A survey into

Protein Folding

– Curves and Energy Functions

Martin Lundgren

Undergraduate Thesis
Supervisor: Antti Niemi

October 2007

Department of Theoretical Physics

Contents

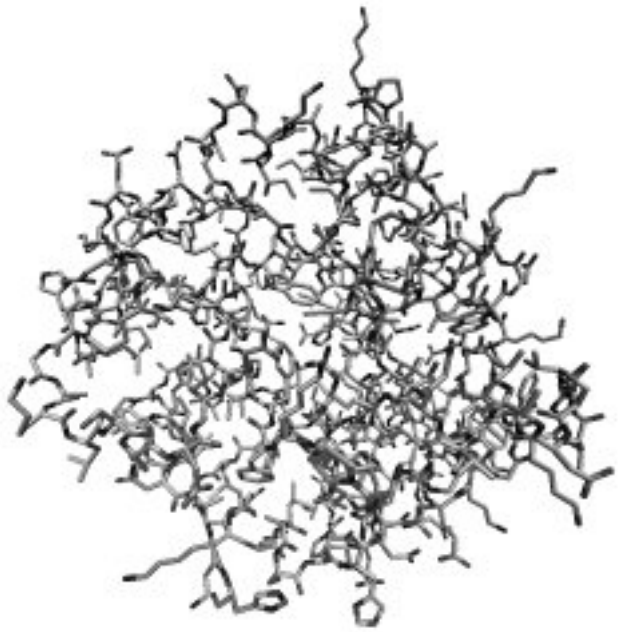

Figure 1: One monomer of the protein triose phosphate isomerase.

1 Proteins

1.1 Introduction to proteins

A protein is a large organic molecule consisting of a chain of amino acids joined together by peptide bonds. Linked in a chain, the individual amino acid is called a residue. The amino acids consists of the backbone atoms which are part of the main chain and almost identical for all different amino acids, and the sidechains which are individual and can vary in both size and chemical properties. The protein have structure on several different levels. First it is the primary structure which simply is the list of residues along the protein chain. The secondary structure is the local structure composed of a few residues. Finally the tertiary structure is the global structure of the protein.

The proteins are used for many different purposes in the cells. They can, for example, be enzymes, taking part in chemical reactions, or they can perform many other vital functions in the cells. While the protein essentially is a long chain of amino acids, the function of the protein depends solely on the 3-dimensional shape of the protein, its native structure, which in turn is uniquely defined by the chain sequence. Hence the knowledge of the sequence of amino acids is only useful for determining the function of a protein if there is a way to find the unique 3D structure from it.

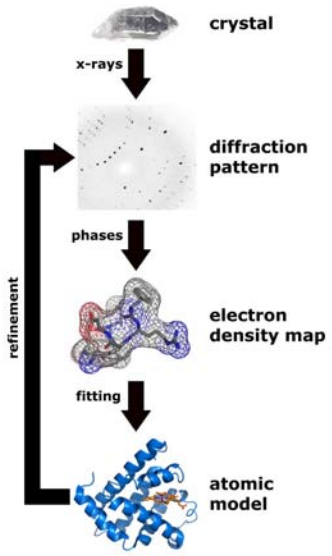

Figure 2: X-ray crystallography. (Image by Thomas Splettstoesser)

1.2 Practical methods

The oldest method for determining the structure of proteins is X-ray crystallography. This is based on the study of the diffraction pattern of X-rays scattered from a pure crystal protein. Using this method the structure of myoglobin was determined in 1958. This method is the one most widely used but it can not be used for proteins that does not crystallize easily.

A more recent method used is NMR-spectroscopy. The idea here is to use the magnetic resonance of the nucleus since, ideally, each nucleus has a unique small shift in the resonance because of its chemical environment. Normally this method can only detect the hydrogen because of the lack of nuclear spin of carbon and oxygen and the high quadropolar moment of nitrogen but if the protein is first enriched with carbon-13 and nitrogen-15, then these can also be detected.

1.3 Theoretical methods

Now that most of the human genome has been mapped we have an enormous amount of genetic data. Since the sequence of amino acids in the protein chain is defined by the DNA code, we also have a massive amount of information about protein chain sequences. However we have much less information on protein 3D structure because of the time consuming methods, described earlier, used to determine it. A good reliable method to determine the 3D structure from the chain sequence would give us very much information on the function of the

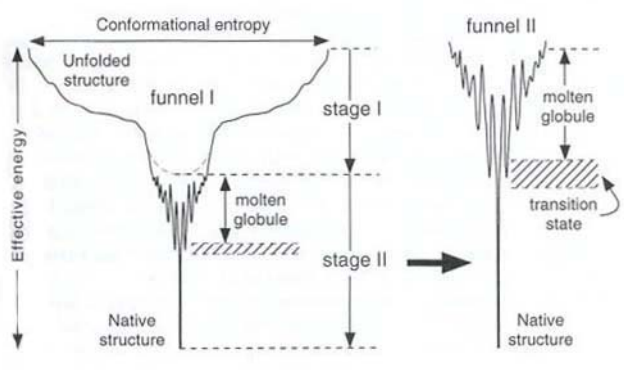

Figure 3: The energy landscape of a folding protein.

protein and the corresponding gene. This knowledge can then be used e.g. in the creation of more effective pharmaceutical drugs.

The inverse of protein folding is to determine the chains that would fold to a particular 3D structure. This knowledge can be used to design a new protein with a desired attribute. A problem with this is that it is widely believed that there are several theoretically possible structures that no chain would ever fold to. In fact, while it seems that the total amount of possible 3D structures are almost infinite, only a small fraction is used in nature, with many proteins sharing a very similar structure.

There are several difficulties in the prediction of protein folding. The amount of possible structures for a polypeptide chain is so large that it quickly becomes impossible to simply test all conformations, even for a chain of just a few residues. Then there has to be a way to actually measure how good the current conformation is. This requires an energy function that can distinguish the native and native-like structures from all other conformations. The potential well around the native conformation is important. A function that gives 0 for the correct conformation and 1 for all the rest would help you test if you have the correct one but would also be totally useless in trying to find it. Another thing that can make matters more complicated is that the native structure might not have the thermodynamically lowest energy. There are also the effects of the local environment to take into account, like the interactions with the solvent and the aid of other proteins in the folding process

The last part necessary is an algorithm for searching the huge conformational space for low energy solutions. Algorithms used include molecular dynamics, Monte Carlo simulations, genetic algorithms and diffusive methods. All of these methods require fast supercomputers to do any kind of realistic simulations on real proteins.

Even though the conformation space is so big in theory, most of the conformations are not possible because of steric and chemical constraints. There seems to be just a few thousand allowed structures, with minor variations.

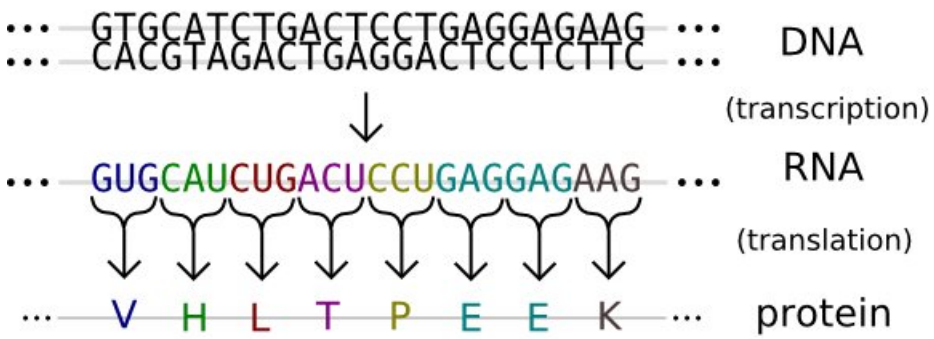

Figure 4: The main pathway, DNA to RNA to protein.

Another question is if the protein folds sequentially or combinatorially. In other words, if the secondary structure formed later in the process affects the secondary structure already formed. Most models use sequential folding simply because it requires much less computer power.

