On the Kinematics of Protein Folding

SEAN CAHILL,[1] MICHAEL CAHILL,[2] KEVIN CAHILL[3]
[1]Department of Computer Science, University of New Mexico, Albuquerque, New Mexico 87131
[2]School of Medicine, Uniformed Services University, Bethesda, Maryland 20814
[3]Department of Physics and Astronomy, University of New Mexico,
Albuquerque, New Mexico 87131

Received 25 June 2002; Accepted 25 November 2002

Abstract: We offer simple solutions to three kinematic problems that occur in the folding of proteins. We show how to construct suitably local elementary Monte Carlo moves, how to close a loop, and how to fold a loop without breaking the bond that closes it.

 J Comput Chem 24: 1364–1370, 2003

Key words: protein folding; homology modeling; loops; Monte Carlo; kinematics; wriggling

Three Kinematic Problems

Monte Carlo searches for the low-energy states of proteins pose three related kinematic problems:

1. How does one design suitably local elementary Monte Carlo moves?
2. How does one configure a main-chain loop between two fixed points?
3. How does one fold a loop without breaking any of its bonds?

The choice of elementary localized moves may be almost as important as the choice of the energy function. The loop problems occur if one adds or deletes residues in a backbone strand in order to model a homologous or mutant protein from a known X-ray structure. They also occur if one has a loop that is unresolved by X-ray crystallography or a primary sequence with two cysteines that might form a disulfide bond.

Local Moves

The positions $\vec{r}_i$ of the atoms of a protein are local coordinates, but they are subject to constraints. The dihedral angles, ϕ_i and ψ_i, describe the state of a protein more efficiently, but they are not local coordinates; a change in a dihedral angle near the center of a protein rotates half of the molecule, moving distant atoms farther than nearby ones. Such thrashing violates the conservation of angular momentum and of energy, and engenders steric clashes. Real proteins do not thrash; they wriggle. So if one uses the dihedral angles as coordinates, then one must craft elementary moves that are suitably local.

How does one combine rotations about dihedral bonds so that the net motion is suitably local? This problem was addressed by Gō and Scheraga[1] and has since been discussed in many articles on proteins[2–5] and polymers.[6–17] But rotations are complicated. They are 3×3 orthogonal matrices with elements that are sines and cosines of the relevant angles. The nonlinearity of these trigonometric functions has held back progress on this problem.

Yet every smooth function becomes linear when examined at a small-enough scale. Rotations of infinitesimally small angles are linear functions of those angles. Linear algebra is relatively simple.

The change $\vec{dr}$ in the position $\vec{r}$ of an atom due to a rotation by a small angle ε about a bond axis represented by the unit vector $\hat{b}$ is the cross-product of $\varepsilon\hat{b}$ with the vector to the point $\vec{r}$ from any point $\vec{c}$ on the axis:

$$\vec{dr} = \varepsilon\hat{b} \times (\vec{r} - \vec{c}) \tag{1}$$

So the change $\vec{dr}$ due to n rotations by the small angles ε_i about the bonds $\hat{b}_i$ is the sum

$$\begin{aligned}\vec{dr} &= \sum_{i=1}^{n} \varepsilon_i \hat{b}_i \times (\vec{r} - \vec{c}_i) \\ &= \sum_{i=1}^{n} \varepsilon_i \hat{b}_i \times (\vec{r} - \vec{a} + \vec{a} - \vec{c}_i) \\ &= \left(\sum_{i=1}^{n} \varepsilon_i \hat{b}_i \right) \times (\vec{r} - \vec{a}) + \sum_{i=1}^{n} \varepsilon_i \hat{b}_i \times (\vec{a} - \vec{c}_i)\end{aligned} \tag{2}$$

Correspondence to: K. Cahill; e-mail: cahill@unm.edu

which is a rotation and a translation. The point $\vec{a}$ is entirely arbitrary; a convenient choice is the average position $\vec{a} = (1/n)\,\Sigma_i\,\vec{c}_i$ of the points $\vec{c}_i$.

