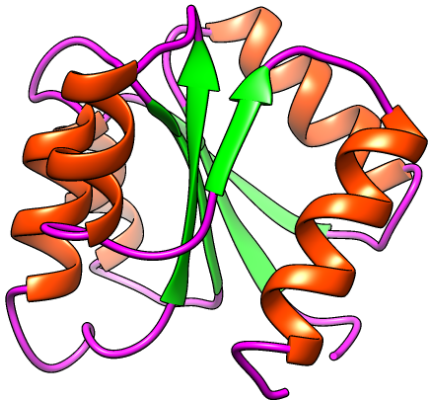
Step 2 – Get Template

- A good template is closest to the target in terms of subfamilies
- This means that high overall sequence similarity is needed
- The template environment like pH, ligands, etc., should be the same as that of the target
- The quality of the experimentally-available template structure - the resolution, R-factor, etc. - should be high
- When choosing a template for a protein-ligand model, it is preferred that the template have the same ligand
- When modeling an active site – a high resolution template structure with ligand is important

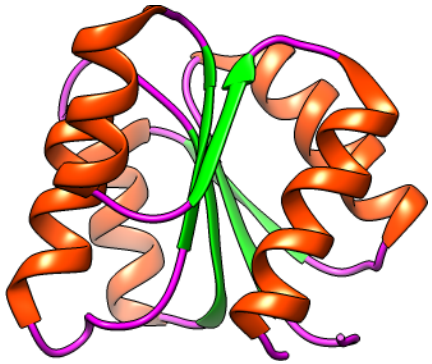
Step 3 – Detect Structurally Conserved Regions (SCRs)

```
Query   ACDEFGHIKLMNPQRST--FGHQWERT----TYREWYEG
Hit #1  ASDEYAHLRILDPQRSTVAYAYE--KSFAPPGSFKWEYEA
Hit #2  MCDEYAHIRLMNPERSTVAGGHQWERT----GSFKEWYAA
        HHHHHHHHHHHHHCCCCCCCCCCCCCCCCCCBBBBBBBBB
```

SCR #1 **SCR #2**



Hit #1



Hit #2

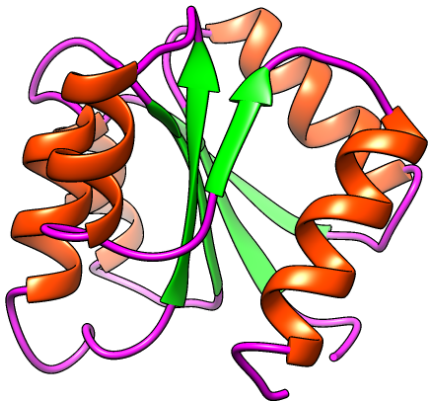
Step 3 – Detect SCRs

- SCRs correspond to the most stable structures or regions (usually in the interior/core) of the protein
- SCRs also often correspond to sequence regions with the lowest level of gapping and highest level of sequence conservation
- SCRs are often the secondary structures

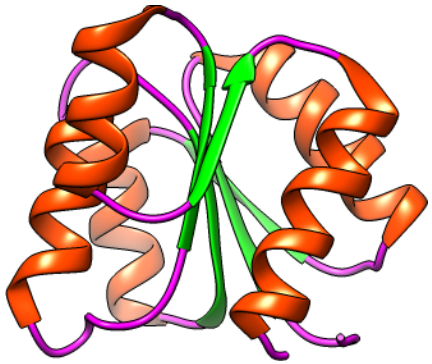
Step 3 – Detect Structurally Variable Regions (SVRs)

```
Query     ACDEFGHIKLMNPQRST--FGHQWERT-----TYREWYEG
Hit #1    ASDEYAHLRILDPQRSTVAYAYE--KSFAPPGSFKWEYEA
Hit #2    MCDEYAHIRLMNPERSTVAGGHQWERT----GSFKEWYAA
          HHHHHHHHHHHHHCCCCCCCCCCCCCCCCCBBBBBBBBB
```

SVR (loop)



Hit #1

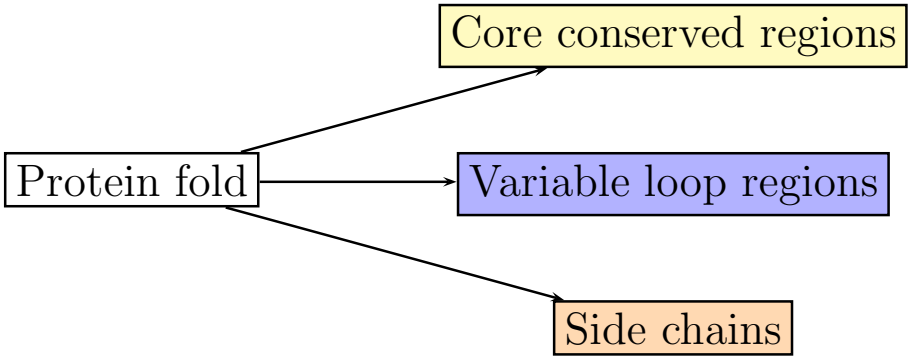


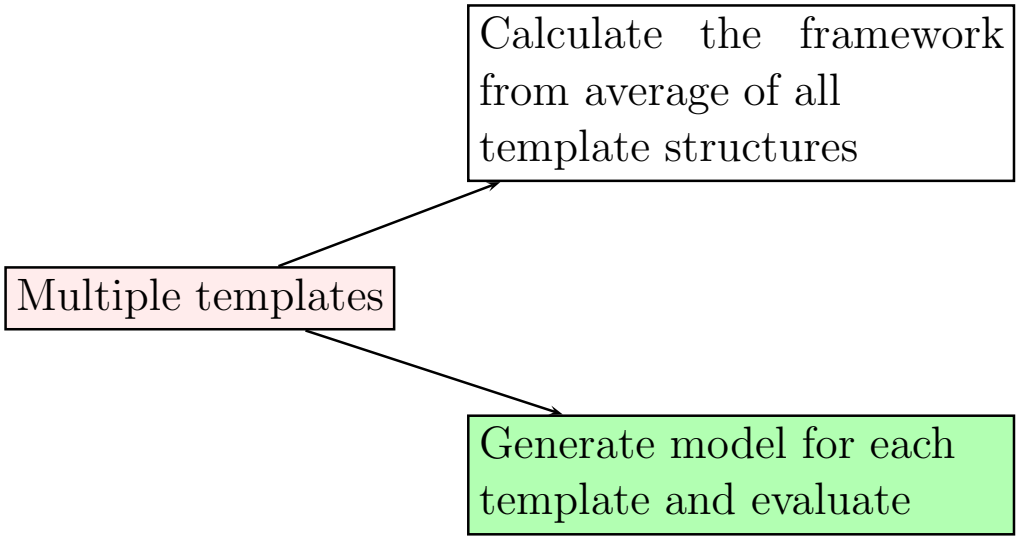
Hit #2

Step 3 – Detect SVRs

- SVRs correspond to the least stable or the most flexible regions (usually in the exterior/surface) of the protein
- SVRs correspond to sequence regions with the highest level of gapping and lowest level of sequence conservation
- SVRs are usually loops and turns

Step 4 – Threading





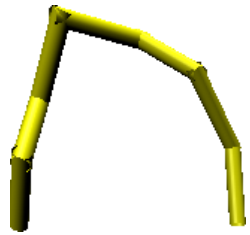
Step 4 – Threading

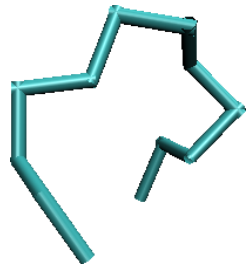
- For identical amino acids, just transfer all atom coordinates (x, y, z) to the query protein (both backbone and side-chain atoms are identical)?
- For similar amino acids, transfer the backbone coordinates and replace side-chain atoms while respecting χ angles
- For different amino acids, one can only transfer the backbone coordinates (x, y, z) to query sequence
- The side chains of different amino acids have to be built at a later stage, when completing the model