1.4 Protein folding

The primary structure of the protein is decided by a, from DNA transcribed, RNA string, where each three-letter combination stands for one of the 20 amino acids, e.g. GCU is for Alanine, and the commands to start and stop. The folding process starts while the primary structure is still being formed and the first thing that happens is that the chain of amino acids collapses into a molten globule. An intermediate step in the process of forming a protein. These early interactions are mainly local and driven by hydrophobic effects. Here the secondary structure is formed, the most common ones being the α-helix and the β-sheet. The α-helix is a coiled sequence where every amino acid residue shares a hydrogen bond with the residue four residues earlier. The typical α-helix is around 10 residues or 3 turns long but they can be much longer. The willingness of each amino acid to form an α-helix depends on the side chain and what effects it has on the ability to hydrogen bond and the allowed angles on the backbone structure. For example Alanine favours α-helix structure while Serine tends to disrupt it. The second type of secondary structure, the β-sheet, is composed of β-strands. A β-strand is a 6 to 8 residues long flat sequence. The backbone atoms of this β-strand then hydrogen bonds with other β-strands to form a β-sheet. β-sheets can be either parallel or anti-parallel depending on the direction of the β-strands. Another type of secondary structure is the turn that is a 3 or 4 residues long chain with hydrogen bonds between the end residues.

While most sequences always form one type of secondary structure there are some that in one protein forms an α-helix and in another a β-sheet. This seems to indicate that non-local interactions may have some part in the formation of secondary structure.

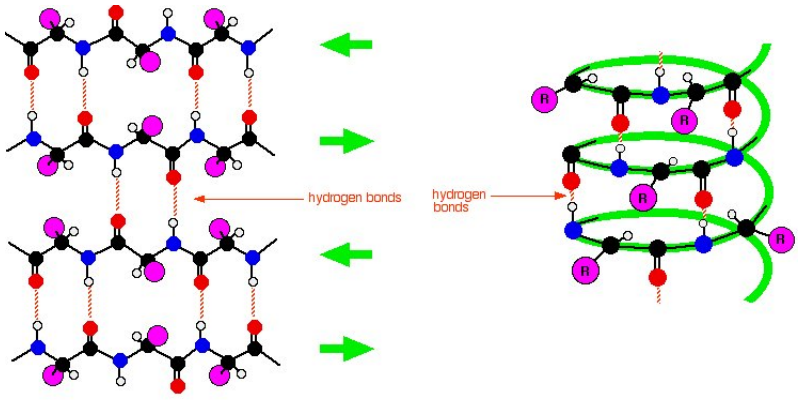

Figure 5: A β-sheet and an α-helix.

The molten globule is similar to the finished protein in that it contains a high level of secondary structure, however it lacks some important parts. First of all it lacks a well-packed tertiary structure, secondly it does not have the functionality of the finished protein, and most importantly it is not stable. The next step in the folding process is a slow search for a native-like core. This search is characterised by how the structure stays quiet in a moderately stable form for a while before it unfolds a bit and folds back into another more compact form. This process can continue several times before it finds the correct structure. When this is complete the last step is a rearrangement on the surface to find the correct native state. Many proteins then go on to form a quaternary structure by assembling with one or more other fully folded proteins. An example of this is hemoglobin. All in all this process can take between a few milliseconds to over a minute depending on the size of the protein. Compare this with the initial collapse into a molten globule, which takes under 100 nanoseconds. Another thing worth mentioning is the Levinthal paradox, which says that if a protein would use some sort of random search to locate its native configuration, it would take an astronomical amount of time to find it. Hence, the protein cannot use a random search and the protein folds by a directed process. There are some theories that the real energy landscape for a folding protein looks like a funnel leading to the native state. This would mean that the protein might fold through different paths and still end up with the same structure.

There are many indications that the main force stabilising the folding process is the backbone interactions, with hydrophobic packing as the main driving force. It seems that polar and charged residues only contributes marginally to the stability of the protein and may in fact destabilise it.

While the 3D structure is uniquely determined by the amino acid sequence, the reverse is not necessarily true. It is not uncommon that two chains where less than 50% of the amino acids are identical fold into the same or similar 3D structure.

1.5 When the folding fails

Besides the protein itself there are some other factors which may affect the folding process. High temperature will cause the protein to misfold, that is to fold into the wrong structure, or unfold the protein from its native structure. The same can happen for high concentrations of solutes or extreme pH. Fully denatured the protein exists just as a random coil with no secondary or tertiary structure. They also lose their biological functions. For some proteins this process is reversible and they will fold back into their native state as soon as the denaturing factors are removed. Others change permanently and can transform into an insoluble mass, e.g. egg whites.

Some cells protect their proteins by producing chaperones, enzymes that help proteins fold and protect them from misfolding and unfolding. A few proteins can never fold in cells without the help of chaperones protecting them from interactions with other proteins during the folding process and some even need the help of several different chaperones to fold.

Misfolded proteins may be the cause of several illnesses such as Creutzfeldt-Jakob disease and Alzheimer's disease. For Alzheimer's disease the problem is that misfolded proteins gathers into an insoluble plaque that the body cannot get rid of.

The prion related illnesses like Mad Cow disease and Creutzfeldt-Jakob disease are a bit harder to explain. They are infectious diseases spread by a misfolded protein with no genetic code and no method of reproducing. The solution to this riddle is that the prions use the bodies own natural system for protein production and then it can refold these proteins into a copy of itself. While some prion diseases are infectious, they can also be inherited or result from a random mutation.

2 Energy functions

2.1 Knowledge based potentials

As was said earlier, the first thing you need to simulate folding proteins is an energy function. Preferably one that can find the correct native conformation among all the other possible and impossible conformations. There are many different ways to do this. From simple homology methods that compare the chain to the database and assign an energy, to physical energy functions derived from first principles and quantum mechanics. I will try to give some representative examples of different energy functions.

The first one to mention is profile score. This function looks at what the environment around the side chain looks like and then compares it to what the environment around the side chain usually looks like for the same amino acid in the database of known protein structures. Using some weights it then assigns an energy depending on the result.

2.1.1 Statistical potentials

A slightly more sophisticated method is the statistical potential, here exemplified by the Sippl potential (1990). It is a general law in statistical physics that the energy of a state is related to the probability of that state being occupied. For a discrete system, with n states and where $f(s)$ is the probability of state s being occupied and $E(s)$ being the energy of state s, this relation looks like this:

$$f(s) = \frac{1}{Z} \exp\left[\frac{-E(s)}{kT}\right] \tag{1}$$

where Z is the Boltzmann sum:

$$Z = \sum_{s=1}^{n} \exp\left[\frac{-E(s)}{kT}\right]. \tag{2}$$

From this it is clear that if you know the probability density function $f(s)$, you can get the energy from:

$$E(s) = -kT \ln\left[f(s)\right] - kT \ln\left[Z\right] \tag{3}$$

but only up to an additive constant $-kT\ln(Z)$.

All that is necessary to find the energy now is an expression for $f(s)$. In this example $f(s)$ is the probability that two residues in the protein are within the distance interval s. While this function $f(s)$ might be interesting, a more useful function is $f_k(s)$, which is the probability of two residues with exactly k residues between them on the chain to be within the distance interval s. Using this function it is easy to see the characteristic peaks from the secondary structure. For example, for $k = 4$ the distance tends to be within the range 5.5 to 6.5 Å in α-helices and within the range 11.0 to 14.0 Å for β-strands.

This explanation of $f_k(s)$ still gives nothing to put into the energy equation. This is where the known conformations enter the picture. There are thousands of proteins with a known 3D-structure, and hence the distances between pairs are also known. Using the frequency of pairs having a certain distance it is easy to calculate the probability of the distance.

Now this would all be well if it were not for the fact that different amino acids have different effects on the folding process and it would be good to separate them in the energy function instead of using $f_k(s)$ which is an average.

The natural solution then is to use different $f(s)$ for different combinations of amino acids. Define $f_k^{ab}(s)$ as the probability of amino acid a and b, with k residues in between to be within a distance interval s. It should be noted that this function is not symmetric in a and b. From now on the ab superscript will refer to a particular amino acid pair and terms without it will refer to an average.

It is more interesting to look at the energy contributions of the individual pair to the averaged energy, the net potential:

$$\Delta E_k^{ab}(s) = E_k^{ab}(s) - E_k(s) \tag{4}$$

This can also be written out in full as:

$$\Delta E_k^{ab}(s) = -kT \ln \left[\frac{f_k^{ab}(s)}{f_k(s)} \right] - kT \ln \left[\frac{Z_k^{ab}(s)}{Z_k(s)} \right] \tag{5}$$

In the last term $Z_k^{ab}(s) \approx Z_k(s)$ so this term can be neglected. The reason for this is that $Z_k(s)$ is really an average of the different $Z_k^{ab}(s)$ and if the variance of $Z_k^{ab}(s)$ around $Z_k(s)$ is not too big then $Z_k^{ab}(s)$ should be close to $Z_k(s)$. However it may be the case for interactions between amino acids with unusual conformational properties to have a $Z_k^{ab}(s)$ value far from $Z_k(s)$. By the time of the printing of the original article, none of these values were actually known.

