Computational Evolution of a Model Polymer that Folds to a Specified Target Conformation

Richard Judson
Center for Computational Engineering, MS 9214
Sandia National Laboratories
Livermore, CA 94551-0969
email: rsjuds@ca.sandia.gov

Abstract

A method is described for folding polymers to specific target conformations. The approach uses a fast but approximate dynamics algorithm, coupled with a genetic algorithm that is used to evolve the large number of free parameters needed. The dynamics algorithm uses a *state transition matrix* approach. At each time step, the distances between pairs of atoms are adjusted by shifting them from D_{ij} to $D_{ij}+S_{ij}$ where S_{ij} is an element of the state transition matrix S. Atom pairs that are attractive have $S_{ij} <0$ and pairs that are repulsive have $S_{ij} >0$. The atomic movement is carried out by gradient minimizing the molecular mechanics energy of the molecule subject to harmonic distance constraints. The method is applied to a simple test case, a 19 atom 2-D polymer. The paper also shows that the S matrices can correctly fold a limited variety of initial conformations that differ from the one used during the evolution phase.

1. Introduction

Of all of the difficult approaches to predicting protein structure, one of the most difficult appears to be simulating the entire folding process using molecular dynamics. Most of the currently practical or at least promising structure prediction methods are instead based on finding homologies between a new sequence and that of one or several proteins whose structure is already known.[1-4] However, predicting structure through the dynamical folding process has to be necessary ultimately. Folding happens on a scale so fast compared to the time needed to fully search the protein's conformation space, that the folding process itself acts to select the final state, and therefore carries useful information about the native conformation.

However, the molecular dynamics approach suffers from at least two major difficulties. The first of these is the time scale. Conventional molecular dynamics algorithms take time steps on the order of 10^{-15} sec, while the complete folding process takes between 10^{-3} and 10^{+3} sec. The longest molecular dynamics runs performed to date, for protein sized ensembles, extend to a only a few nsec. To overcome this hurdle, we either need to wait for several years until computer power increases sufficiently, or to devise new approximations that allow longer time steps to be taken. The second problem, which is probably more difficult in the long run, is that of accurately determining force field parameters. We know that today's protein force fields would not predict the correct folding of a protein, even given enough CPU time. This is known from trajectories that have been run for short times starting from native protein conformations in which the proteins tend to wander away from the native state.

In this paper, I present one approach to addressing these two problems and apply it to a simple model polymer. Just as the problem has two parts, so does the proposed solution. First, a simple rule-based dynamics algorithm is described, similar to the method of Feldmann and Rawn[5] which allows large steps to be taken, and hence speeds up the computational folding process.

In the approach used here, atoms feel two sets of forces. One is a standard molecular mechanics (MM) force field that has bonded and short-range non-bonded terms. Current short range force fields appear to be well enough characterized to be valid in folding simulations. The long range interactions are treated in a novel way. The folding process is carried out in a series of steps. At each step, the current distances between pairs of atoms i and j, denoted D_{ij}, are calculated. All of the atoms are then moved in an attempt to adjust these distances to be $D_{ij}+S_{ij}$ where S is the *state transition matrix*. For atom pairs that are attracted to one another, $S_{ij}<0$ and for pairs that repel one another, $S_{ij}>0$. If D_{ij} is greater than some cutoff distance, the movement rule is not enforced. The move, or transition under the influence of S is carried out by performing a gradient minimization of the energy which is the sum of the internal and short range MM force field and a set of harmonic distance constraints. The basic MM force field prevents the long range force field from either tearing the molecule apart or forcing atoms to come unphysically close to one another. Schematically, the process proceeds as

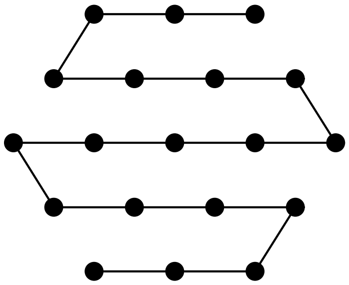
$$D(0) \rightarrow D(1) \approx D(0) + S \rightarrow D(2) \approx D(1) + S \rightarrow \ldots \rightarrow D(T) \approx D(T-1) + S \qquad (1)$$

where the algorithm converges when $D(T) = D(T-1)$, i.e. when the attractive and repulsive pieces of the combined short and long ranges forces balance out. The equalities in Eq. 1 are only approximate because the basic MM force field acts alongside the constraints.