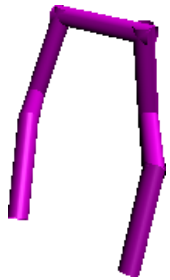
Step 5 – Loop Modeling

Query FGHQWERT
Hit #1 YAYE--KS

Look up a fragment database







Step 5 – Loop Modeling

Loops result from substitutions and ins/dels in same family

Mini protein folding problem – loops can be very long in membrane proteins

Ab-initio methods generate various random loop conformations and evaluate/score

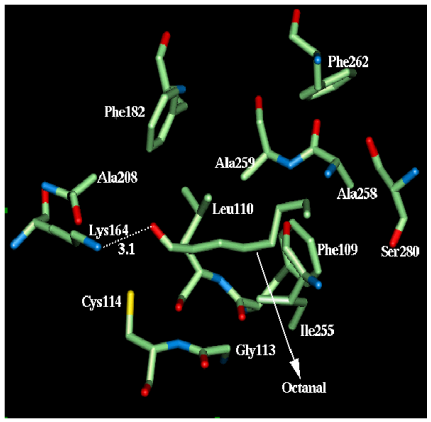
Compare the loop sequence string to PDB, get hits, and evaluate/score

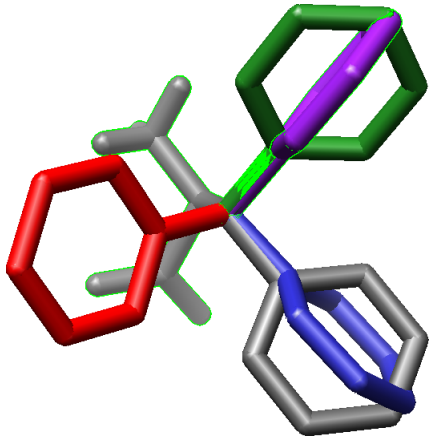
Some comparative modeling methods have fewer loops to be added because of extensive multiple sequence alignment of profiles

Step 5 – Loop Modeling

- Ab-initio loop modeling – Monte Carlo, Monte Carlo with simulated annealing, MD, main chain dihedral angle search biased with the data from PDB, inverse kinematics-based, etc.
- Energy functions used: physics-based (CHARMM, AMBER, etc.) or knowledge-based (built with statistics obtained from PDB)?
- Ab-initio methods – allow simultaneous addition of several loops, which yields a conformational ensemble view for the loop

Step 5 – Side Chain Modeling





Side-chain packing is critical to studying ligand binding to proteins

Side chain builder

- Predicted from similar structures
- Built from steric and energetic considerations with robust conformational search algorithms
- Combination of rotamer library and energy evaluations

Step 5 – Side Chain Modeling

- Rotamer libraries have been created (statistical analysis of torsion angles of side chains of amino acids) from structures in the PDB
- Two main effects in predicting side chains
  - How it sits on top of the main chain(very critical)?
  - Continuous variation of side chain torsions - only 6% varies +/- 40 degrees from the rotamer libraries
- Current techniques predict side chains up to 1.5Å accuracy for a fixed backbone for the core residues
- Solvation and H-bond terms are very important in modeling exposed side chains

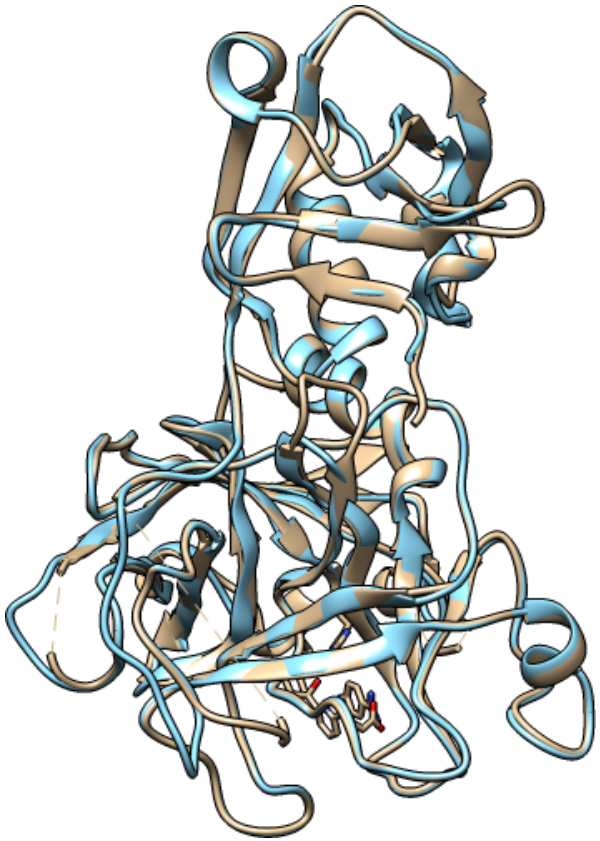
Step 5 – Side Chain Modeling

- Methods available – SCWRL, SCAP, MODELLER, Insight II, WhatIf, SCREAM etc.
- Evaluation of all three methods for backbone < 4Å IRMSD to native all work equally – 50% of $\chi 1$ and 35% of $\chi 2$ and $\chi 3$
- SCWRL – Decomposition of protein to non-interacting parts, collision free energy function. Fast, works quite well
- SCREAM – works well – accurate energy analysis – computationally intensive

Step 6 – Refinement

- Completed model may undergo a short energy minimization
- Physics-based or knowledge-based functions may be used
- The minimization may help remove steric clashes and improve favorable interactions in the completed model prior to the final evaluation of the built model for the target

Comparative Modeling – Example



Beige – template, pdb:5ce1
Blue – model, created by Swiss-Model. Sequence identity = 40%

Evaluation of Model

Given a predicted structure:

- Ramachandran plot – allowed regions for backbone torsions
- Calculate the Hydrogen-bond network – use Quanta or WhatIf or MolProbity – normally calculated for heteroatoms with distance cutoff
- Identify hydrophobic residues on the surface
- Identify hydrophilic residues in the core – satisfied with salt bridges?
- Voids in the core are typically small two water cluster?