All this is good if the database is large enough, however it is easy to see that even if the database is large, the number of e.g. Glycine and Asparagine pairs with $k = 6$ might not be big enough to give a statistically valid result. Therefore it is important to separate the actual probability density $f(s)$ from the probability density obtained from the database, $g(s)$. However the data set is large enough that a good approximation is $f_k(s) = g_k(s)$ for the averages.

A reasonable first approximation of $f_k^{ab}(s)$ is $f_k(s)$, and by giving $g_k^{ab}(s)$ more and more effect as the number of known pairs increases, the approximation looks like:

$$f_k^{ab}(s) \approx \frac{1}{1+m\sigma} f_k(s) + \frac{m\sigma}{1+m\sigma} g_k^{ab}(s) \tag{6}$$

where m is the number of measurements of the pair and σ is a weight. For small m this approximation goes like $f_k(s)$ while for $m \to \infty$ it goes like $g_k^{ab}(s)$.

Entering this into the expression for the net potential of the pair gives:

$$\Delta E_k^{ab}(s) = kT \ln \left[1 + m_{ab}\sigma\right] - kT \ln \left[1 + m_{ab}\sigma \frac{g_k^{ab}(s)}{f_k(s)} \right] \tag{7}$$

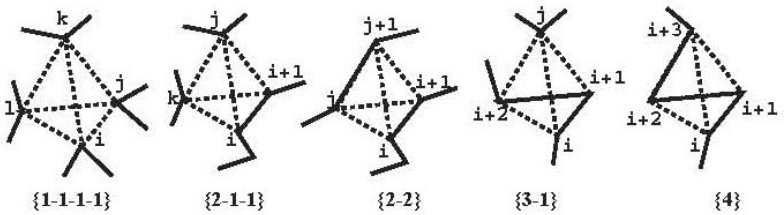

Figure 6: The five different classes of Delaunay tetrahedrons.

From this the total net energy of a sequence S in conformation C can be calculated as a sum of all interactions:

$$\Delta E(S,C) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \Delta E_k^{ab}(s) \tag{8}$$

where N is the total number of residues and s is the distance interval corresponding to $d_{ij} = |\mathbf{x}_i - \mathbf{x}_j|$ for the two amino acids.

This potential is a two-body potential. One way to get more accurate information is to instead use a four-body potential. Then it is not the best idea to use separation along the chain, k, instead the idea is to look at tetrahedrons of residues. Not all of them but only the tetrahedrons consisting of nearest neighbours. This would result in an energy function looking like this:

$$Q_{ijkl}^{\alpha} = -kT \ln \left[\frac{f_{ijkl}^{\alpha}}{p_{ijkl}^{\alpha}} \right] \tag{9}$$

where Q_{ijkl}^{α} is the four-residue contact energy, f_{ijkl}^{α} is the observed frequency of the residue composition i,j,k,l in a tetrahedron of type α and p_{ijkl}^{α} is the expected random frequency of observing i,j,k,l in a type α tetrahedron. The type of the tetrahedron depends on how the residues in it are connected to each other.

2.1.2 Pseudo-physical potentials

Another type of more advanced knowledge based potentials use a more pseudo-physical approach and tries to have a physical reasoning behind the different potentials used. The following potential is from Fujitsuka et al. (2004):

$$V = V_\omega + V_\phi + V_\psi + V_{vdW} + V_{HB} + V_{HP} + V_{Rama} + V_{pairwise} \qquad (10)$$

where the first three terms are the torsion angle terms. They make sure that the bonds are not twisting too much. Especially the ω-term keeps the peptide bond stable.

The next term is for van der Waal interactions. This is separated into several different parts. First there is the separation between local and non-local interactions. The local interactions are between residue i and $i+4$ on the chain, while the non-local are between residues further apart, but not too far apart as these residues have too little inpact on each other to be of any significance. The local potential also separates between backbone-backbone interactions and backbone-side chain interactions. It should be noted that the Fujitsuka potential represents each amino acid by the C, C_α and N on the main chain, and a centroid of the side chain.

V_{HB} is for hydrogen bonding. It is also composed of several parts. The first is for the real hydrogen bonds. It includes an attractive part for the right combination of atoms (carbonyl oxygen and amide hydrogen) and a repulsive part for the wrong combinations. There is also a burial term that depends on how deep the bond is buried in the protein. The second part is a four-body interaction that takes into account the cooperativity of two neighbouring hydrogen bonds. Three different types of cooperativity are considered, α-helices and parallel and anti-parallel β-sheets. The last part is the Born self-energy and it represents the free energy cost upon burial of charged, amide and carbonyl groups.

V_{HP} is for hydrophobic interactions. It measures the buriedness of the C_α and the centroid of the side-chain by looking at the local density of peptide atoms.

The Ramachandran plot is the plot of the relative energy versus the angles ϕ and ψ, i.e. the dihedral angles on both sides of the C_α. These angles have some typical values for α-helices and β-sheets. The V_{Rama} potential makes sure that there are two wells in the Ramachandran plot for the correct values of ϕ and ψ to make sure that there is a tendency to form secondary structure. The potential varies in strength for different amino acids to simulate the likelihood of different amino acids to form different types of secondary structure.

The last part of the potential is $V_{pairwise}$. This takes into account all other attractive interactions between the side chains, like aromatic interactions and charge-charge interactions. For most pairs this potential is Gaussian. The only exception is the interaction between two Alanines which has the form of a Lennard-Jones potential (see next chapter).

2.2 Physical potentials

2.2.1 Common terms

The physical energy functions are different from the knowledge based and statistical energy functions in that they are not primarily based on comparing the chain you wish to study with similar chains with known 3D structure, however it still uses them for calibration. Instead the idea is to estimate all the forces affecting the protein to find its actual energy. The use of these functions are, however, not limited only to proteins. Because of the fact that they often work at atom instead of residue level they can be used straight away to model most biomolecules. The simplicity of most of these functions is necessary because biomolecules can have 100,000 atoms and there is a limit to how much even a supercomputer can calculate.

There are a few terms that are similar for almost all physical energy functionals. First it is the bond energies:

$$E = \sum_{bonds} K_b(b - b_0)^2 + \sum_{angles} K_\theta(\theta - \theta_0)^2 + \tag{11}$$

$$\sum_{dihedral} K_\chi(1 + \cos(n\chi - \delta)) + \sum_{impropers} K_{imp}(\phi - \phi_0)^2$$

where K_b, K_θ, K_χ and K_{imp} are the force constants, b is the bond length, θ is the valence angle, χ is the dihedral angle, n is the multiplicity, δ is the phase angle, ϕ is the improper angle and b_0, θ_0 and ϕ_0 are the equilibrium values.

These terms treat the elastic energy necessary to stretch, bend and twist the bonds between atoms. What could be noted is that this is a first approximation and that there are higher order expansions that mixes the terms. The energy functions that takes this into account are called class 2 energy functions while this is a class 1 energy function. The increased accuracy of class 2 functions is especially useful at geometries far away from the minimum energy. In general class 1 energies work well for simulations at close to room temperature. There are also some other options such as the use of a Morse function for bonds, which allows for bond breaking, or a cosine based term for the valence angle.

The bond energies takes care of the interactions between bonded atoms. There are however other interactions like:

$$\sum_{nonbond} \left(\varepsilon_{ij} \left[\left(\frac{R_{min,ij}}{r_{ij}} \right)^{12} - \left(\frac{R_{min,ij}}{r_{ij}} \right)^{6} \right] \right) + \frac{q_i q_j}{\varepsilon r_{ij}} \tag{12}$$

where the first term is a Lennard-Jones (LJ) potential where ε_{ij} is the LJ well depth, $R_{min,ij}$ is the minimum interaction radius and r_{ij} is the distance between atoms i and j. The LJ term is used to describe the van der Waals interactions. $R_{min,i}$ and ε_i are typically unique for each atom type and then combined using special combining rules to form $R_{min,ij}$ and ε_{ij}. Typically the 1,2 and 1,3 (i.e. atoms bonded to each other or separated by 2 covalent bonds) interactions

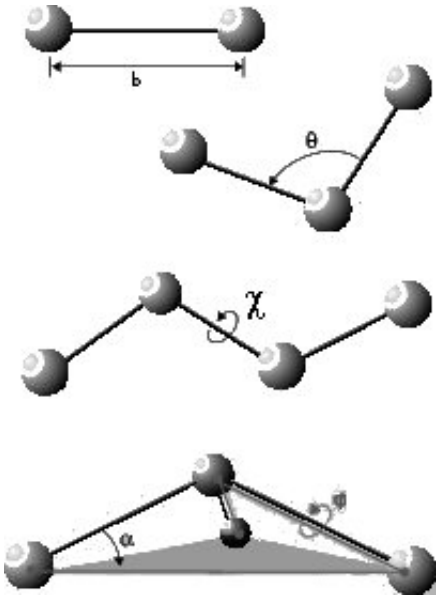

Figure 7: Definitions of the different angles.

are neglected. For computational purposes the long range interactions are also neglected if the interaction distance is too large. There are some other form of this potential. Their purpose is mainly to make the repulsive potential wall associated with Pauli exclusion a bit softer. Examples of other forms include the Buckingham potential that uses an exponential term to treat repulsion, a buffered 14-7 term, instead of the usual 12-6, and the simple change of the r^{12} term into an r^9 term. However, just like the bond energies the LJ potential seems to work fine for simulations close to room temperature.