The novel addition presented here is a method for determining appropriate values for elements of the state transition matrix S. I use a genetic algorithm[6] (GA) to evolve S matrices that cause the molecule to fold to a desired target conformation. GAs are optimization methods based on Darwinian evolution. Populations of individuals interact with one another through selection and mating operations to produce individuals that have increasingly higher "fitness". The GA creates many S matrices, and evaluates the difference between the target state and the final state produced by the folding process. It then refines the set of S matrices and repeats. Over a period of many generations the GA eventually finds one or more *S* matrices that correctly fold the molecule. The measure of fitness is the RMS deviation between the internal distance matrices of the target conformation and the conformation produced by the folding process, denoted $f(S) = \left\| D_{Target} - D(T,S) \right\|$. GAs have been used for a wide variety of global optimization problems ranging from jet engine design[7] to pulse optimization[8] to horse race handicapping[9]. Over the last several years, they have been widely used in several fields of chemical modeling.[10]

A third issue that also arises when folding using a dynamics algorithm is how sensitive the folding pathway is to the initial state. In an earlier paper[11], I demonstrated another method for evolving folding pathways, but that approach failed when even reasonable variant starting conformations were chosen. The approach presented here partially solves that problem. As I will show, a given *S* matrix will correctly fold a limited variety of initial conformations so long as they do not differ too much from the initial conformation used when evolving the *S* matrix. Furthermore, if the target final state is subjected to the folding algorithm, it is stable and will not unfold. This approach of using a GA to evolve a state transition matrix and then testing it against a variety of initial conditions draws on the work of Koza in Genetic Programming.[12] In that paradigm, however, the functional form of the interaction is evolved, and not simply a set of parameters.

To test this approach, I have applied it to a simple 2-D polymer, similar to lattice models used elsewhere.[13-16] The method was presented with an initial conformation which had to fold into a specified target conformation in 50 steps of the folding process. Several *S* matrices were found that successfully folded the polymer. Next, each of these *S* matrices was applied to several ensembles of initial conformations to test for robustness with respect to initial conditions. For families of conformations not too different from the one used during the evolution process, some fraction of initial states folded properly.



The n-atom polymer folds via a series of discrete states or conformations, obeying the following set of rules. At each step τ, the full $n \times n$ distance matrix $D(\tau)$ is calculated, where $D_{ij}(\tau)$ is the distance from atom i to atom j. To move to the next state, a new constraint matrix $C(\tau)$ is formed where $C_{ij} = D_{ij} + S_{ij}$. S_{ij} is an element of the state transition matrix S, which is independent of τ. The modified energy

$$E = E_{MM} + \frac{1}{2}\sum_{i,j} k_b (r_{ij} - C_{ij})^2 \tag{3}$$

is then gradient minimized to produce the new conformation. (Note that r_{ij} and D_{ij} refer to the same distance.) The elements of S are constant during the folding process and act as a surrogate for the long range force field. The parameters S_{ij} lie in the range -0.5 Å$<S_{ij}<$0.5 Å, so that they can represent both attractive and repulsive interactions. C_{ij} and S_{ij} are zero for $j=i$ and $j=i\pm1$. C_{ij} is also set to zero if $D_{ij} > r_{cut}$ where r_{cut} is 4 Å in this paper. The entire folding process uses T iterations of this process, where T is 50 during the evolution phase. The number has to be large enough so that, most of the time, the polymer reaches a stable conformation, i.e. further iterations do not cause any more conformational changes. Once all 50 steps have been completed, a final gradient minimization is performed using only E_{MM} (Eq. 2). Some test runs were performed that indicated that most, but not all cases would converge in 50 steps, so all of the evolution was done at this level. During the analysis phase, the initially successful S matrices were also tested for 200 steps.