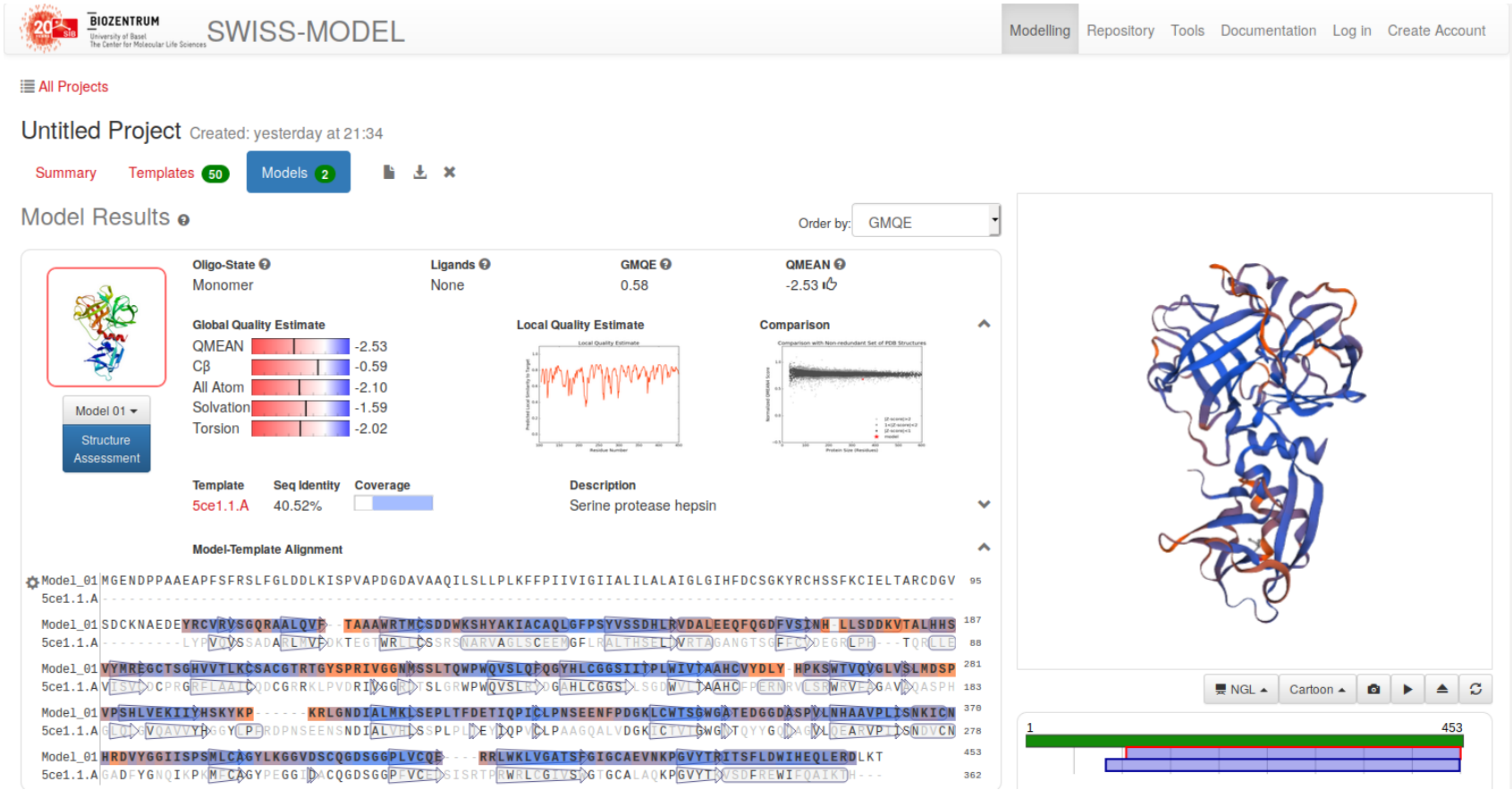
The Swiss-Model Pipeline

1. **Input Data:** FASTA sequence or UniProtKB ID.
2. **Template Search:** Sequences are searched against a template library called STML using either BLAST or HHblits.
3. **Template Selection:** Multiple templates are selected and ranked according to scoring functions (more on that later)
4. **Model Building:** The modeling engines are called ProMod3 and OpenStructure. More details in notes.
5. **Refinement** Energy minimization to resolve small clashes is performed using the CHARMM27 force field.
6. **Quaity Estimation:** See next slide...

Evaluation of Model By Swiss-Model

- GMQE (Global Model Quality Estimation) is a quality estimation which combines properties from the alignment and the template search method.
- The score is a number between 0 and 1, reflecting the expected accuracy of a model built with that alignment and template and the coverage of the target.
- Higher numbers indicate higher reliability.
- QMean score is a composite estimator based on different geometrical properties and provides both global (i.e. for the entire structure) and local (i.e. per residue) absolute quality estimates on the basis of one single model.

Evaluation of Model By Swiss-Model



Comparative Modeling on the WWW

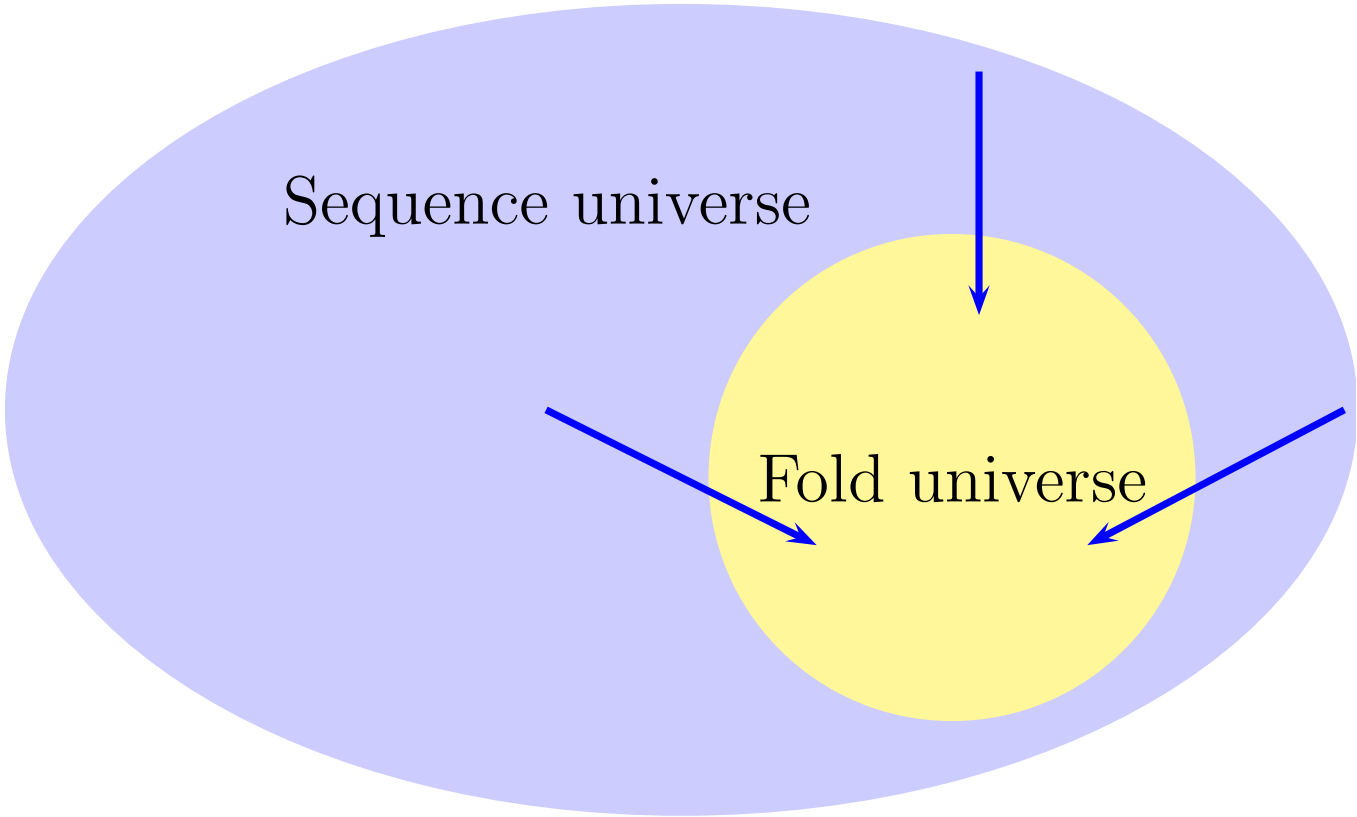
- Prior to 1998, comparative modeling could only be done with commercial software or command-line freeware
- The process was time-consuming and labor-intensive
- The past few years has seen an explosion in automated web-based comparative modeling servers
- Now anyone can! (but you still have to know what you're doing...)

What are Folds Anyway?

