Protein Design

- Why ?
- Experiments
- Computational Problems

- Monte Carlo, pruning methods

- Energies, energy differences (ΔE, ΔG)
  - why energies are difficult

Protein design

Not talking about

- design to change enzyme specificity
  - anything to change ligand binding

Am talking about

- you have a useful protein – probably enzyme
  - want a more stable version
  - stability ?
    - pH
    - solvents
    - temperature ←
- assumption
  - If I write a sequence, you can synthesise it

Experiment

Trial and error

- propose changes to sequence, try it out – not much fun

For binding

- phage display, *in vitro* evolution

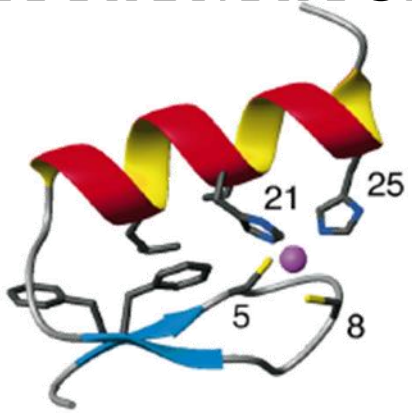
- Computational methods…

History

1997

designed **QQYTAKIKGRTFRNEKELRDFIEKFKGR**

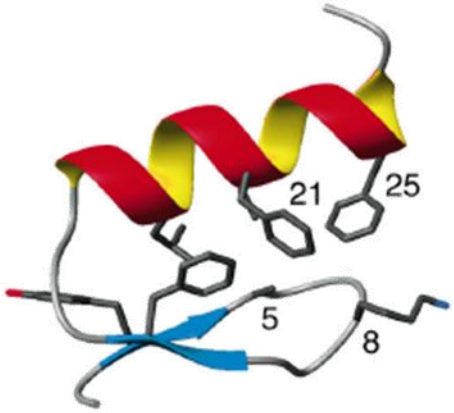
native **KPFQCRICMRNFSRSDHLTTHIRTHTGE**



Zn-binding protein

- redesign sequence
- about 20% similar to start

- synthesised
- structure solved by NMR



These methods are not routine

Dahiyat, B.I, Mayo, S.L. (1997), Science 278, 82-87

Specify the problem

Make a useful structure more stable

native protein

"improved" protein

Rules

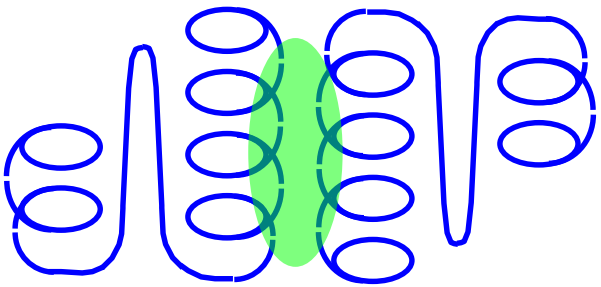
- structure should not change
- some sites are fixed (active site, other binding)

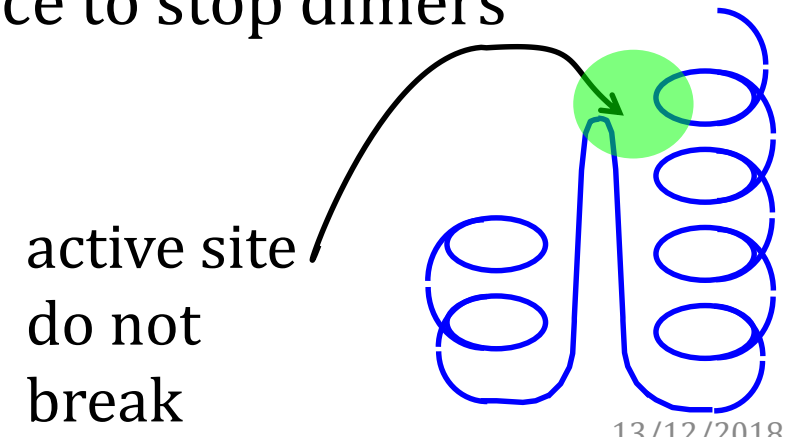
Fixing / specifying residues

Examples

- lysine (K) often used for binding
  - change a residue to K and protein does not fold
  - mission:
    - adapt the rest of the residues to be stable