The quality of the fold, and hence of a particular state matrix S is judged by how close the final conformation is to the target. This is measured by calculating the RMS deviation between the final distance matrix, at step T, and distance matrix of the target state.

$$f(S) = \sqrt{\frac{(n-1)(n-2)}{2} \sum_{i,j>i+1}^{n} \left(D_{i,j}(T,S) - D_{i,j}(\text{Target})\right)^2} \tag{4}$$

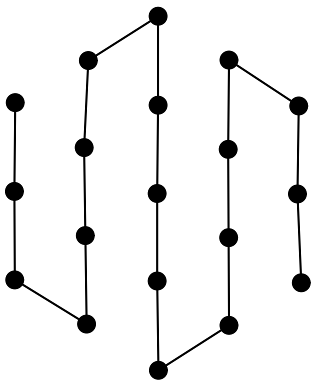
where *n* is the number of atoms. Obviously, a perfect fold yields *f(S)*=0. A GA was used to evolve state transition matrices that brought the final conformation successively closer to the target.

For the evolution process, I use a modified version of a standard "simple" GA described in Goldberg's book.[6] As in all GAs we use populations comprised of a number of individuals. Each individual is specified by a chromosome or bit string which is decoded to give the elements of the *S* matrix. For the 19 atom problem presented here, there are 153 non-zero, unique elements [(n-1)(n-2)/2] each represented by 6 bits, yielding a 918 bit chromosome. The fitness is the RMS deviation given in Eq. 3. The initial chromosomes for the population are chosen at random and the fitness for each individual is calculated. The individuals in this first generation then produce offspring who will be parents for the next generation. Parents produce children under the action of the selection, recombination and mutation operators. Roulette wheel selection is used. In crossover, two parents' chromosomes are cut at a random locus and the right and left halves of the two chromosomes are interchanged and given to the two children. Both children are placed into the population. Pairs of parents in the selected group are chosen and children formed until the new population is full. (A fixed-sized population is used.) The elitist strategy is also used, which means that one copy of the current best individual is always passed directly from one generation to the next. After recombination, each of the new chromosomes is passed to the mutation operator which, with a probability given by the mutation rate, randomly flips the bit at the chosen loci in the chromosome. Finally the fitness value of each individual is calculated and the cycle begins again. Several populations are evolved independently.

During the evolution phase, a single initial "denatured" conformation was used, which was an almost linear zig-zag with angles of $\pm 6^\circ$. Once several *S* matrices had been found that folded this conformation to the target state, four different ensembles of initial conditions were folded to test the robustness of the *S* matrix. Each ensemble contained 20 conformations. For each initial conformation in an ensemble, a random increment was added to each angle in the standard initial conformation. The angles then lie in the range $\theta_0 - \delta < \theta < \theta_0 + \delta$ where θ_0 is $\pm 6^\circ$ and δ for the four ensembles is 0.6°, 3°, 6° and 12° respectively. The final quality of a given S matrix was measured by what fraction of the initial conformations folded correctly. These tests were carried out for T=50 and T=200.

3. Numerical Results

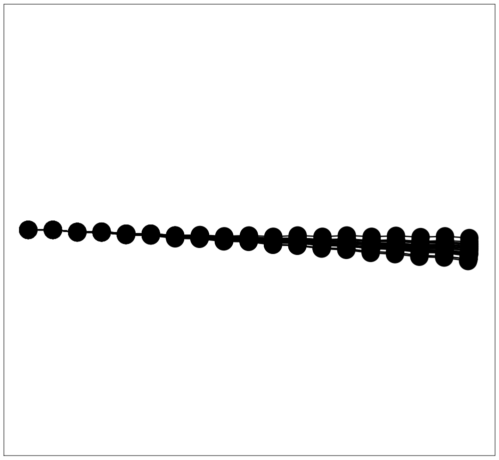
The test problem consisted of folding a 19 atom polymer. The target conformation is shown in Figure 1. This is one of a degenerate set of conformations having the global energy minimum for E_{MM} given in Eq. 2. The run used 10 populations of 100 individuals each, run for 200 generations. The GA parameters for the evolution stage are given in Table 1. A folding sequence took about 6 sec on an SGI R4000 Indigo, so the total evolution time was about 14 cpu days.