- **Family:** clear evolutionary relationship
  - Sequence identity $\geq 30\%$, but similar functions and structures indicate common descent even on low sequence identity
  - Globins family has members with sequence identities of only 15%
- **Superfamily:** probable common evolutionary origin
  - Low sequence identities, but structural and functional features suggest a common evolutionary origin
  - Actin, ATPase domain of heat shock protein, and hexakinase together form a superfamily
- **Fold :** major structural similarity (possibly no common ancestor)
  - Proteins have a common fold if they have the same major secondary structures in the same arrangement and with the same topological connections, though length of regions can change
  - Structural similarities can arise just from the physics and chemistry of proteins favoring certain packing arrangements and chain topologies

Many Sequences Map to Limited Folds

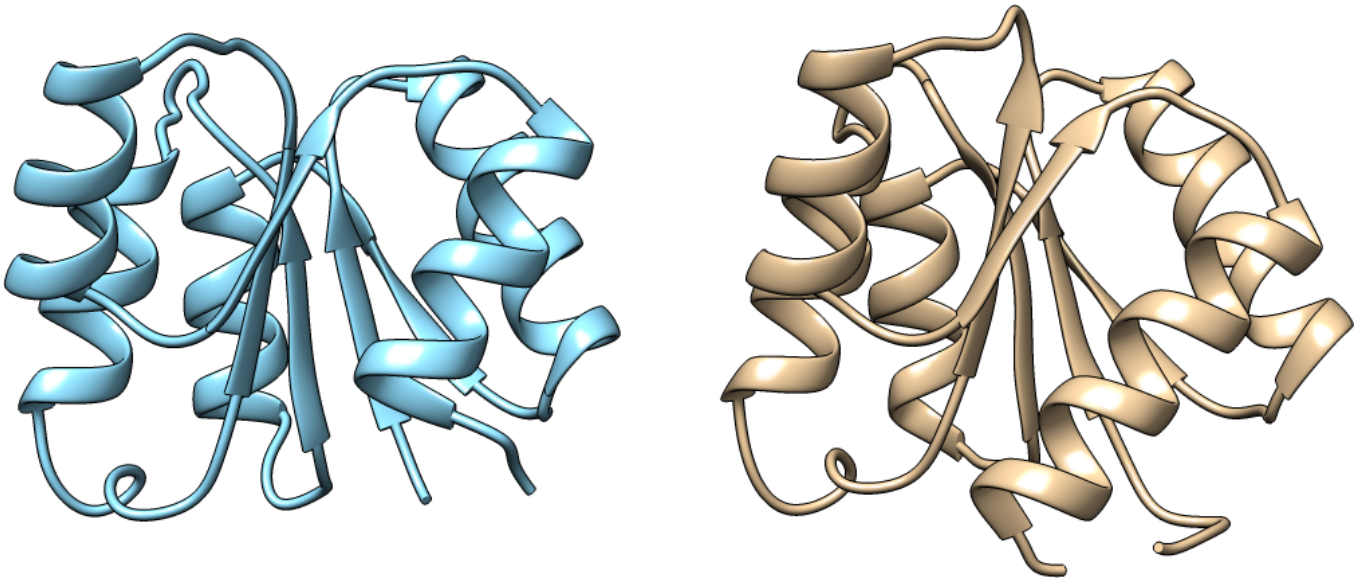
- A large number of protein sequences adopt similar folds
- Even when sequence identity is lower than 25%
- Such sequences are known as remote homologues



Threading: Towards Ab-initio Methods

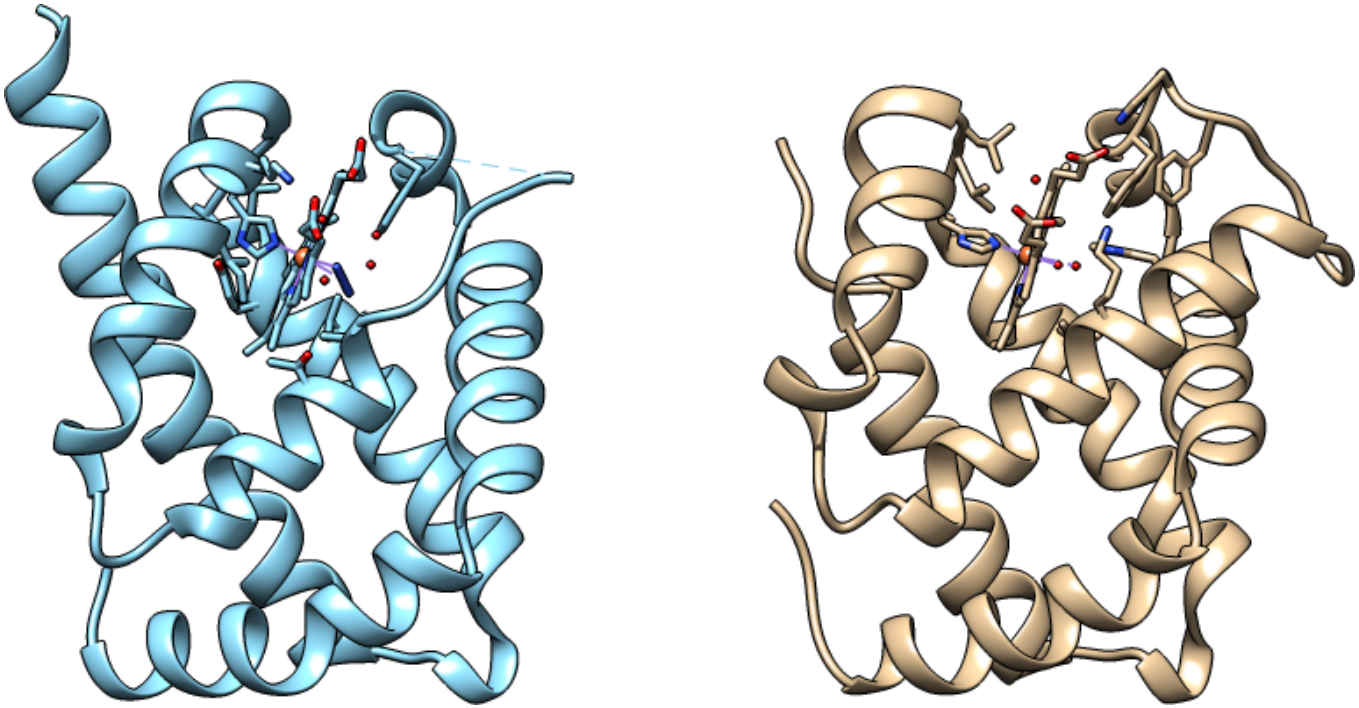
- Remote homologues necessitate a novel computational approach
- Between comparative modeling and ab initio: Threading
- Threading may detect structural similarities that are not accompanied by detectable or significant sequence similarities
- The environment of a particular amino acid is expected to be more conserved than the actual sequence identity of the amino acid

When to Resort to Threading?



A pair of structures within the same family that have sequence identity of 47%: Probably comparative modeling can be applied here

When to Resort to Threading?

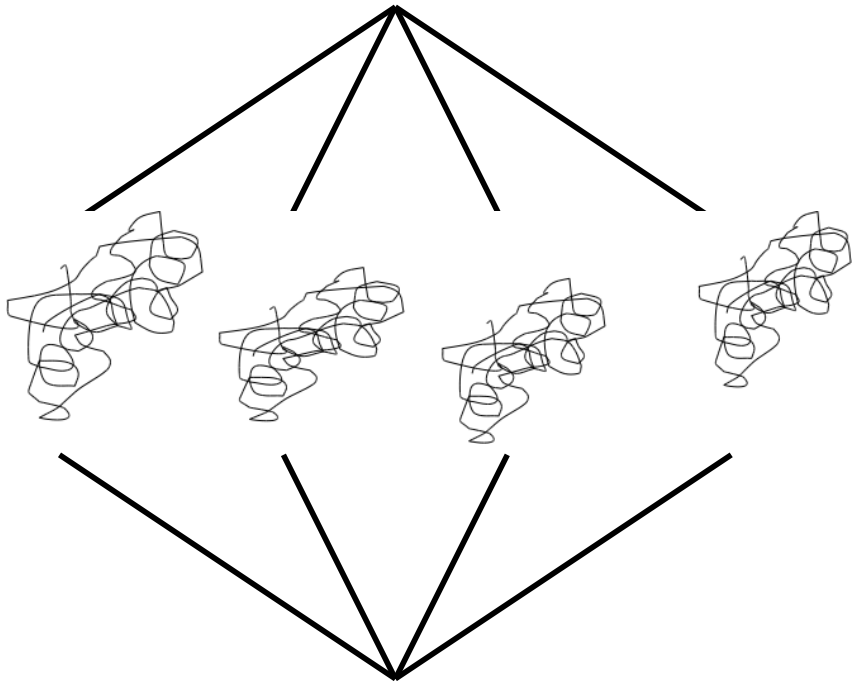


A pair of structures within the hemoglobin super-family that have sequence identity 17%: beyond the applicability of comparative modeling

A Simplistic View of Threading

Target sequence G-A-L-T-E-S-Q-V-P-...

