The Physics & Biology of Protein Folding
But first, just what is a protein?
By understanding folding and protein conformational change scientists hope to be able to design mediation strategies to fight such diseases as Alzheimer's, Parkinson’s that are due to protein misfolding. Once you understand the physics of protein folding one can design drugs to cure diseases.
In order to understand more about proteins, it is necessary to understand protein structure.
Amino Acid Structure

![Amino Acid Structure Diagram]

- **Amino Group**
- **Side Chain**
- **Carboxylic Acid Group**
- **α - carbon**

![Carboxylic Acid Group Diagram]

- **Central Carbon Atom**
- **Amino Group**
- **Side Chain**
Nonpolar and uncharged. Insoluble in water. Tend to associate with other like molecules to avoid water.
Polar or charged. Water soluble. Form hydrogen bonds with other hydrophilic side chains.
May form interfaces between hydrophobic and hydrophilic molecules.
Amino Acids Combine

\[
\begin{align*}
\text{HO}_2\text{C} = \text{O} & \quad + \quad \text{HO}_2\text{C} = \text{O} \\ 
\text{C} - \text{N} - \text{H} & \quad \rightarrow \quad \text{C} - \text{N} - \text{H} \\
\text{H} & \quad \text{R} \\ 
\text{H} & \quad \text{R}
\end{align*}
\]

peptide bond

\[
\begin{align*}
\text{HO}_2\text{C} = \text{O} & \quad + \quad \text{HO}_2\text{C} = \text{O} \\ 
\text{C} - \text{N} - \text{H} & \quad \rightarrow \quad \text{C} - \text{N} - \text{H} \\
\text{H} & \quad \text{R} \\ 
\text{H} & \quad \text{R}
\end{align*}
\]

peptide bond
Amino Acids Combine

Leucine

Alanine

Glutamate
Protein Structure

Primary Structure:

Amino Acids combine to form these long linear sequences. Due to a variety of factors these chains do not remain linear.
Protein Structure

Secondary Structure: Alpha Helix

Hydrogen bonds form when a hydrogen atom is covalently bonded to one electronegative atom (usually oxygen or nitrogen) and is simultaneously attracted to another electronegative atom. This attraction is the hydrogen bond (shown as dashed lines below)

Copyright © Pearson Education, Inc., publishing as Benjamin Cummings.
alpha helix

hydrogen bond

peptide bond
Protein Structure

Secondary Structure: Beta Sheet
Protein Structure

Tertiary Structure: What has happened?
Image generated using VMD software and RCSB Data Bank.
Protein folding problem:
“Predicting 3-dimensional structure from sequence”

- A unique folded structure (native conformation, native fold) is assumed by a given sequence, although infinitely many conformations can be accessed.
- Which? (Protein folding problem)
- How, why? (Folding kinetics)

Basic postulate:
Thermodynamic equilibrium $\rightarrow$ Global energy minimum

Courtesy Dr. Banu Ozkan
Why is it hard?
- many possible conformations for the protein
- many may have similar energies
- calculated energies are estimates
- hard to tell the correct structure

Courtesy Dr. Banu Ozkan
DESIGN

Why is it hard?
• many possible sequences
• don’t know what structure each sequence adopts
• calculated energies are estimates
• hard to tell the correct structure

Courtesy Dr. Banu Ozkan
Putting it all together

(a) Primary structure

(b) Secondary structure

(c) Tertiary structure

(d) Quaternary structure
The Protein folding Problem

As you can now see the question of protein folding is a huge question.
A protein’s function is related to its:
Amino acid sequence (primary structure).
How sequences of amino are linked (secondary structure).
How the proteins are folded (tertiary structure).
How proteins combine with other proteins (quaternary structure).

Why do some amino acid sequences form helixes and some sheets?

How do proteins fold? Scientists feel that they have a pretty good handle on the underlying dynamics behind protein folding. But, they still cannot predict protein conformation based on the dynamics.

Knowing what you know about the chemical structure of proteins and the what we have learned from our study of physics make a list of all possible causes that your group can come up with that might drive the folding of proteins.
Molecular Dynamics (MD):

1. Atoms are classical point-masses that move in a physical potential

   **Bonded terms**
   - Bonds
     - $k_b(r-r_0)^2$
   - Angles
     - $k_a(\theta-\theta_0)^2$
   - Dihedrals
     - $k_d[1+\cos(n\phi-\phi_0)]$
   - Improper Dihedrals
     - $k_i(\omega-\omega_0)^2$

   **Non-bonded terms**
   - Electrostatics
     - $(q_1q_2)/(\varepsilon r)$
   - van der Waals
     - $E[(r_m/r)^{12}-(r_m/r)^6]$

2. The potential is fitted to experiments and quantum mechanical calculations

3. The potential is transferable

4. Atoms are propagated by classical (Newtonian) dynamics

5. Thermodynamic data is obtained using statistical mechanics

*Courtesy Dr. Arjan van der Vaart*
TOTAL POTENTIAL ENERGY

\[ U = \sum \frac{1}{2} K_b (b - b_0)^2 + \sum \frac{1}{2} K_\theta (\theta - \theta_0)^2 \]

All Bonds  All Angles

\[ + \sum K_\phi \left[ 1 - \cos(n\phi + \delta) \right] \]

All Torsion Angles

\[ + \sum \epsilon \left[ \left( \frac{r}{r_0} \right)^6 - 2 \left( \frac{r}{r_0} \right)^4 \right] \]

All nonbonded pairs

\[ + \sum 332 q_i q_j / r \]

All partial charges

ENCAD Parameters from 1979 (Lifson)

Courtesy Dr. Banu Ozkan
TOTAL POTENTIAL ENERGY

- The total potential energy or enthalpy fully defines the system, $U$.
- The forces are the gradients of the energy.