When these *S* matches were used to fold the ensembles of initial conformations, the fraction of successful folds decreased with the amount of deviation from the standard conformation. The ensemble of initial states are shown in Figure 3 for the four ensembles. The standard initial conformation is denoted by open circles. The success rates are given in Table 2 for T=50. The same data for T=200 is given in Table 3. Figure 4 shows a typical misfolded conformation. Folds that went wrong tended, as this case shows, to go very wrong.

Table 1 - Parameters for the GA run

Number of generations	200
Number of populations	10
Population size	100
Elements in *S* matrix	153
Bits per element	6
Range of *S* matrix elements	-0.5 Å<S_{ij}<0.5 Å
Mutation Rate	0.5
Crossover Rate	0.95



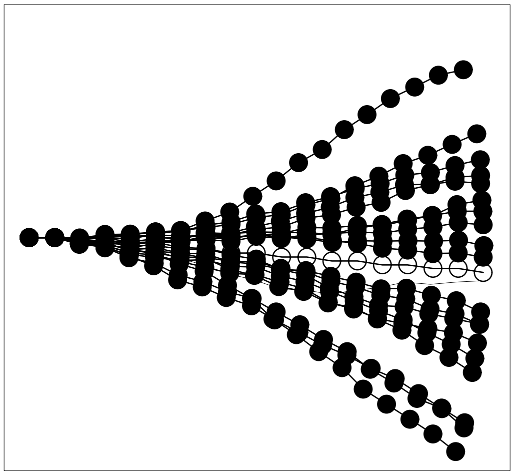


Table 2 - Success Rates for the 3 state transition matrices - *T*=50

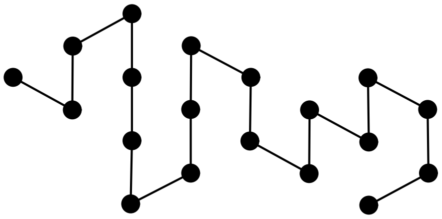
Range of Initial Variation	***S*(1)**	***S*(2)**	***S*(3)**	***S*(4)**	***S*(5)**
0.6°±**6**°	18/20	20/20	14/20	6/20	17/20
3°±**6**°	6/20	12/20	5/20	0/20	6/20
6°±**6**°	1/20	5/20	4/20	1/20	5/20
12°±**6**°	0/20	6/20	1/20	0/20	1/20

Two of the *S* matrices (*S*(1) and *S*(3)) saw the target as a metastable state at *T*=50, and pushed away from that for later times. The other three (*S*(2), *S*(4) and *S*(5)) correctly fold all of the low deviation initial conditions after 200 steps. No further change in the conformations were seen for *T*>200. Only *S*(2) correctly folded a significant fraction of all the ensembles. To see if *S*(2) could be further refined, it was subjected to a simple Monte Carlo local optimization scheme. A total of 1000 trials were taken in which the binary representation of *S*(2) was changed by randomly flipping 3 bits, and another 10000 were taken in which a single bit was flipped. If a move improved the number of initial conditions correctly folded, it would be accepted, otherwise it would be rejected. No improvement was seen, implying that the initial conformations fall into two distinct classes.

Table 3 - Success Rates for the 3 state transition matrices - *T*=200

Range of Initial Variation	***S*(1)**	***S*(2)**	***S*(3)**	***S*(4)**	***S*(5)**
0.6°±**6**°	0/20	20/20	0/20	20/20	20/20
3°±**6**°	0/20	12/20	0/20	9/20	12/20
6°±**6**°	0/20	7/20	0/20	3/20	8/20
12°±**6**°	0/20	10/20	0/20	0/20	2/20

A few words need to be said about the efficiency of the GA search process. The first correctly folding *S* matrix was found in generation 131, but a matrix that folded to a state differing from the target by a single shift defect was found much earlier, at generation 94. The approach presented here would probably benefit, as have other GA applications, from a hybrid optimization algorithm where some local search in parameter space accompanies the global search being carried out by the GA. As it is, the later phase of the evolution proceeded largely through the action of the mutation operator, which is performing an inefficient random walk through parameter space.