Fold library



Build model

Calculate energetic score

Rank and evaluate model

An Algorithmic View of Threading

Step 1: Construction of template library from known folds

Step 2: Design of (energetic) scoring function

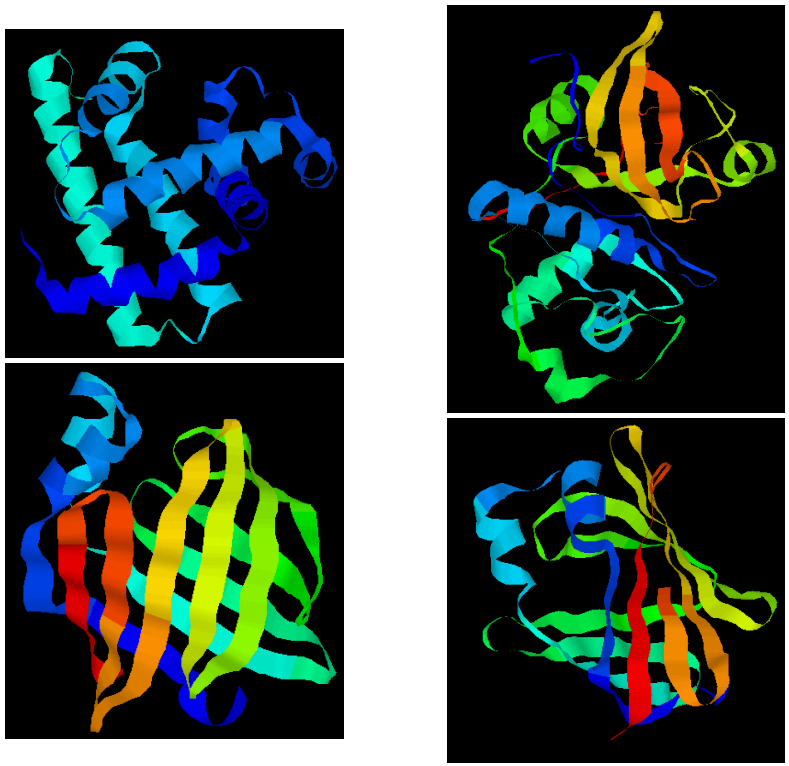
Step 3: Sequence-template alignment

Step 4: Template selection and model construction

Step 5: Model completion, refinement, and evaluation

Threading: Problem Statement

- Threading essentially approaches the following problem:
- Given a query sequence, find which fold "fits" best from a library of known folds
- This essential component of threading is often known as fold recognition (recognizing the best fitting fold among the library of available folds)

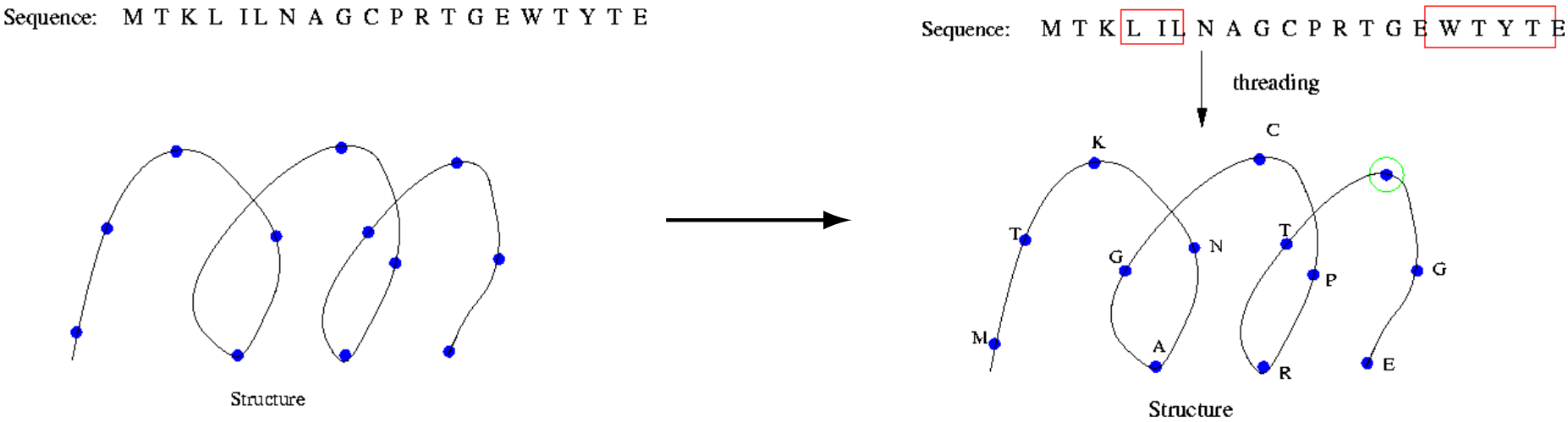


Fold Library

MTYKLILNGKTKGETTTEAVDAA
TAEKVFQYANDNGVDGEWTYTE

Query sequence

Threading: Pictorial Presentation



Threading: Sequence-structure Alignment

- Finding the best fold for a query sequence entails addressing the sequence-template alignment problem, which consists of two sub-problems:
- Design of an effective scoring function to determine the "goodness" of a fit that results from an alignment
- Known as the threading energy function or potential
- Rapid alignment algorithm so that the sequence can be efficiently threaded to many folds during the search for the best one
- Different methods spanning from hard to soft threading consider aligning sequence to sequence (like comparative modeling), sequence to structure, sequence to contact environment etc.

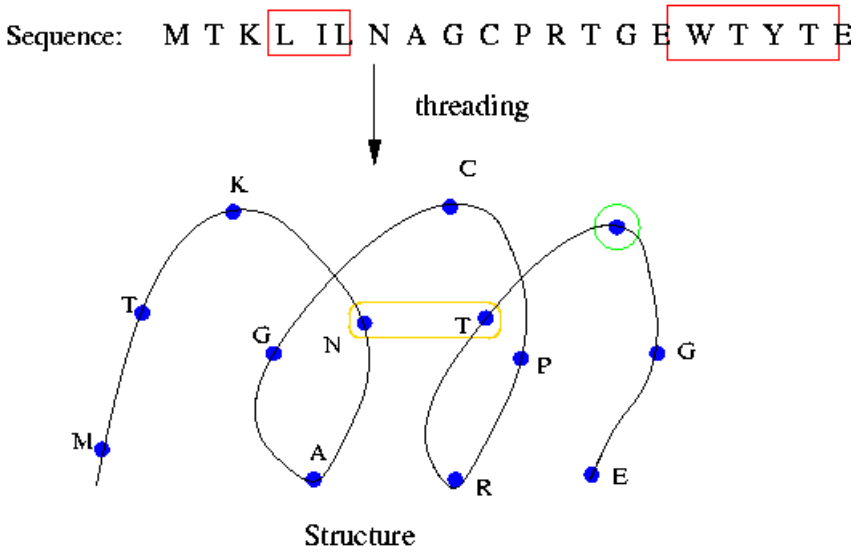
Threading Energy Function

- Sequence-template alignments are scored using a threading energy (objective) function
- The function scores the compatibility between the query sequence of amino acids and their corresponding positions in a given template
- The objective function essentially scores compatibility using specifically-chosen parameters such as:
  - Amino-acid preferences for solvent accessibility
  - Amino-acid preferences for particular secondary structures
  - Interactions between neighboring amino acids
  - Inexpensive physics-based terms are also incorporated