The rotation is less local than the translation because its effect is proportional to the length of the vector $(\vec{r} - \vec{a})$. But it is easy to make the net rotation

$$\left(\sum_{i=1}^{n} \varepsilon \hat{b}_{ii}\right) \times (\vec{r} - \vec{a}) \tag{3}$$

vanish. A set of n vectors $\hat{b}_i$ is said to be *linearly dependent* if there are coefficients ε_i such that

$$\sum_{i=1}^{n} \varepsilon_i \hat{b}_i = \vec{0} \tag{4}$$

Any set of n vectors $\hat{b}_i$ in a space of d dimensions is linearly dependent if $n > d$. Because the bond vectors $\hat{b}_i$ are three dimensional, any four or more are linearly dependent. So if we use at least four bond vectors $\hat{b}_i$, then we always may find angles ε_i that make the sum [eq. (4)] vanish.

We may find these angles by performing a singular-value decomposition. Every $n \times m$ real matrix B may be written as the product of an $n \times n$ orthogonal matrix U, an $n \times m$ matrix Σ, and an $m \times m$ orthogonal matrix V in the singular-value decomposition:

$$B = U\Sigma V \tag{5}$$

The matrix Σ is zero except for its diagonal matrix elements, which are the non-negative singular values of the matrix B.

To find small angles ε_i, such that

$$\sum_{i=1}^{n} \varepsilon_i \hat{b}_i = \vec{0} \tag{6}$$

we set $n = 4$ and form a 3×4 matrix B whose columns are the four bond vectors $\hat{b}_i$. Its singular-value decomposition is

$$\begin{aligned} B &= \begin{pmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \end{pmatrix} \\ &= U \begin{pmatrix} s_1 & 0 & 0 & 0 \\ 0 & s_2 & 0 & 0 \\ 0 & 0 & s_3 & 0 \end{pmatrix} \begin{pmatrix} V_{11} & V_{12} & V_{13} & V_{14} \\ V_{21} & V_{22} & V_{23} & V_{24} \\ V_{31} & V_{32} & V_{33} & V_{34} \\ V_{41} & V_{42} & V_{43} & V_{44} \end{pmatrix} \end{aligned} \tag{7}$$

Because the matrices U and V are orthogonal, their rows and columns are orthonormal vectors. In particular

$$\sum_{k=1}^{4} V_{ik} V_{4k} = \delta_{i4} \tag{8}$$

So if we take the small angles to be $\varepsilon_i = xV_{4i}$, where x is a scale factor, then the orthonormality [eq. (8)] of the rows of V will imply

$$V\varepsilon = \begin{pmatrix} V_{11} & V_{12} & V_{13} & V_{14} \\ V_{21} & V_{22} & V_{23} & V_{24} \\ V_{31} & V_{32} & V_{33} & V_{34} \\ V_{41} & V_{42} & V_{43} & V_{44} \end{pmatrix} \begin{pmatrix} xV_{41} \\ xV_{42} \\ xV_{43} \\ xV_{44} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ x \end{pmatrix}$$

and so

$$B\varepsilon = \sum_{i=1}^{4} \hat{b}_i \varepsilon_i = U \begin{pmatrix} s_1 & 0 & 0 & 0 \\ 0 & s_2 & 0 & 0 \\ 0 & 0 & s_3 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ x \end{pmatrix} = \vec{0} \tag{9}$$

LAPACK[18] is stable tested software that solves many problems of linear algebra. Its subroutine dgesvd performs singular-value decompositions in double-precision arithmetic. The call to

dgesvd('N', 'A', 3, 4, B, 3, S, U, 1, V, 4, WORK, 402, INF)

returns the matrix V. The small angles ε_i may be taken to be

$$\varepsilon_i = xV_{4i} \tag{10}$$

in which x is a random number in the range $-\delta < x < \delta$, and δ is small enough for the small-angle approximations [eqs. (1–3)] to be valid. We used $\delta = 0.0125$. WORK is a double-precision array of dimension LWORK, here taken to be 402. If the call is successful, then INF is returned as zero.

We use the word *wriggling* to denote this way of canceling the nonlocal effects of rotations. We tested our wriggling algorithm in Monte Carlo simulations of protein folding against an algorithm in which successive dihedral angles were varied independently, *thrashing,* and also against one in which the dihedral angles were varied in groups of four, *fourfold thrashing.* To separate kinematic from dynamic issues, we used as a nearly perfect but highly artificial energy function the RMSD between the main-chain atoms of our wriggling protein and those of its pdb file.

Because of our use of the RMSD as an energy function, the proteins of our simulations are phantoms; they can pass through themselves. A real but approximate energy function would reject all moves into excluded volume; it therefore would reject many thrashing moves because of their large-scale motions. The use of the RMSD in our tests deprives wriggling of one of its key advantages over thrashing, namely that its localized motions are less likely to involve steric clashes. Thus the utility of wriggling in simulations with real energy functions may be greater than is indicated by our tests.