- change all residues, but not those in active site
- change some residues at surface to be soluble
- change some residues at surface to stop dimers





Scores versus search

score / energy

- a function $f(\{s\}, \{\mathbf{r}\})$
  for sequence $\{s\}$, coordinates $\{\mathbf{r}\}$
- if I change a residue in s
  - does my structure become more stable ?

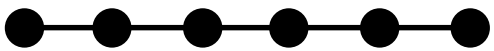
search method

- many possible sequences
- how to decide which residues / sequences to try out

Searching

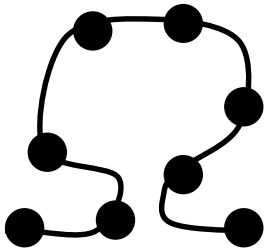
Imagine we have score function $f(\{s\}, \{\mathbf{r}\})$

Systematic search



for n_{res} we have $20 \times 20 \times \cdots = 20^{n_{res}}$

Friendly search space ? (can I just optimise each site ?)



- change here affects there affects …

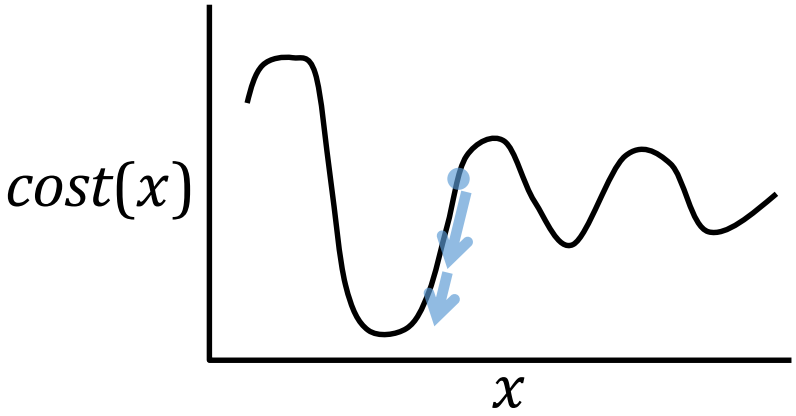
Consequence

- if I change this one, I have to change that one, then next, …

Optimisation Problem

Easier problems

- gradient information
- can recognise minima



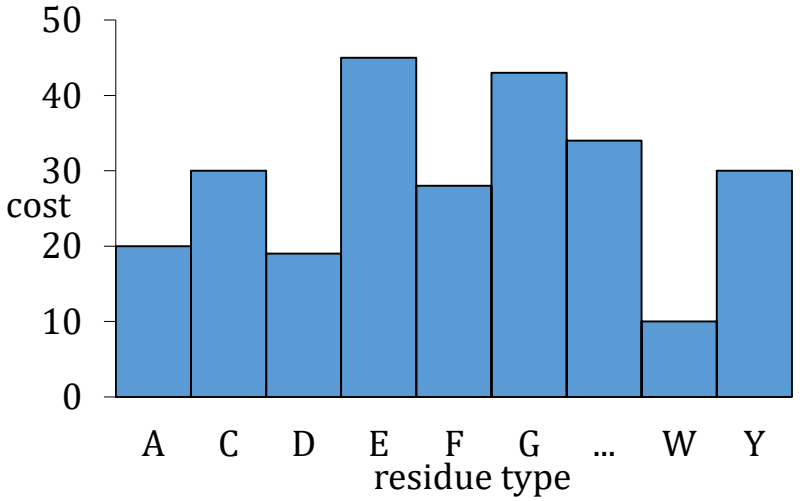
- we have directions
  - if a move in one direction is good, try to keep going