Threading Energy Function

how preferable to put two particular residues nearby: E_p (Pairwise potential)

alignment gap penalty: E_g ? (gap score)?



how well a residue fits a structural environment: E_s (Fitness score)

sequence similarity between query and template proteins: E_m (Mutation score)

Consistency with secondary structures: E_{ss}

$$E = E_p + E_s + E_m + E_g + E_{ss}$$

Minimize E to find the best sequence-template alignment

Threading Energy Function

There are three main approaches to the alignment sub-problem:

1. Sequence-sequence alignment (1D-1D)
   - Align query sequence with template sequences
   - This alignment guides the threading of sequence into structure
2. Consider structural environment in addition to sequence (3D-1D)
   - Align query sequence to a string of descriptors that describe the 3D environment of the considered folds
3. Consider pairwise contacts in folds
   - Contact graph guides the alignment of the query sequence
   - Most successful threading methods fall in this category

Sequence - Sequence Alignment to Template

- Essentially similar to the process used in comparative modeling
- Advantage:
  - Simple dynamic programming methods can used to align the query sequence to the sequences of the templates
- Disadvantages:
  - Templates may have low sequence similarity to query sequence
  - Fails to consider interactions between neighboring amino acids

There is no rigorous boundary between comparative modeling and threading in terms of methodology: rule of thumb is that when the alignment takes into consideration structural aspects (besides sequence aspects) and the templates are remote homologues, then we talk about threading rather than comparative modeling.

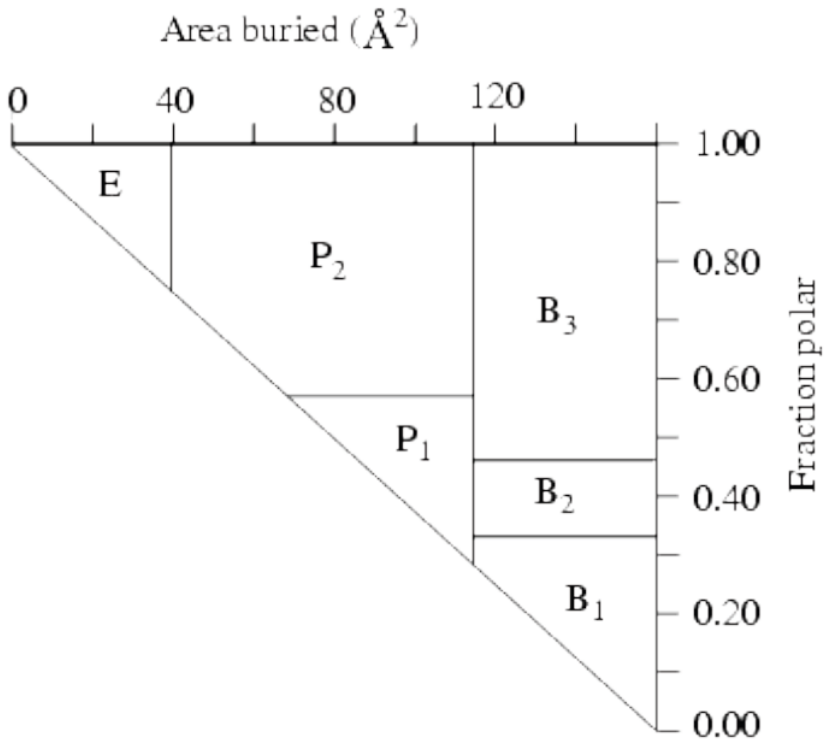
Sequence - Sequence Alignment to Template

1. Sequence-sequence alignment (1D-1D)
   - Align query sequence with template sequences
   - This alignment guides the threading of sequence into structure
2. **Consider structural environment in addition to sequence (3D-1D)**
   - **Align query sequence to a string of descriptors that describe the 3D environment of the considered folds**
3. Consider pairwise contacts in folds
   - Contact graph guides the alignment of the query sequence
   - Most successful threading methods fall in this category

Sequence - Sequence Alignment to Template

- Instead of aligning the query sequence to a template sequence, the query sequence is aligned to a string of descriptors that capture the 3D environment of the template structure
- For each amino-acid position in a template structure, one determines:
  - How buried it is (buried, partly buried or exposed)?
  - The fraction of surrounding environment that is polar (or apolar)?
  - The local secondary structure (helix, sheet, or other)
- This information is encoded in a scoring matrix that (similar to the scoring matrices used for sequence alignment) is used to guide the alignment in a dynamic programming framework

Amino Acid Environments



Each environment is further divided into three sub-classes according to the secondary structure of the amino acid (α-helix, β-strand, or other)?

- **B1:** buried and hydrophobic environment
- **B2:** buried and moderately polar environment
- **B3:** and buried and polar environment
- **P1:** partially buried and moderately polar environment
- **P2:** partially buried and polar environment
- **E:** exposed to solvent

Bowie, J.U., Luthy, R., and Eisenberg, D. 1991. A method to identify protein sequences that fold into a known three-dimensional structure. Science 253: 164–170.

Tabulating Amino Acid Environments

- Each position in the template structure is mapped into one of the 18 possible environment classes (a vector of 18 descriptors)
- Different amino-acids prefer different environments
- This preference is captured by compiling descriptors for each amino acid over structures in the PDB
- The number of times an amino acid appears in a specific environment class is tabulated to obtain frequencies
- These frequencies are normalized for each amino acid to obtain probabilities of the form P(x, y)
- P(x,y) = probability of finding amino acid x in environment class y

Amino-acid Environment Matrix

20 amino acids

Environment class	W	F	Y	...
B1α	1.00	1.32	0.18	...
B1β	1.17	0.85	0.07	...
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

18 classes

$$\text{Score} = \ln \frac{Pr(\text{residue j in environment i})}{Pr(\text{residue j in any environment})}$$

From Environment Matrix to Profile

- Each template structure is mapped to an environmental profile
- Each position in the template is mapped by the sequence identity of the amino acid (find column) and the actual environment of that amino acid in the structure (find row)
- Each position is scored in this way total profile score of the template is additive, so obtained by summing over the scores of the amino acids

Environment class	W	F	Y	...
B1α	1.00	1.32	0.18	...
B1β	1.17	0.85	0.07	...
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

Environment and Sequence Alignment

- Aligning the query sequence to the profile of a specific template is similar to the traditional sequence-sequence alignment with DP:
- For each amino acid s_i in the query sequence, the cost of modeling it through amino-acid tj in the template is either:
  - that of "mutation" : environment cost provides this
  - that of insertion: gap penalty incurred

	$e_1 e_2 \ldots e_n \leftarrow$ environment classes
s_1	
s_2	
.	
.	
.	
s_m	
$\uparrow$	
new sequence	

Sequence - 3D Profile Alignment

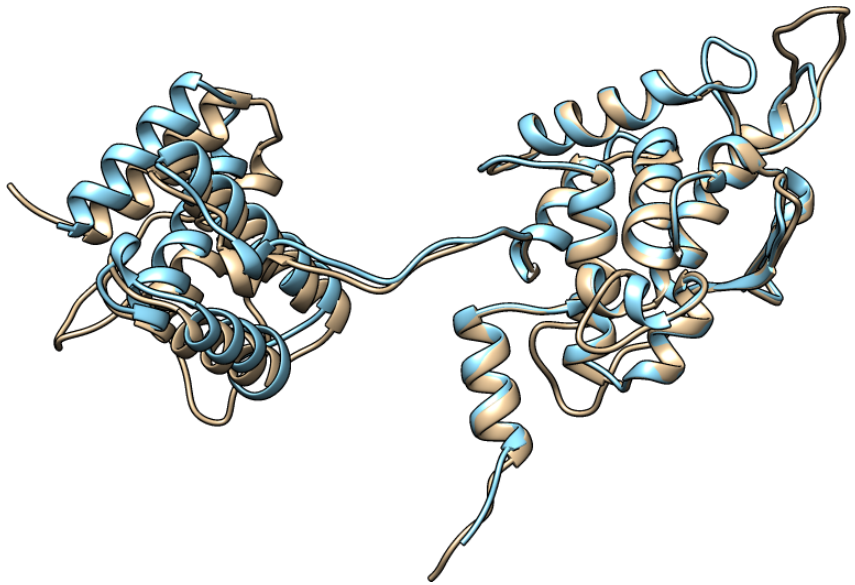
- Advantages:
  - Provides good-quality models
  - It considers not only sequence, but structural considerations encoded in the environment of an amino acid
- Disadvantages:
  - Amino acids are considered/threaded independently of one another
  - This is inherent in the additive score used in the dynamic programming formulation of the problem