The second term is a Coulomb potential where q_i and q_j are the partial atomic charges and ε is the dielectric constant, usually set to 1. This term, together with the LJ term, can treat hydrogen bonding reasonably well. However, what this energy function can not treat explicitly is electronic polarizability. Instead this is included by choosing partial atomic charges that overestimate molecular dipoles. A future improvement of this theory would be to include an explicit treatment of polarizability.

2.2.2 Commonly used force fields

So far, there has only been talk about energy functions and potentials. One more thing is however necessary to form a proper force field, the set of parameters. The regularly used force fields are accompanied by hundreds of parameters. All of these has to be acquired by calibrating the force field on a previously known structure.

The most commonly used force fields are the Assisted Model Building and

Energy Refinement (AMBER), the Optimized Potential for Liquid Simulations (OPLS) and Chemistry at HARvard Macromolecular Mechanics (CHARMM). These force field are constantly updated with new parameter sets. There is also different parameter sets depending on how to treat the solvent, and if the force field is all atom or united atom. All atom force fields obviously treats all atoms while united atom force fields include hydrogen atoms next to carbon atoms into the carbon parameters. This is used to save some computation time.

The energy functionals looks very similar. First for AMBER:

$$\begin{aligned} V = &\textstyle\sum_{bonds} \frac{K_b}{2}(b-b_0)^2 + \sum_{angles} \frac{K_\theta}{2}(\theta-\theta_0)^2 \\ &+ \textstyle\sum_{dihedral} \frac{K_\chi}{2}(1+\cos(n\chi-\delta)) \\ &+ \textstyle\sum_{j=1}^{N-1}\sum_{i=j+1}^{N}\left(4\varepsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12}-\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right]\right)+\frac{q_iq_j}{4\pi\varepsilon r_{ij}} \end{aligned} \tag{13}$$

for OPLS:

$$\begin{aligned} V = &\textstyle\sum_{bonds} K_b(b-b_0)^2 + \sum_{angles} K_\theta(\theta-\theta_0)^2 + \\ &+ \textstyle\sum_{dihedrals} V_{1,i}\left[\frac{1+cos(\phi_i)}{2}\right] + V_{2,i}\left[\frac{1+cos(2\phi_i)}{2}\right] + V_{3,i}\left[\frac{1+cos(3\phi_i)}{2}\right] \\ &+ \textstyle\sum_{j=1}^{N-1}\sum_{i=j+1}^{N}\left\{4\varepsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12}-\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right]+\frac{q_iq_je^2}{r_{ij}}\right\}f_{i,j} \end{aligned} \tag{14}$$

and for CHARMM:

$$\begin{aligned} V = &\textstyle\sum_{bonds} K_b(b-b_0)^2 + \sum_{UB} K_{UB}(S-S_0)^2 + \sum_{angles} K_\theta(\theta-\theta_0)^2 \\ &+ \textstyle\sum_{dihedral} K_\chi(1+\cos(n\chi-\delta)) + \sum_{impropers} K_{imp}(\phi-\phi_0)^2 \\ &+ \textstyle\sum_{j=1}^{N-1}\sum_{i=j+1}^{N}\left(\varepsilon\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12}-\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6}\right]\right)+\frac{q_iq_j}{c_1r_{ij}} \end{aligned} \tag{15}$$

Most of the terms are easily recognizable from the last section but there are a few that might need further explanation. In the OPLS equation, f_{ij} is a scaling factor, where $f_{ij} = 0.5$ for 1,4 interactions and 1 for all the rest. In the CHARMM equation S is the Urey-Bradley 1,3- distance, i.e. the distance between two atoms separated by two covalent bonds.

3 Continuum models

3.1 The future

As our database of known proteins grows, the knowledge based force fields will get better and better but it is still obvious that they are not the way forward. The physical force fields seems like a much better starting point for the force fields of the future as they have been the standard tool for the last 30 years, but they also have a serious flaw. They need a huge amount of parameters and if you want to increase the precision you have to expand the terms leading to even more parameters needing to be determined. Remembering that a protein is a long chain of amino acids where each bond is a very small part of the chain, it should be possible to let the bond length tend to zero and treat the chain as a continuous curve. Not only is this method good because we can then use the tools from differential geometry on the curve, but also instead of defining the curve with hundreds of angles and bond lengths we can now define it with just two variables: curvature, $\kappa(s)$, and torsion, $\tau(s)$.

3.2 The Frenet frame

The easiest way to describe a curve in space is by using a Frenet frame. A Frenet frame is a moving reference frame along a curve $R(s)$. The frame is defined by three orthonormal vectors $T(s)$, $N(s)$ and $B(s)$; the tangent, normal and binormal vectors respectively. If the curve is unit-speed, that is $|R'(s)| = 1$ then the tangent vector is defined as the derivative of the curve, $R'(s)$, where ' always stands for derivation with respect to s. Now we can define $N(s)$ as $T'(s)/||T'(s)||$. T and N spans the osculating plane, which is the plane that the curve seems to be residing in in a close vicinity of s. A planar vector always stays in this plane for all s. The third vector, B, which points out of the osculating plane is defined as $B(s) = T(s) \times N(s)$. The three vectors T, N and B now forms an orthonormal basis. What is left now is the definitions of curvature and torsion. We have already seen the curvature term in the definition of N since $\kappa(s) = ||T'(s)||$ or $T'(s) = \kappa(s)N(s)$ so the curvature can be seen as the factor that decides how much the curve will bend in the osculating plane. The torsion can be found from $B'(s) = -\tau(s)N(s)$ and can be seen as the factor that decides how the curve twists out of the osculating plane. These relations can be summarized in the Frenet-Serret theorem, here displayed in matrix-form:

$$\begin{pmatrix} T' \\ N' \\ B' \end{pmatrix} = \begin{pmatrix} 0 & \kappa & 0 \\ -\kappa & 0 & \tau \\ 0 & -\tau & 0 \end{pmatrix} \begin{pmatrix} T \\ N \\ B \end{pmatrix} \tag{16}$$

3.3 Energy terms

Now that we have a way of describing a curve in space, there are some other constraints we would like to put on that curve. Remember the earlier physical energy functions. They all contained energy terms for the elastic bending,

twisting and stretching of bonds. What is needed is an energy term that makes sure that the curve does not bend or twist to sharply and since the bending and twisting of the curve is governed by κ and τ, those are the variables needing to be minimized. The common way of doing this is by minimizing this energy:

$$E = \int_0^L |\kappa|^2 + |\tau|^2 \, ds \tag{17}$$

where ds is integrated over the length of the curve. Of course, for a real protein model there has to be more energy terms than this since optimization using only this energy will only give a straight line.

3.4 Thick chain model

What has been described here is an infinitely thin curve in space. Known as the worm-like chain (WLC) model, this has been used for polymers like DNA, but for proteins more work has been done on the thick chain model. The main difference between the two is that, while the WLC model can be described as the continuum limit of a chain of spheres with decreasing radius, the curve in the thick chain model has a non-zero thickness and can instead be seen as a chain of coins with decreasing distance, forming a tube.

The thick chain model requires some different constraints. It is not enough that the curve behaves nicely and does not make any quick turns. We also need to make sure that it does not overlap. For this it is not enough with a simple two-body potential. It is possible to find a circle passing through any three points in space. By limiting the allowed radius of this circle it is possible to limit the bending of the curve while at the same time it makes sure that segments far away from each other along the curve does not end up too close to each other when it bends.