Feldmann and Rawn[5] use a similar dynamics algorithm to that described here. They have preliminary results for folding selected proteins, in a relatively small number of hours on a parallel distributed network of workstations. Their folding constraints (which play the role of the elements of S matrix used here) are based principally on hydrophobic and hydrogen bonding contacts. Additionally the constraints are divided into local (on-chain) and global (off-chain) sets. The local constraints drive the construction of helices, strands and turns, while the global constraints drive the organization of tertiary structure. DGEOM[17] is the principal tool used to enforce the folding constraints. The strengths of their constraint parameters are tuned to give correct folding. A more general parameter determination scheme could possibly be useful. However, their current folding method is too expensive to place in the middle of the GA optimization loop described here. Either a modification of the GA is needed that requires fewer function evaluations, or a less expensive variant of the Feldmann-Rawn method would be needed. Note however, that the expensive evolution calculation needs to be done only once in principle, because it is being used to derive the general state transition matrix. Once that is available, folding a new protein is relatively inexpensive.

Acknowledgments: The author acknowledges partial support by the Department of Energy under contract DE-ACO4-94AL85000 and by the Structural Biology Initiative at Lawrence Livermore National Laboratory. I would also like to thank R. Feldmann for useful conversations regarding the methods described here.

References

(1) Bosie, J. U., Eisenberg, D., *Curr.Opinions Str. Bio* , **3**, 437-444 (1993).
(2) Moult, J., Ed., *Proteins, Str. Func. Genet. (In Press),* 1995.
(3) Ngo, J. T., Marks, J. Karplus, M., *Computational Complexity: Protein Structure prediction and the Levinthal Paradox,* Birkhauser, New York, 1994.
(4) Wodak, S. J., Rooman, M.J., *Curr.Opinions, Str. Bio.* , **3**, 247-259 (1993).
(5) Feldmann, R. J., Rawn, J.D., "The Topology of Protein Folding", to be submitted to the Proc.Nat.Acad.Sci., USA.

(6) Goldberg, D., *Genetic Algorithms in Search, Optimization, and Learning*, Addison Wesley, Reading, Mass., 1989.

(7) Bramlette, M. F., Cusic, R., in *Proceedings of the Third International Conf.erence on Genetic Algorithms*, J. D. Schaffer, Ed., Morgan Kaufman, San Mateo, CA, 1989, pp. 213.

(8) Judson, R. S., Rabitz, H., *Phys.Rev.Lett.* , **68**, 1500 (1992).

(9) de la Maza, M., in *Proceedings of the Third International Conference on Genetic Algorithms*, J. D. Schaffer, Ed., Morgan Kaufman, San Mateo, CA, 1989, pp. 208-212.

(10) Judson, R., in *Reviews in Computational Chemistry*, K. B. Lipkowitz Boyd, D.B., Ed., VCH Publishers, New York, 1996.

(11) Judson, R. S., *J.Phys.Chem.* , **96**, 10102-10104 (1992).

(12) Koza, J., *Genetic Programming*, MIT Press, Cambridge, 1992.

(13) Dill, K. A., *Biochemistry* , **24**, 1501 (1985).

(14) Chan, H. A., Dill, K.A., *Ann.Rev. of Biophysics and Biophysical Chemistry* , **20**, 447-490 (1991).

(15) Skolnick, J., Kolinski, A, *Science* , **250**, 1121 (1990).

(16) Dandekar, T., Argos, P., *Protein Engineering* , **5**, 637-645 (1992).

(17) DGEOM: Distance Geometry, 590, Blaney, J. M., Crippen, G.M., Dearing, A., Dixon, J.S., Quantum Chemistry Program Exchange, Indiana University, Bloomington, IN, 1988.