Sequence - Sequence Alignment to Template

1. Sequence-sequence alignment (1D-1D)
   - Align query sequence with template sequences
   - This alignment guides the threading of sequence into structure
2. Consider structural environment in addition to sequence (3D-1D)
   - Align query sequence to a string of descriptors that describe the 3D environment of the considered folds
3. **Consider pairwise contacts in folds**
   - **Contact graph guides the alignment of the query sequence**
   - **Most successful threading methods fall in this category**

RAPTOR-X – The Next Generation

- Integrating global and local context specific information.
- Includes both alignment and template selection.
- Currently among state of the art.



RAPTOR-X – The Next Generation

- Given a protein sequence S and a template T and one of their alignments A, let $P(A|S,T)$ denote the probability of A being generated from S and T using the alignment method.
- We define the potential of A, denoted as $U(A|S,T)$, as follows: $U(A|S,T) = \log \frac{P(A|S,T)}{P_{ref}(A)}$ where $P_{ref}(A)$ is the background (or reference) probability of A, i.e. the probability of A being generated from two randomly selected proteins with the same lengths as S and T, respectively.
- Intuitively, an alignment is good as long as its probability is much better than the expected probability.

Protein threading using context-specific alignment potential Jianzhu Ma, Sheng Wang, Feng Zhao and Jinbo Xu, ISMB 2013

RAPTOR-X – The Next Generation

- An alignment is optimal if it maximizes its potential. That is, given a sequence and a template, we can find their optimal alignment by maximizing the alignment potential function.
- Estimation is done using a probabilistic graphical model that estimates the log-likelihood of one pair of residues being aligned based on their context-specific information:
  $F(A|T,S) = \sum_{i=1^L} E(a_{i-1}, a_i, T, S)$
- E is a neural network with one hidden layer, estimating the log-likelihood of state transition from a_{i-1} to a_i, based on protein features in a local window (of size 11) centered at the two residues to be aligned.
- Protein features include sequence similarity, structure-derived amino acid substitution matrix, secondary structure and solvent accessibility similarity.
- Global alignment using a distance-based potential.

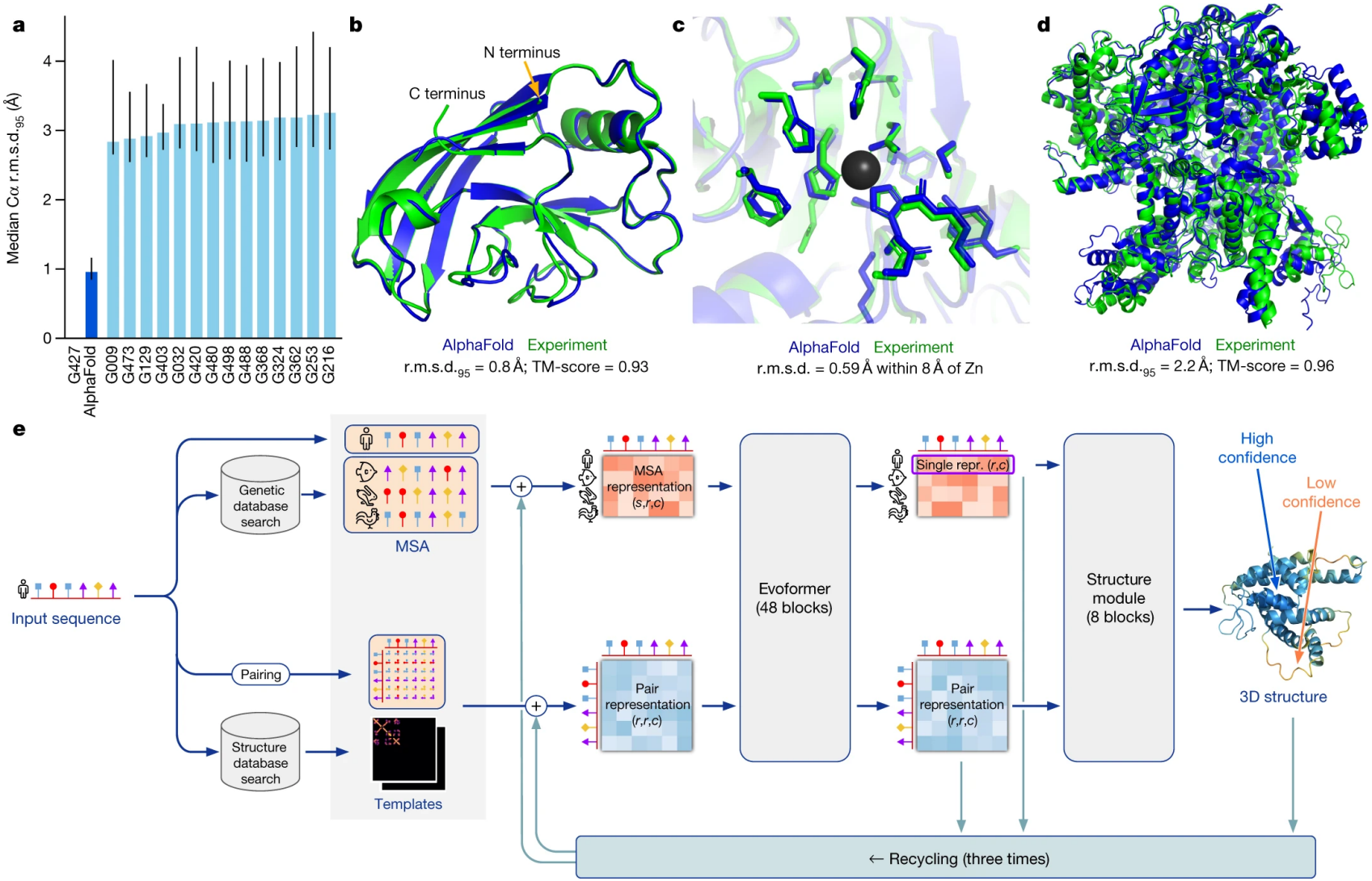
Final Words on Threading

- More than 99% threading instances can be solved directly by linear programming
- The rest can be solved by branch-and-bound with only several branch nodes
- Less memory consumption
- Less computational time
- Easy to extend to incorporate other constraints

Ab Initio Folding

- State of the art methods use fragment assembly, deep learning, HMM.
- Search is done using optimization techniques (Monte Carlo, MD etc.).
- Example – Rosetta (also expanded to Rosetta@home using distributed computing), TASSER, I-TASSER etc.
- Deep learning: DeepFold, Alphafold etc.

AlphaFold 2



Highly accurate protein structure prediction with AlphaFold, Jumper et al., Nature, 596, pages 583–589 (2021)

AlphaFold – Overview

- Sequence based search + pairwise interactions + geometric modules work together in attention based learning.
- Evoformer – the building block of the network.
- The prediction of the protein structures is viewed as a graph inference problem in 3D space in which the edges of the graph are defined by residues in proximity.
- The elements of the pair representation encode information about the relation between the residues.
- The columns of the MSA representation encode the individual residues of the input sequence while the rows represent the sequences in which those residues appear.
- Within this framework, update operations are applied in each block, in series.