Contrast with sequences / discrete problems

Discrete problems

- no gradients
- no directions – labels are arbitrary (ACDE or ECAD)

- lots of local minima
  - diagram is for just 1 of n_{res} sites

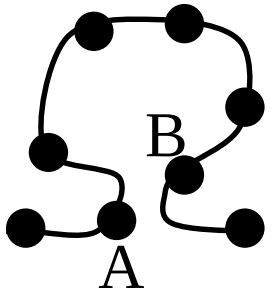


a bad method

- some starting sequence $\{s\}$ from s_1 to $s_{n_{res}}$
- score $E_0 = f(\{\mathrm{s}\}, \{\mathbf{r}\})$
- pick a position i
- change s_i to some different residue type (trial)
- score $E_{trial} = f(\{\mathrm{s}_{trial}\}, \{\mathbf{r}\})$
- if $E_{trial} < E_0$ then
  accept s_{trial} say $\{s\} \coloneqq \{s_{trial}\}$

Will it work ?

- simplest example of correlations
- any change to A – must change B



Taking bad moves

I cannot simply look for good moves – alternatives ?

- change two residues at a time ? (400 possibilities)
- three residues at a time (8000 choices)
  - will not generalise

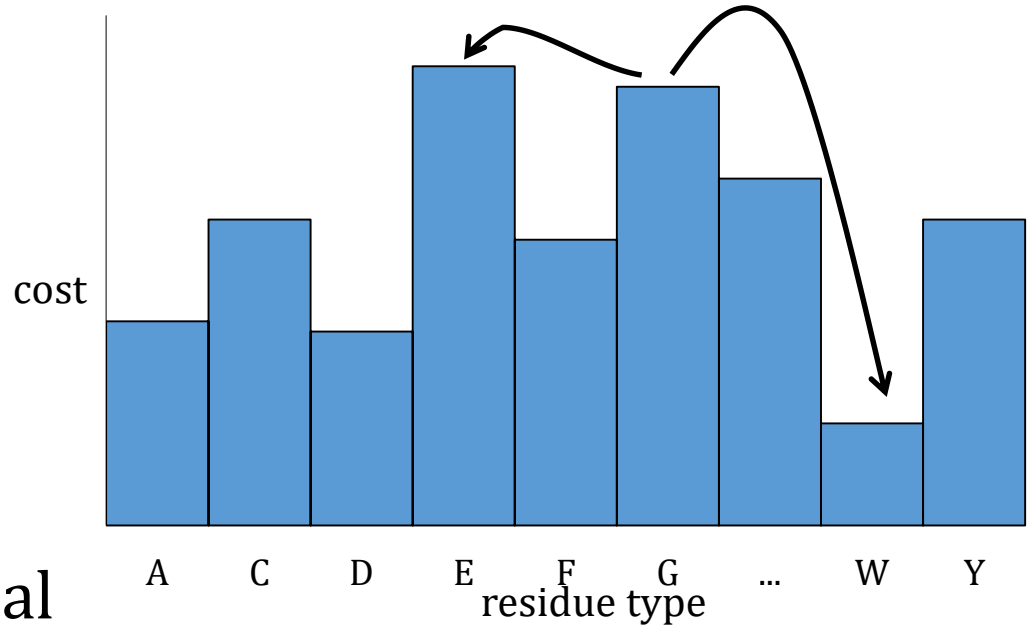
Different philosophy

- change one residue at a time, but allow the system to sometimes get worse

Monte Carlo - accepting bad moves

- decide on a move (change some s_i)

- if system is better
  - keep trial move
- if system is worse
  - slightly worse
    - probably keep trial
  - much worse
    - throw away



Name of procedure: Monte Carlo

- what else do we need ?

Annealing

At the start, we are far from optimum

- take big moves, accept bad energies more often

Later, we are closer to a good sequence

- do not accept so many bad moves
- try to optimise details, locally

Name for this idea – annealing

- formalise this ? secret overhead

Monte Carlo properties

Deterministic ?

- you and I write programs without bugs
  - get different answers

Do we get to optimum ?