While this constraint may be easy to explain and draw, it is not that easy to find an intuitive mathematical description of it. A formula for the radius of a circle through any three points is given by:

$$R(\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3) = \frac{|\mathbf{r}_2 - \mathbf{r}_1| \, |\mathbf{r}_3 - \mathbf{r}_1| \, |\mathbf{r}_3 - \mathbf{r}_2|}{4A(\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3)} \tag{18}$$

where $A(\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3)$ is the area of the triangle formed by the three points. The goal is now to prevent this value from getting smaller than a certain value Δ. This is done by simply awarding the curve an infinite amount of energy if that is the case. The full energy function is:

$$\int_0^L \int_0^L \int_0^L V(R(r(s), r(s'), r(s''))) ds ds' ds'' + \int_0^L |\tau|^2 \, ds \tag{19}$$

$$V(R) = \begin{cases} 0 & R \geq \Delta \\ \infty & R < \Delta \end{cases} \tag{20}$$

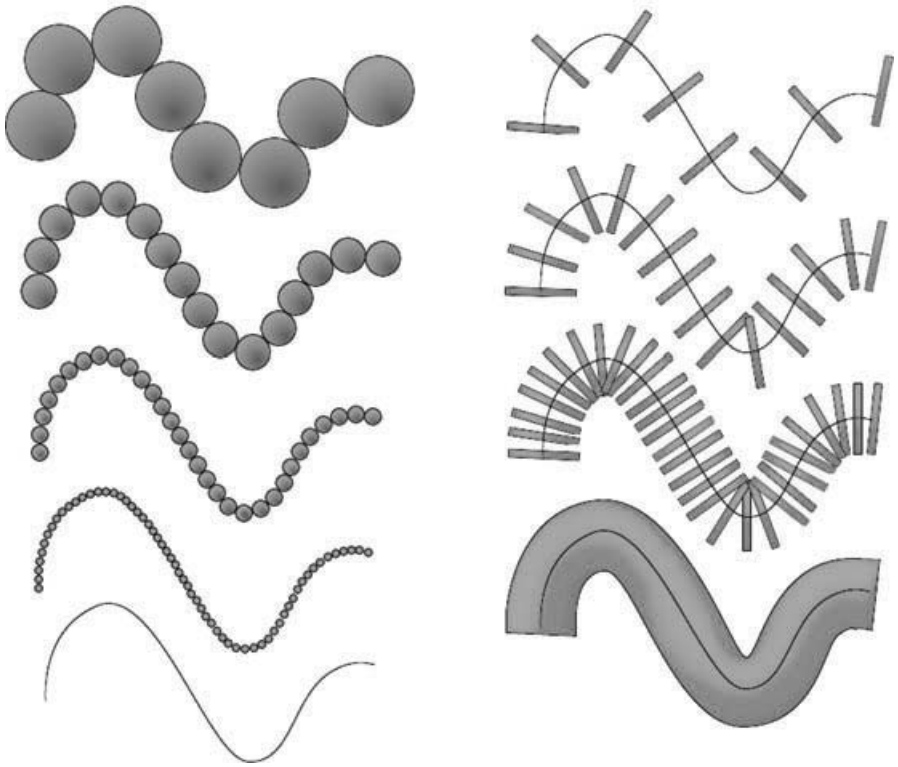

Figure 8: The different continuum limits of a sequence of spheres with diminishing radii and a sequence of discs with diminishing distance.

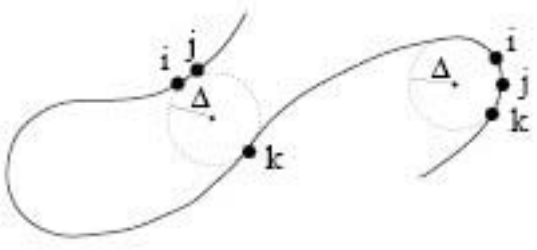

Figure 9: It is possible to create a circle through any three points in space.

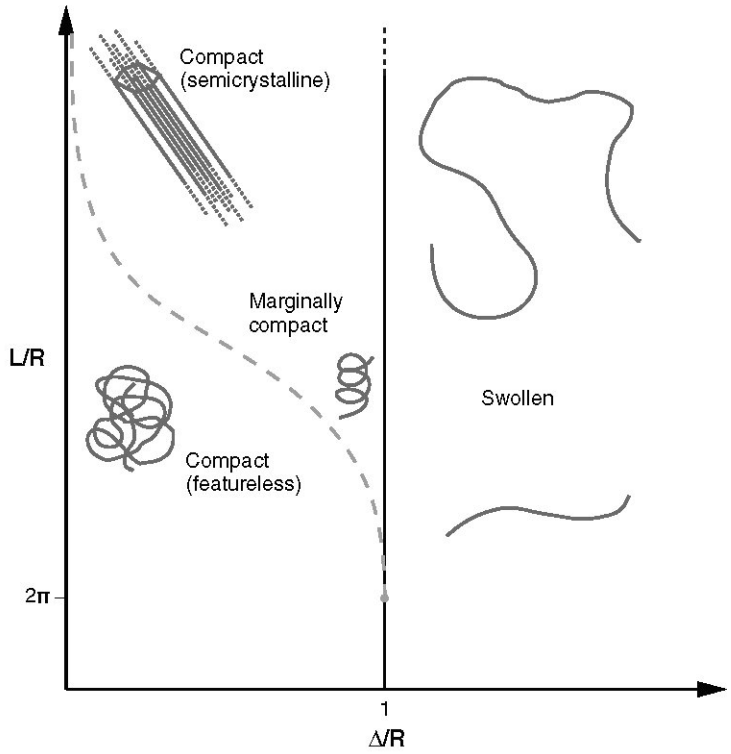

Figure 10: Phase diagram for a self-attracting tube with length L, radius Δ, and a range of interaction R.

There is also the possibility to have a small negative part for R slightly bigger than Δ, to get a small attraction for interactions like hydrogen bonds.

So far we have only discussed the constraints on the chain. We have to remember that protein folding is much about attractive forces. This whole tube will be self-attracting, mainly because of hydrophobic packing, that is, it wants to have as little area as possible exposed to the solvent outside. For a theoretical tube the result of this attractive force depends on the range of the attractive interaction, R. For example, if the total length of the tube is long compared to the interaction distance while the tube thickness is short then the tubes form semi-crystalline structures.

The interesting part is the marginally compact phase is when $R \approx \Delta$ and $L \approx 2\pi R$. There the tubes tend to form a few well recognizable structures like helices and sheets. The similarity between these and the secondary structure of proteins is remarkable.

4 Generating a curve

4.1 Constructing the curve

The last part of this project was to construct a Maple function that could, from a given curvature and torsion, generate the corresponding curve. I wanted the function to be able to handle really general curvature and torsion and not limit myself to, say, differentiable functions. The main reason for this is, except the generality, the ability to optimize the curve by making small changes to the curvature and torsion. Hence they are defined as a vector with as many values as needed. For this to work you also need to specify δ, the distance along the curve between any two values. The last input parameter to be specified for the function is simply the number of values used.

The curve always starts in the origin with the T, N and B vectors in the x, y, z direction respectively. This is not a restriction in any way since you can always rotate and translate the curve to transfer it where you want it. What then happens is that the function takes a step of length δ along the T vector. There it uses the new values for κ and τ to generate the Frenet frame at that point. Then it takes another step of length δ along the T vector and so it continues. For each step it saves the position vector in the matrix R, which coincidentally is the output of the whole function. This R matrix can then be plotted to see what the curve looks like.

Here is the full function:

CreateCurve:=proc(k, τ, δ, L) local t, n,b,r,R,i,$t1$,$n1$; uses LinearAlgebra;
$t := Vector[1,0,0]; n := Vector([0,1,0]); b := Vector([0,0,1])$
$r := Vector([0,0,0]); R := Matrix(3, L)$
for i from 1 to L do
$R[1,i] := r[1]; R[2,i] := r[2]; R[3,i] := r[3];$
$t1 := t + k[i] \cdot n \cdot \delta;$
$n1 := n + (-k[i] \cdot t + \tau[i] \cdot b) \cdot \delta;$
$t := \frac{t1}{\sqrt{t1 \cdot t1}};$
$n := \frac{(n1-((n1.t)t))}{\sqrt{(n1-(n1.t)t)\cdot(n1-(n1.t)t)}};$
$b := CrossProduct(t, n);$
$r := r + \delta \cdot t;$
end do;
R
end proc

To show what it can do I have made some test runs (fig. 11-14). The first two images are of functions with known curvature and torsion and their presence are mainly to show that the function works. To show the effects of curvature and torsion I have made two random curves with different limits for the curvature and torsion, with all the other variables equal. They could be the starting point for an algorithm to simulate proteins by adding forces between parts of the curve and using an algorithm to optimize the shape.