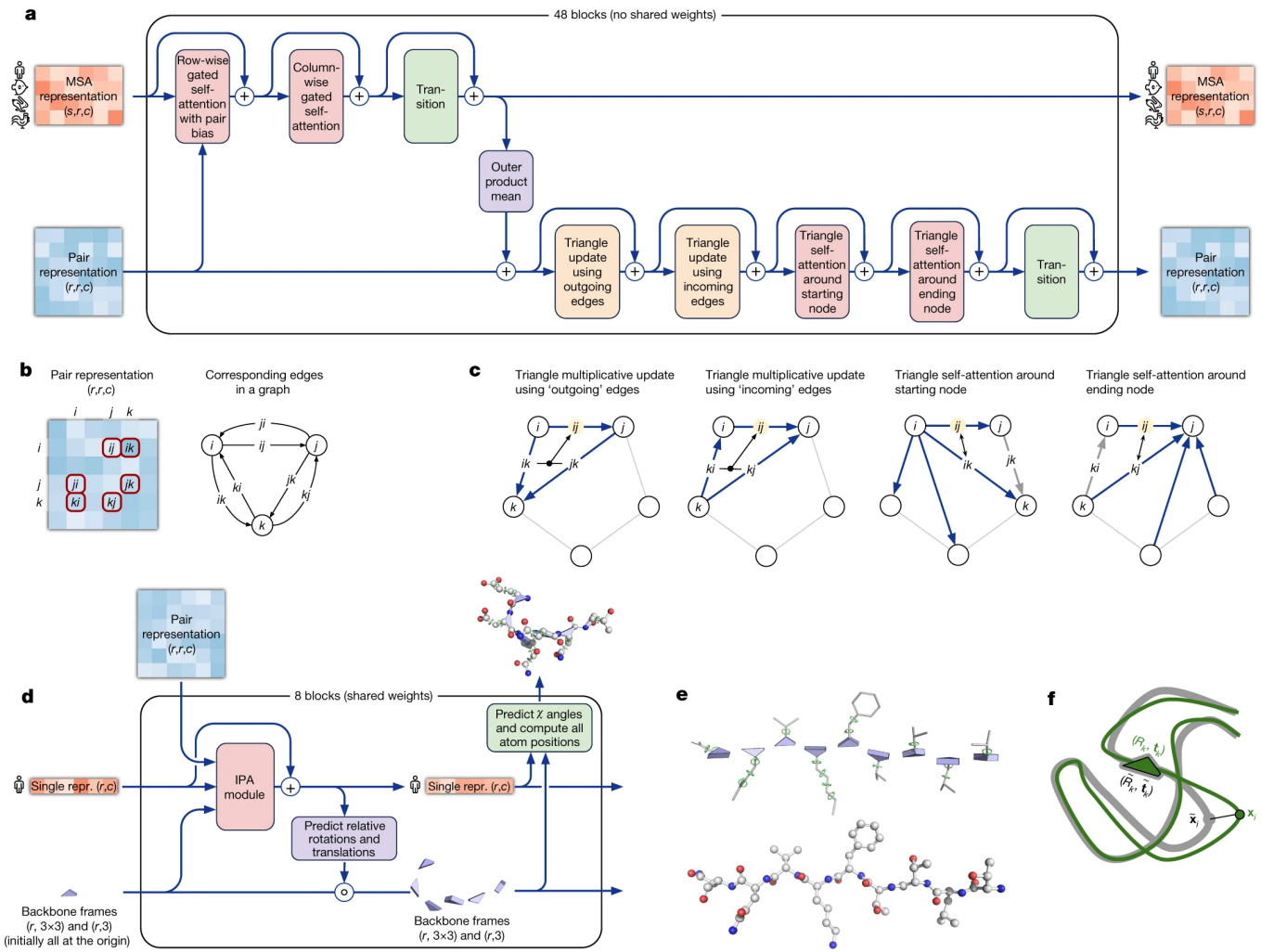
AlphaFold – Evoformers

- Evoformers produce an $N_{seq} \times N_{res}$ array (number of sequences by number of residues) that represents a processed MSA and an $N_{res} \times N_{res}$ array that represents residue pairs.
- The MSA representation is initialized with the raw MSA and is iteratively refined.
- The Evoformer blocks contain a number of attention-based and non-attention-based components.
- The key innovations in the Evoformer block are new mechanisms to exchange information within the MSA and pair representations that enable direct reasoning about spatial and evolutionary relationships.

AlphaFold – The Structure Model

- The structure module – a rotation and translation for each residue – global rigid body frames.
- Representations are initialized in a trivial state with all rotations set to the identity and all positions set to the origin.
- It rapidly develops and refines a highly accurate protein structure with precise atomic details.
- Breaking the chain structure to allow simultaneous local refinement of all parts of the structure.
- A novel equivariant transformer allows the network to implicitly reason about the unrepresented side-chain atoms and a loss term that places substantial weight on the orientational correctness of the residues.
- Iterative refinement takes place by repeatedly applying the final loss to outputs and then feeding the outputs recursively into the same modules.

AlphaFold – Evoformers



Highly accurate protein structure prediction with AlphaFold, Jumper et al., Nature, 596, pages 583–589 (2021)

AlphaFold – Not the End of the Road

3D viewer

Model Confidence:

- Very high (pLDDT > 90)
- Confident (90 > pLDDT > 70)
- Low (70 > pLDDT > 50)
- Very low (pLDDT < 50)

AlphaFold produces a per-residue confidence score (pLDDT) between 0 and 100. Some regions below 50 pLDDT may be unstructured in isolation.

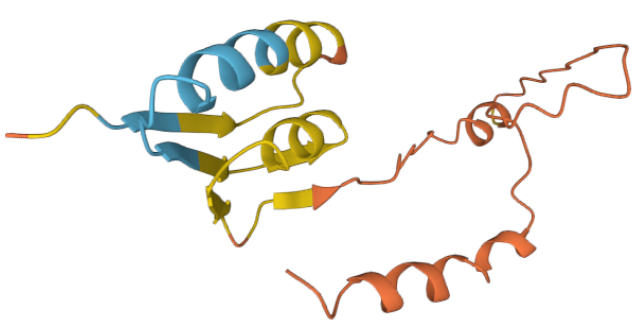
Views

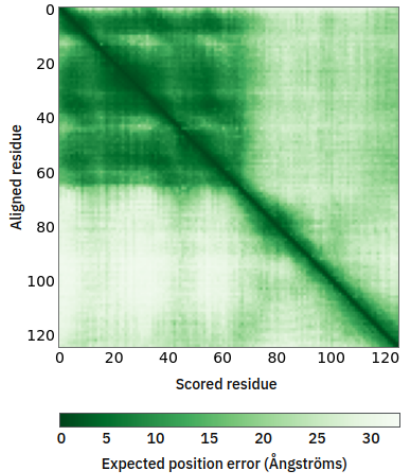
Predicted aligned error

Sequence Features (coming soon)

Sequence of AF-Q47664-F1 Chain 1: E. coli topA... A

```
1  11  21  31  41  51  61  71  81  91  101  111  121
HQPQVALRLEALVAWCMLRAGASQLQRLRDRNIPLTVTVDKVAASGGYMMACVADKIVSAPFAIVGSIGVVAQMPNFNRFLKSKDIDIELHTAGQYKRTLTLLGENTEEGRGEIPRRVERNASVV
```





AlphaFold 3.0 – More Improvements

- Available since 2024, significant improvement over 2.0
- Both 2.0 and 3.0 are available as open source.
- `https://github.com/google-deepmind/alphafold3`
- The replacement of the Structure module in AlphaFold2 by a Diffusion module in AlphaFold3.
- Extend the capability to ligands and DNA/RNA.
- The Pairformer module in AlphaFold 3 replaces the Evoformer module in AlphaFold2 – some internal changes.