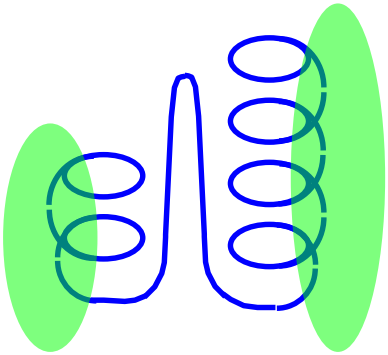
- almost never

Difficult ?

- very easy to program

Fundamental problem – search space is too big

reducing the search space – types of residue



look at green areas

- should be charged or polar
- not 20^n maybe about 11^n

Can do the same for buried residues (A, I, L, M, F, V)

- gives 6^n

reduce search space – remove dead ends

Consider one position with 20 possibilities

- I can have a residue of type a
  - what is the best score you could possibly have for a given his neighbours ? a_{best}
  - what is the worse score you could have ? a_{worst}

At this site loop over the 20 amino acids

- for the 20 possibilities a
  - for 19 alternatives b
    - if $b_{worst} < a_{best}$
      - a cannot be possible at this position

Name: pruning / dead end elimination / wegschneiden

Pruning

- sometimes finds only one amino acid type is possible at a position

- usually makes search space much smaller

side-chain conformations / rotamers

Side chain conformations ? same problem as modelling

- need score of side-chain, but you do not have coordinates

- use coordinates from rotamer library
  - maybe simplified to one angle χ_1

Original problem - choose from 20 amino acid types

- now ≈ 20 × 3 types

Fits naturally into

- Monte Carlo – try a new amino acid rotamer
- Pruning method – which amino acid+rotamer can be excluded ?

score functions / energies

How sophisticated ?

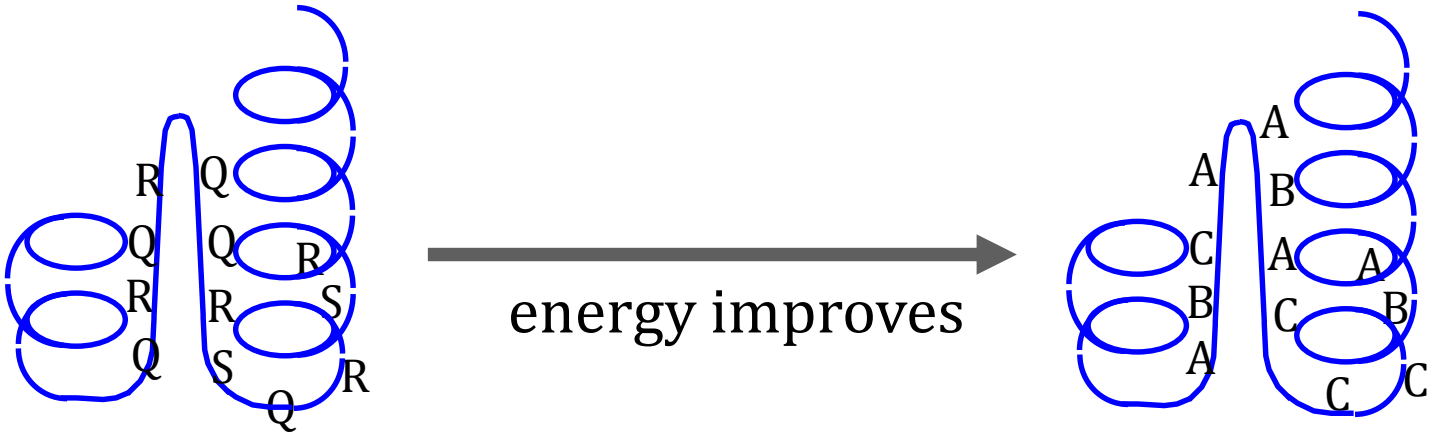
- backbone is fixed
- using rotamers
  - no need to worry about bonds, angles, torsion angles