These are just random curves but it could be interesting to compare them with some real proteins anyway. Fig. 15 shows the backbone of a protein from the protein database and fig. 1 shows a more complete visualization of a protein. If the backbone model had been a bit more smooth instead of drawing straight lines between the residues it could easily have been mistaken for a randomly generated curve made with my program (compare with fig. 13). This indicates that this method of defining curves by their curvature and torsion is a viable way of describing proteins. However, this comparison should not be taken too far. It is important to remember that the curves are indeed completly random and, unlike the proteins, have no primary, secondary or tertiary structure. Most noticeable is the lack of a well-packed tertiary structure.

4.2 Random values

The original CreateCurve function can accept any values for κ and τ. For the first experiments I simply used random values between a minimum and a maximum value with equal probability for each value in that interval. However, a more realistic first approximation of a protein would be that there is a preferred value where the probability is highest and a decreasing probability the further away you get. A reasonable choice is the Gaussian distribution:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} \tag{21}$$

where $P(x)$ is the probability of the value x, μ is the mean value and σ is the standard deviation. For the purpose of generating random numbers with this distribution I have constructed the following functions:

```
GetGaussian := proc (MeanMinimum)
    local a, b, c, d, e, f, randbool; uses RandomTools;
  randbool := rand(1..2);
  c := 0;
  while 1 ≤ c or c ≤ 0 do
    a := Generate(float(range = 1..2)) − 1;
    b := Generate(float(range = 1..2)) − 1;
    c := a ∗ a + b ∗ b;
  end do;
  d := √(−2 ∗ ln(c)/c);
  e := −a ∗ d; f := b ∗ d;
  if MeanMinimum = 1 then
    f
  else
    if randbool() = 1 then
      e
    else
      f
    end if
```

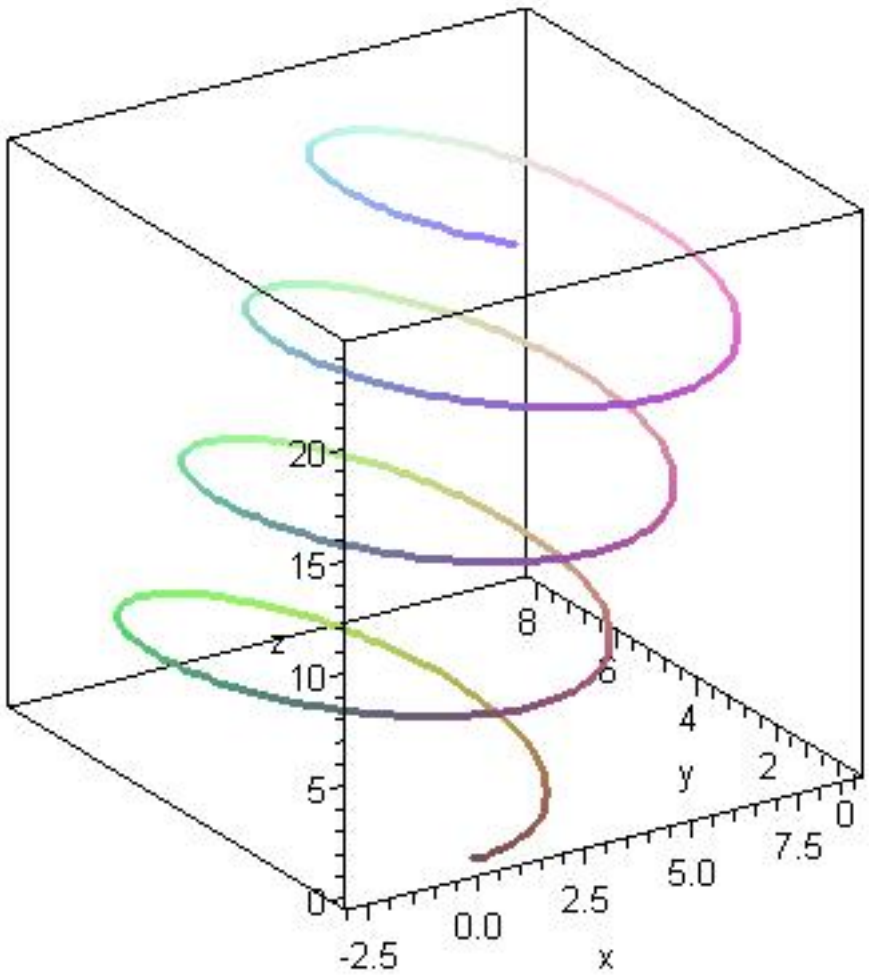

Figure 11: Helix with the following equation (4cos(t),4sin(t),t) Created using only the curvature $\kappa = \frac{4}{17}$ and $\tau = \frac{1}{17}$. Observe that the helix is not supposed to go straight up because of the original definition that the z-direction is peripendicular to the curve at its starting point.

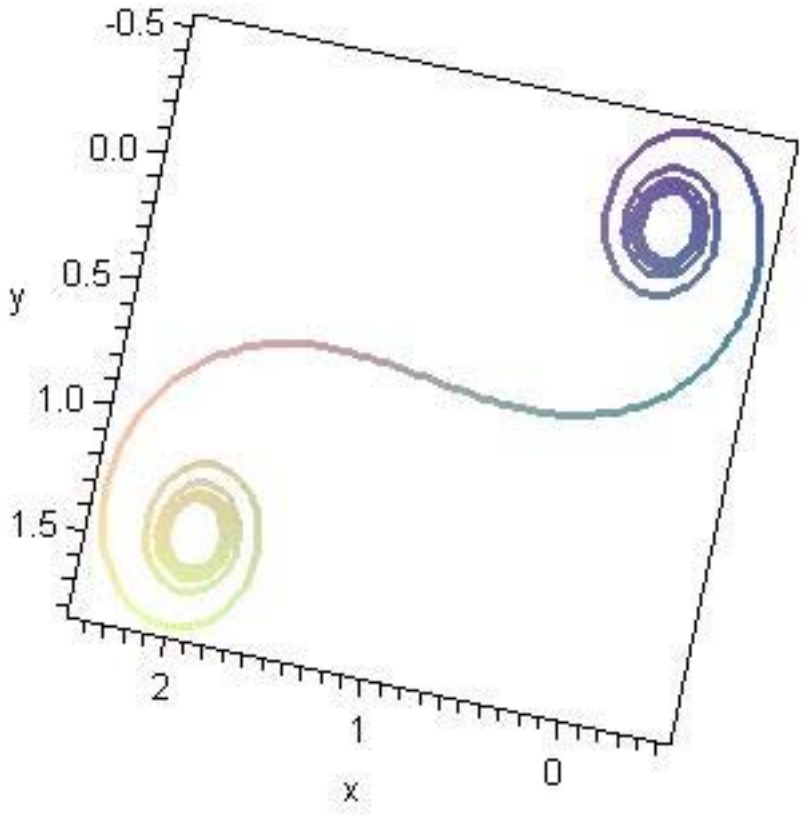

Figure 12: A plane curve with $\kappa(s) = s$ for $s = [-8, 8]$ using 500 points with the appropriate distance between them.