The energy is a sum of independent terms for:
- Bonds, Bond angles,
- Torsion angles and non-bonded atom pairs.

Courtesy Dr. Banu Ozkan
Molecular Dynamics Theory

- **Force** = $-\frac{dU}{dx}$ (slope of potential, $U$); acceleration, $m \ a(t) = \text{Force}$.
- All atoms move together so force between atoms change with time.
- Analytical solution for $x(t)$ and $v(t)$ is impossible; numerical solution is trivial.

\[
\begin{align*}
  x(t+\Delta t) &= x(t) + v(t) \Delta t + \left[4a(t)-a(t-\Delta t)\right] \Delta t^2/6 \\
  v(t+\Delta t) &= v(t) + \left[2a(t+\Delta t)+5a(t)-a(t-\Delta t)\right] \Delta t/6
\end{align*}
\]

- **Kinetic energy** $U_{\text{kinetic}} = \frac{1}{2} \sum m_i v_i(t)^2 = \frac{1}{2} n k_B T$
- Total energy ($U_{\text{potential}} + U_{\text{kinetic}}$) must not change with time
- Time step, $\Delta t$, must be very small at 10-15 seconds or 0.001 ps.
The RCSB Protein Data Bank

http://www.rcsb.org/pdb/home/home.do

The Center for Biological Physics

http://biophysics.asu.edu/CBP/
Instructional Goals

1. Connections: Students will be able to identify and explain the integrated nature of modern scientific research:
   English (writing grant proposals), biology, chemistry, physics, mathematics and computer science.

2. Programming: At the conclusion of this unit students will be able to:
   
   Explain what a computer is and how a computer works.

   Describe what a computer program is.

   Identify the 4 main types of computing Languages.

   Explain how computer languages differ.

   Describe how Scripting languages work.
But first, a brief history.

http://www.msu.edu/course/lbs/126/lectures/history.html
2. Programming:
At the conclusion of this unit students will be able to:

Describe what a computer program is.

CH. 1 -3

http://chortle.ccsu.edu/CS151/cs151java.html
2. Programming:
At the conclusion of this unit students will be able to:

Identify the 4 main types of computing Languages.
Ch. 4

http://chortle.ccsu.edu/CS151/cs151java.html
Identify the 4 main types of computing Languages.

**Machine language**

Sometimes referred to as machine code or object code, machine language is a collection of binary digits or bits that the computer reads and interprets. Machine language is the only language a computer is capable of understanding.
Explain how computer languages differ. (From Wikipedia)

**Assembly language**

A program written in assembly language consists of a series of *instructions* mnemonics that correspond to a stream of executable instructions, when translated by an *assembler*, that can be loaded into memory and executed.

For example, an x86/IA-32 processor can execute the following binary instruction as expressed in *machine language*:

**Binary:** 10110000 01100001 (Hexadecimal: 0xb061)

The equivalent assembly language representation is easier to remember (more *mnemonic*):

```
mov al, #061h
```

This instruction means:

*Move the hexadecimal value 61 (97 decimal) into the processor register named "al".*

The mnemonic "mov" represents the opcode 1011 which *moves* the value in the second operand into the register indicated by the first operand. The mnemonic was chosen by the instruction set designer to abbreviate "move", making it easier for the programmer to remember. A comma-separated list of arguments or parameters follows the opcode; this is a typical assembly language statement.
A high-level programming language is a programming language that, in comparison to low-level programming languages, may be more abstract, easier to use, or more portable across platforms. Such languages often abstract away CPU operations such as memory access models and management of scope.

Stereotypically, high-level languages make complex programming simpler, while low-level languages tend to produce more efficient code. In a high-level language, complex elements can be broken up into simpler, though still fairly complex, elements for which the language provides abstractions, keeping programmers from having to "reinvent the wheel." For this reason, code which needs to run particularly quickly and efficiently may be written in a lower-level language, even if a higher-level language would make the coding easier.
Scripting – JavaScript, Python

**Scripting languages**, also called **script languages**, are **programming languages** that control **applications**. Scripts are executed directly from their **source code**, which are generally text files containing language specific markup.[1][2] Thus, "scripts" are often treated as distinct from "**programs**", which execute independently from any other application.

The name "script" is derived from the written script of the **performing arts**, in which dialogue is set down to be spoken by human actors. Early script languages were often called **batch languages** or **job control languages**. Such early scripting languages were created to shorten the traditional edit-compile-link-run process.

**Computer languages** are created for varying purposes and tasks — different kinds and styles of programming. One common programming task is known as **scripting**, or connecting pre-existing components to accomplish a new related task. Those languages which are suited to scripting are typically called **scripting languages**. Many languages for this purpose have common properties: they favor rapid development over efficiency of execution; they are normally implemented with **interpreters** rather than **compilers**; and they are strong at communicating with program components written in other languages.