Mainly

- van der Waals
- electrostatics

Score functions / Energies

Is energy enough ? Is this relevant ?



native protein

"improved" protein

Question here is stability

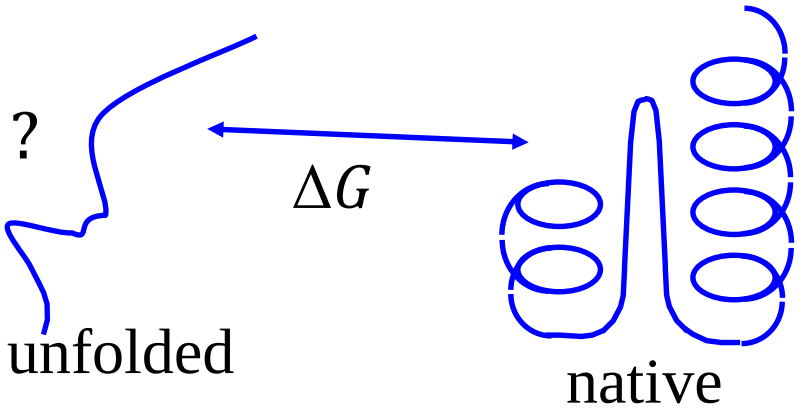
- two problems

energy differences

Define stability

- free energy change upon folding ?

$$\Delta G = \Delta H - T\Delta S$$



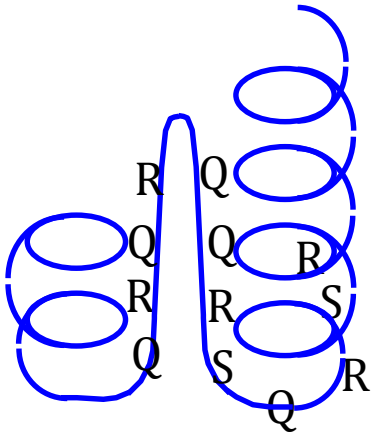
consider ΔH (what we can model)

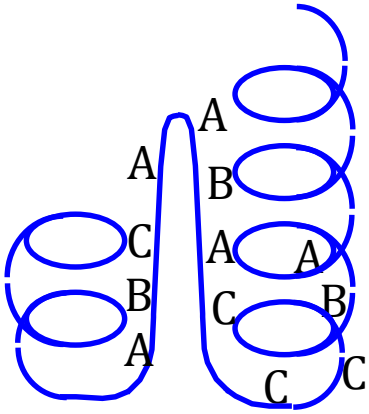
Have I considered $H_{unfolded}$?

Am I looking at the correct energy change ?