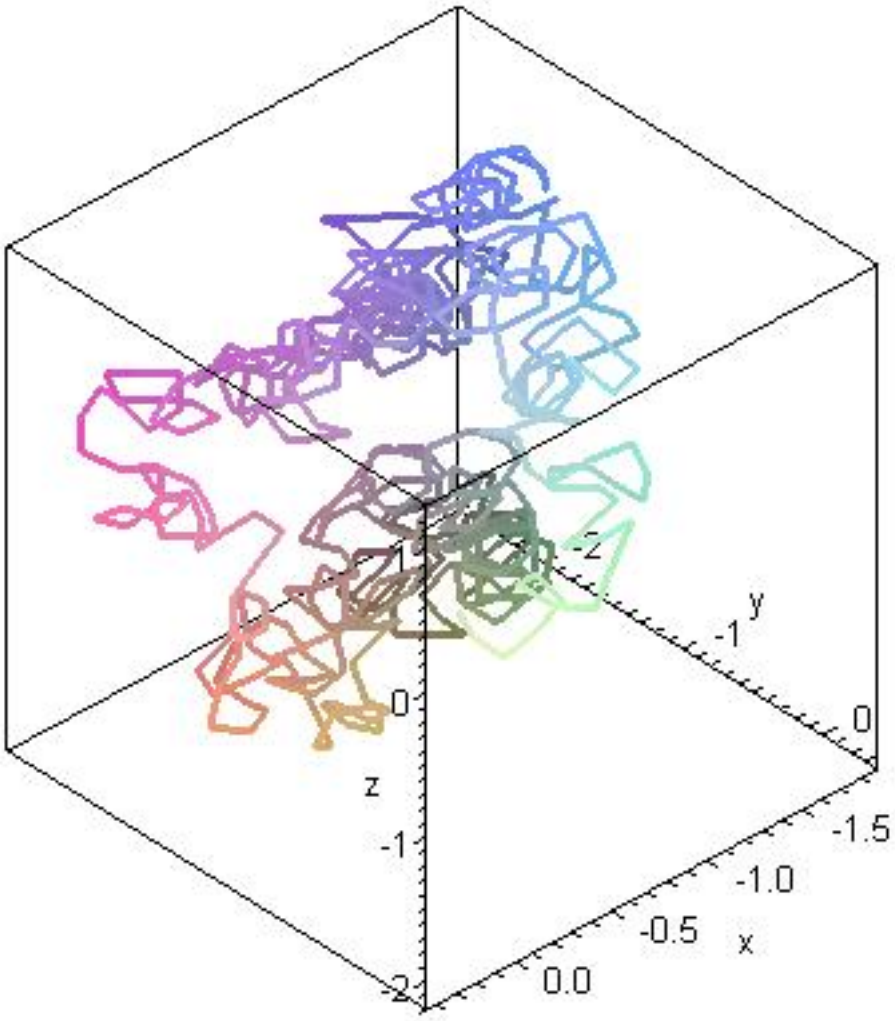

Figure 13: A randomly generated curve with $\kappa \in [0, 30]$ and $\tau \in [-30, 30]$. $\delta = 0.2$, $L = 500$

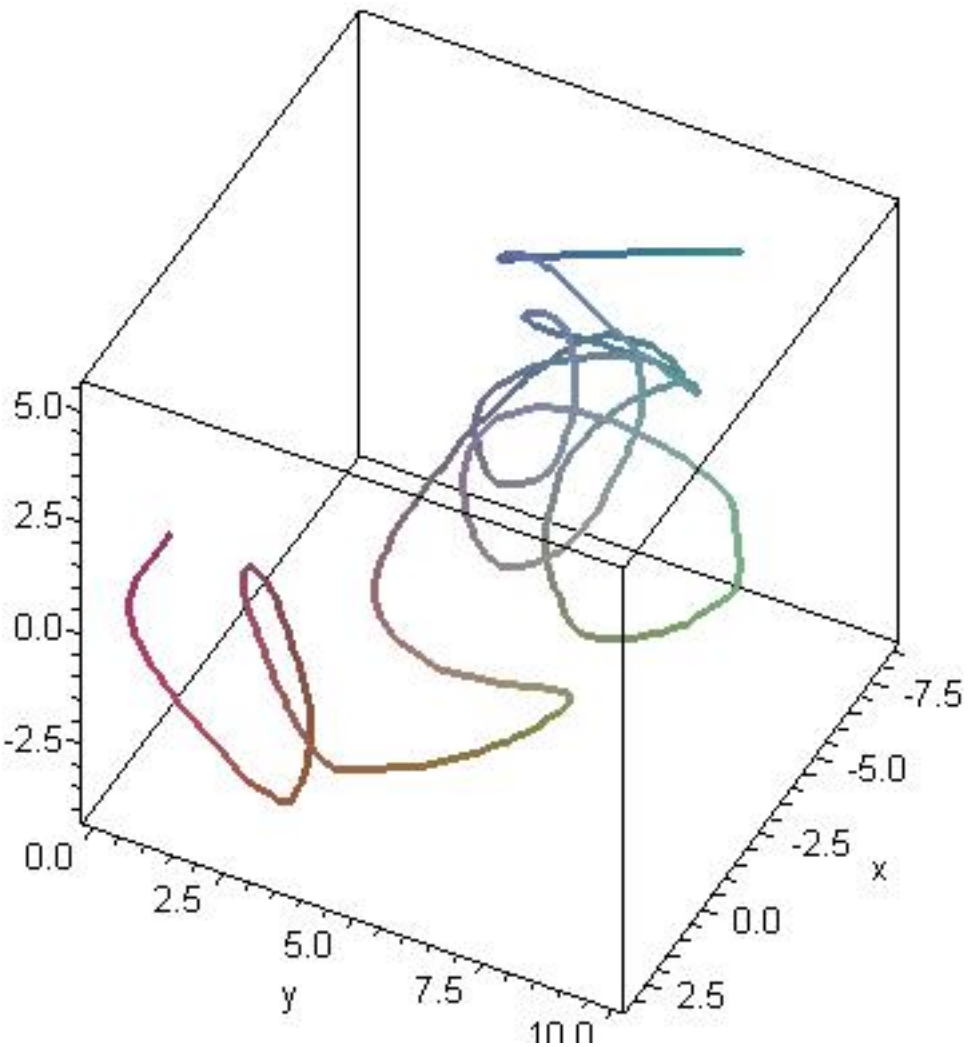

Figure 14: A randomly generated curve with $\kappa \in [0,1]$ and $\tau \in [-1,1]$. $\delta = 0.2$, $L = 500$

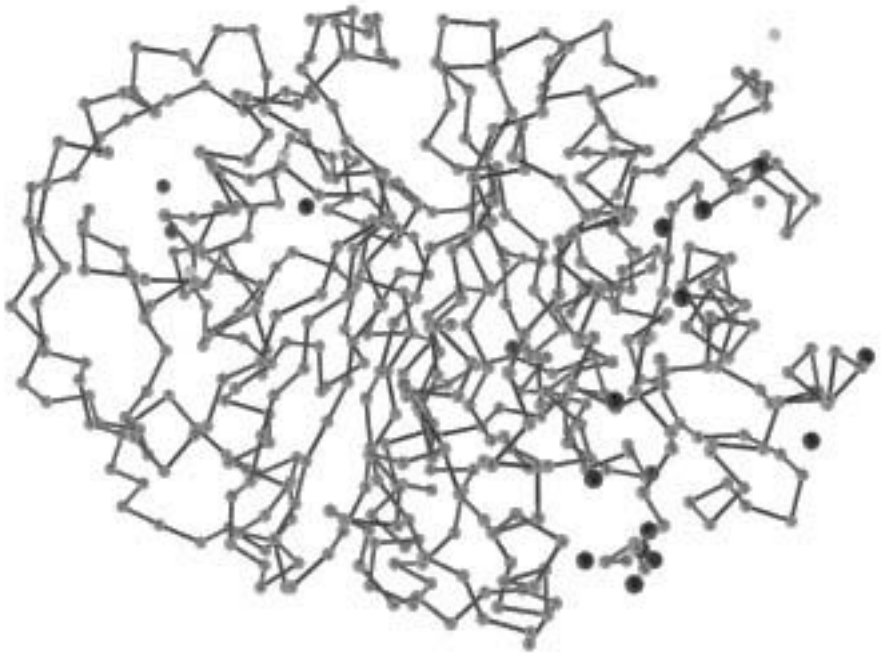

Figure 15: The backbone atoms of a protein from the protein database.

```
    end if
end proc

GetRand := proc (Mean, Deviation, MeanMinimum) local R;
    R := Mean + Deviation * GetGaussian(MeanMinimum);
end proc
```

The GetGaussian function generates random numbers that have a Gaussian distribution with $\mu = 0$ and $\sigma = 1$. The GetRand function then adjusts this value to the desired μ and σ. I have also included the possibility of only using the part of the distribution greater than the mean value, by setting $MeanMinimum = 1$. For $\mu = 0$ this means that only positive values are allowed.

One reason for making the function generate one random number at a time instead of list of a specified length is that now it is possible to change the desired mean and standard deviation values individually. This could for example be used to adjust the parameters depending on the last generated value, or, for protein modelling, adjust the parameters depending on the residue.