Change an amino acid and native has better energy

- what if unfolded also has better energy ?
  - think of a surface residue

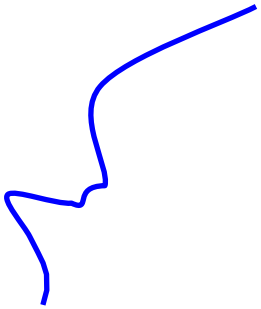


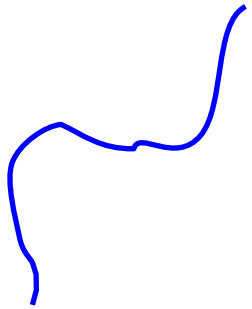


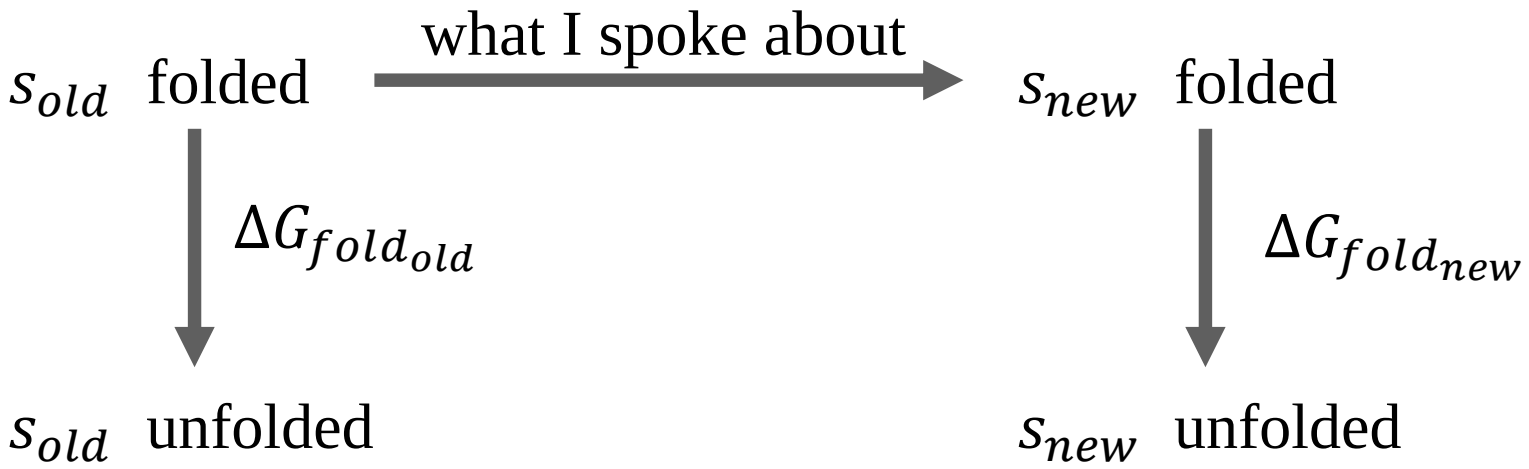
s_{old} folded → s_{new} folded

s_{old} folded → stability$_{old}$ → s_{old} unfolded

s_{new} folded → stability$_{new}$ → s_{old} unfolded







What really matters

$$\Delta G_{fold_{old}} - \Delta G_{fold_{new}}$$

It is not enough just to look at structures

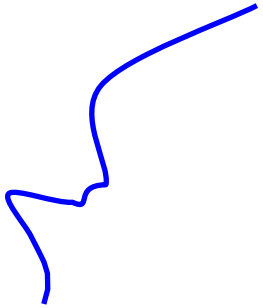
- should also look at non-structures / unfolded

What have I neglected ?

Free energy change has entropy $\Delta G = \Delta H - T\Delta S$

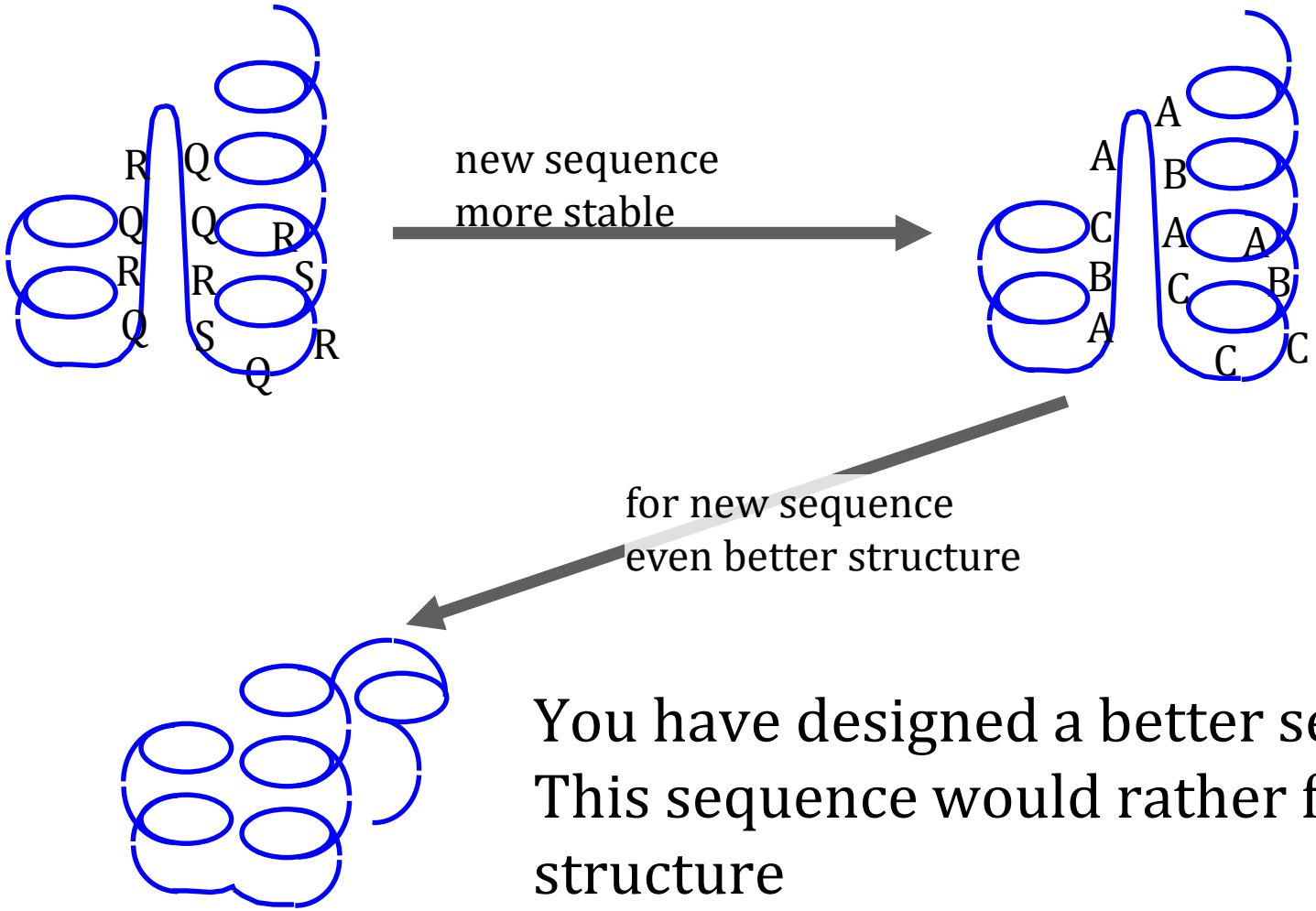
- my energy models do not have ΔS
- is this important ?

- the unfolded states are very disordered



Negative design

Another complication – alternative folds



You have designed a better sequence
This sequence would rather fold to a new structure

Negative design – cure ?

Typical approach

- at each optimisation step
  - check alternative folds – may not be easy

- tricks in scoring function

How well do methods work ?

Success stories

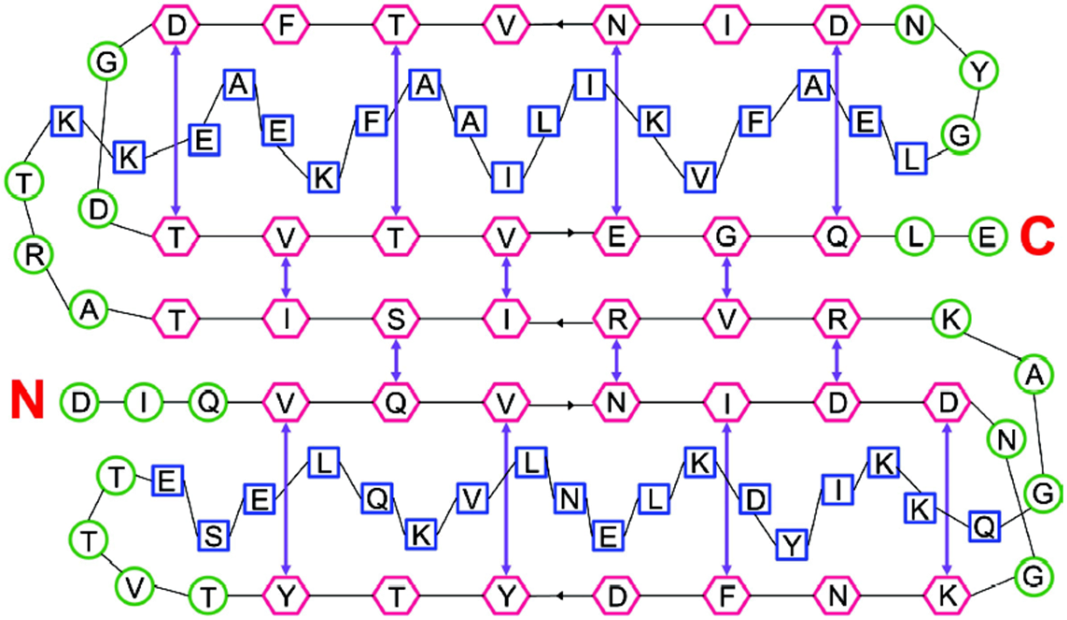
- example at start of lecture
- many more individual stories