4.3 Optimizing the curve

The first thing needed for an optimizing algorithm is an energy function to measure how good the current curve is. I want to build a function to straighten out a curve. For this I need an energy function to measure how much the curve bends. Fortunately I have already defined equation 17, which has all the right properties. The following is a Maple function for calculating the energy of a curve. The function simply takes the curve, as defined earlier, as input and outputs the energy.

```
Energy := proc (k, τ, L) local En, i;
    En := 0;
    for i to L do
        En := En + k[i]^2 + τ[i]^2
    end do;
    En
end proc
```

Now that the energy function is well defined, the last remaining part is the actual optimization algorithm. The algorithm is rather simple. It makes a copy of the curve and changes one of the angles by generating new values for the curvature and torsion at that point. Then it checks to see if the energy of the new curve is lower than that of the original one. If that is the case then it returns the new curve, otherwise it returns the old one. The input parameters are easy to recognize as the normal definition of a curve, from above, and the parameters for the GetRand function, with the added possibility of defining different parameters for curvature and torsion. The Maple version of the algorithm looks like this:

```
Alg := proc (k,τ, L, meank, meant, deviationk, deviationt)
 local RandRes, i, Res, k2, tau2;
   RandRes := rand(1..L);
   k2 := Vector(L, k);
   tau2 := Vector(L, τ);
   Res := RandRes();
   k2[Res] := GetRand(meank, deviationk, 0);
   tau2[Res] := GetRand(meant, deviationt, 1);
   if E(k2, tau2, L) < E(k, τ, L) then
      k[Res] := k2[Res];
      τ[Res] := tau2[Res];
   end if;
   k, τ
end proc
```

Figure 16 and 17 shows a curve before and after it has gone through the optimization algorithm 10000 times. Figure 18 shows the variation in curve energy during this process.

It is easy to see that the function could be made to run much faster. Because of the current definition of the energy of the curve, the difference in energy between the two curves depend entirely on the particular point where the curves differ. It would have been much more efficient to just look at that point instead of summing over the entire curve. My reason for writing it this way is that now the optimization algorithm and the energy are entirely separate functions and if I wanted to change the energy function this could easily be done without worrying about anything else.

5 Conclusion

The complete understanding of how a protein folds would truly be a great achievment. Based on this survey I would say that we are still far away from that goal. The knowledge-based potentials are for many reasons not a way forward, mainly because they demand a large database of known 3D-structures and they does not give any information about the folding process. The physical force field is the best method used today, it has been used for decades and will probably still be the dominating method for years to come. The question is if it is closing in on the limit for what it can do, when every step towards increased accurracy leads to a much greater step backwards in terms of decreased performance and increased need for calibration? Continuum methods are an interesting new way forward that has had some success in the study of DNA. It solves some of the problem of the physical force fields by requiring a much lesser amount of constants to describe the main chain. However this method is still under development. It has shown some promise in the simulation of secondary structure, but it is not yet known how good it can simulate an entire protein with all its chemical properties.

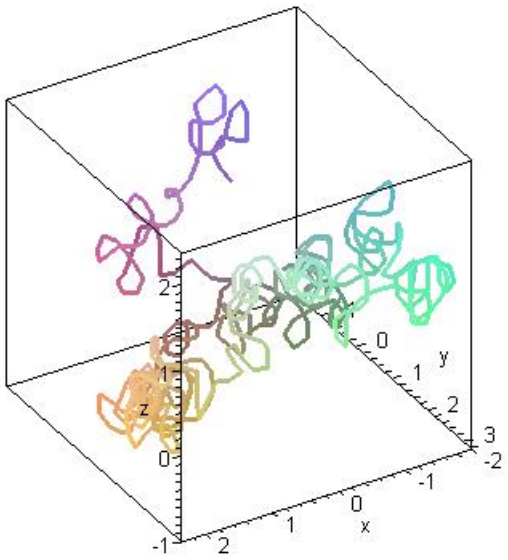

Figure 16: Random curve generated with GetRand function, $\mu = 0$, $\sigma = 10$, $\delta = 0.2$, $L = 500$.

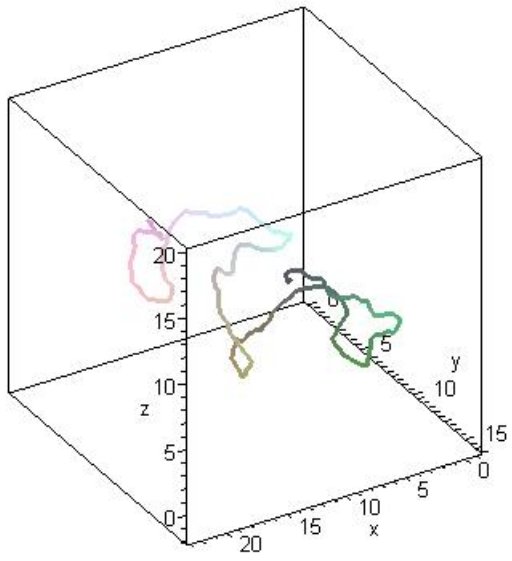

Figure 17: The same curve after 10000 iterations of the straightening algorithm.

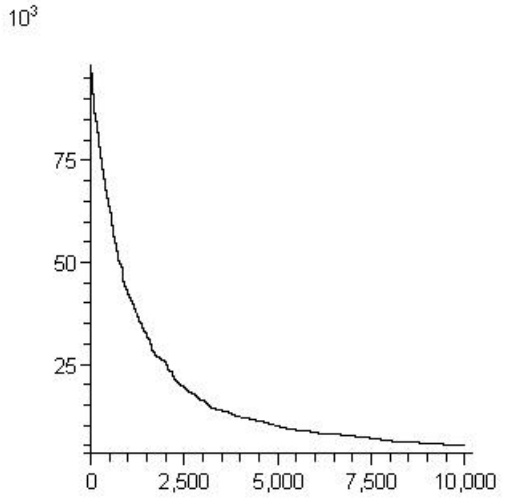

Figure 18: The energy of the curve for each iteration.

References

[1] Alexander, D. and Mackerrell, Jr. "Empirical Force Fields for Biological Macromolecules: Overview and Issues" J. Comp. Chem. 25:1584-1604 (2004).

[2] Arai, M and Kuwajima, K. "Role of the molten globule state in protein folding" Adv. Protein Chem. 53:209-282 (2000)

[3] Banavar, Jayanth R. et. al. "Geometry and Physics of Proteins" Proteins 47:315-322 (2002).

[4] Banavar, Jayanth R. et. al. "Unified perspective on proteins: A physics approach" Physical Review E 70, 041905 (2004).

[5] Clark, Jim. "The Structure of Proteins" 2004. http://www.chemguide.co.uk/organicprops/aminoacids/proteinstruct.html, accesed July 2007.

[6] Ewig, Carl S. "Derivation of Class II Force Fields. VII. Derivation of a General Quantum Mechanical Force Field for Organic Compounds" J. Comp. Chem. 22, 15, 1782-1800 (2001).

[7] Fujitsuka, Yoshimi et. al. "Optimizing Physical Energy Functions for Protein Folding" Proteins 54:88-103 (2004)

[8] Heindlich, M. et. al. "Identification of Native Protein Folds Amongst a Large Number of Incorrect Models" J. Mol. Biol. 216:167-180 (1990)

[9] Klingenberg, Wilhelm. "A Course in Differential Geometry" Springer, 1978.

[10] Kollman Peter; Duan, Yong and Wang, Lu. "Watching a Protein Fold" 1998 http://www.psc.edu/science/kollman98.html, accesed July 2007.

[11] Krishnamoorthy, Bala and Tropsha, Alexander. "Development of a four-body statistical pseudo-potential to discriminate native from non-native protein conformations" Bioinformatics 19, 12:1540-1548 (2003)

[12] MacKerrell, A. D. et. al. "All-Atom Empirical Potential for Molecular Modeling and Dynamics Studies of Proteins" J. Phys. Chem. B. 102:3586-3616 (1998)

[13] Marenduzzo, D et. al. "Continuum model for polymers with finite thickness" J. Phys. A 38, 17, 277-283 (2005).

[14] McCleary, John. "Geometry from a Differentiable Viewpoint" Cambridge: Cambridge university press, 1994.

[15] McDonald, Nora A. and Jorgensen, William L. "Development of an All-Atom Force Field for Heterocycles. Properties of Liquid Pyrrole, Furan, Diazoles, and Oxazoles." J. Phys. Chem. B. 102:8049-8059 (1998)

[16] Onuchic, J. N. et. al. "Theory of protein folding: The energy landscape perspective" Ann. Rev. Phys. Chem. 48:545-600 (1997)

[17] Plotkin, Steven S. and Onuchic J. N. "Understanding protein folding with energy landsacpe theory Part 1: Basic concepts" Quart. Rev. Biophys. 35, 2 (2002)

[18] Sippl, Manfred J. "Calculation of Conformational Ensembles from Potentials of Mean Force" J. Mol. Biol. 213:859-883 (1990)

[19] Thomasson, W. A. "Unraveling the Mystery of Protein Folding" http://opa.faseb.org/pdf/protfold.pdf, accesed July 2007.

[20] Yee, David C. "Introduction to protein folding ? The process and factors involved" March 1998 http://www.proteindesign.com/Sections-index-req-viewarticle-artid-1-page-1.html, accesed July 2007.