Not at all routine

- many failed attempts not spoken of

Spectacular Success

- "topology" order of secondary structure units
- write down a topology that does not exist in nature



Kuhlman, B.; Dantas, G.; Ireton, G.C.; Varani, G.; Stoddard, B.L.; Baker, D.
Science 2003, 302, 1364-1368.

Methods

- pure Monte Carlo

Result

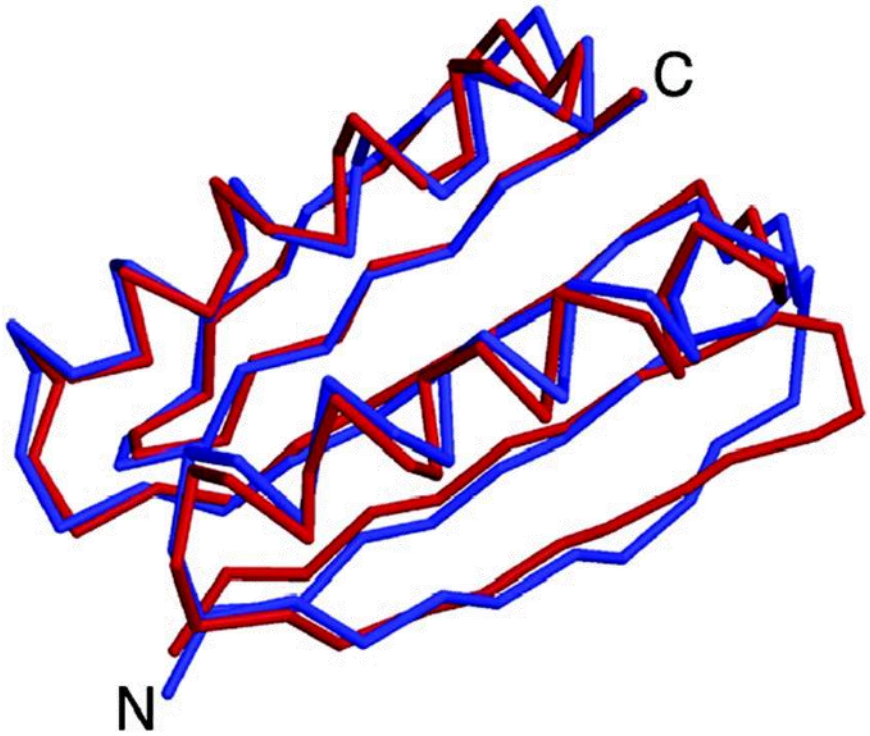
- apparently new sequence

Structure

- as predicted
- solved by X-ray
- phasing story

Problem solved

- unclear (how many failures ?)



Optimism

You do not have to find the optimal sequence

- think of man, monkey, horse haemoglobin
- lots of room for changes in sequence

Pessimism

Designed sequences must

- fold
- be expressed and produced

for Klausur

- why optimise sequences ?
- search space – size, reducing
- optimisation properties – continuous, discrete, Monte Carlo
- rotamers
- score function – energies, ΔG
- negative design