In our tests of various move sets, we let each algorithm fold 10 highly denatured coils of phosphoglycerate kinase (16PD.pdb, 415 aa), five of which were stretched. The algorithm made a list of the rotatable bonds, which we took to be all the main-chain N—C_α and C_α—C′ bonds, except for the N—C_α bonds of the prolines. In 16PK.pdb, there were $N_B = 810$ rotatable bonds. The thrashing code varied all N_B main-chain dihedral angles in each Monte Carlo sweep, but the wriggling and fourfold-thrashing codes, which

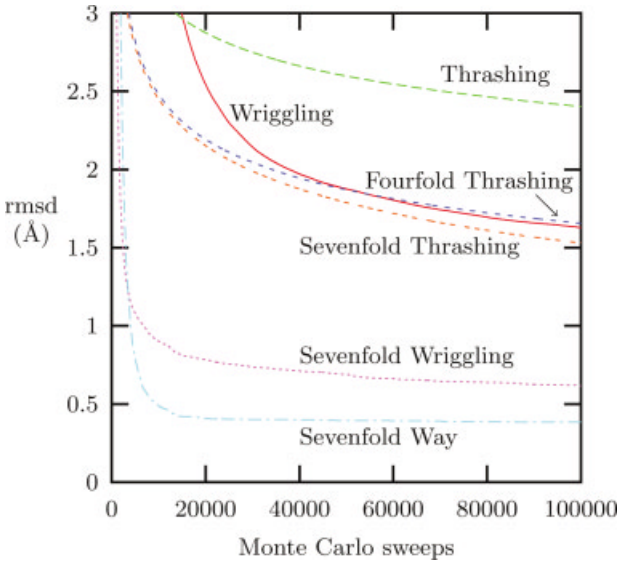

Figure 1. For the protein phosphoglycerate kinase, 16PK, the lines trace the mean values of the RMSD for 10 runs guided by algorithms that, respectively, use thrashing, fourfold thrashing, wriggling, sevenfold thrashing, sevenfold wriggling, and the sevenfold way.

apply their moves to groups of four bonds at a time, started with bonds 1–4 and ended with bonds $(N_B - 3) - N_B$. After 100,000 Monte Carlo sweeps, the wriggling, thrashing, and fourfold-thrashing algorithms reduced their mean RMSD's to 1.63 ± 0.02, 2.40 ± 0.02, and 1.66 ± 0.03 Å, respectively. In Figure 1 the mean RMSD's are plotted against the number of sweeps. These results are similar to those we obtained earlier[19] using C_α RMSD's.

The use of $n = 4$ bonds $\hat{b}_i$ is the simplest way of canceling the highly nonlocal effects of rotations, but it is not the best way. In further tests we found much lower RMSD's by using $n = 7$ bonds $\hat{b}_i$. We call this *sevenfold wriggling*. The matrix B is now 3×7, each of its seven columns being a bond vector $\hat{b}_i$. The call is to

dgesvd('N', 'A', 3, 7, B, S, U, 1, V, 7, WORK, 460, INF).

The angles are given by $\varepsilon_i = xV_{7i}$ where $|x| < \delta$ is a random number, and δ is small enough that the small-angle approximations [eqs. (1–2)] are valid. We used $\delta = 0.0125$.

Sevenfold wriggling dropped the mean RMSD for 16PK to 0.62 ± 0.02 Å, as shown in Figure 1. We also experimented with using more than seven bonds: $n = 8$ gave 0.62 Å; $n = 9$ gave 0.70 Å; $n = 10$ gave 0.76 Å; and $n = 20$ gave 1.30 Å. Sevenfold thrashing gave a mean RMSD of 1.53 ± 0.03 Å.

By using seven or more bonds, we may cancel not only the net rotation

$$\sum_{i=1}^{n} \varepsilon_i \hat{b}_i = 0 \tag{11}$$

but also the net translation

$$\sum_{i=1}^{n} \varepsilon_i \hat{b}_i \times (\vec{a} - \vec{c}_i) = 0 \tag{12}$$

To do this, we write these two conditions in terms of the six-vectors

$$s_i = \begin{pmatrix} \hat{b}_i \\ \hat{b}_i \times (\vec{a} - \vec{c}_i) \end{pmatrix} \tag{13}$$

as

$$\sum_{i=1}^{n} \varepsilon_i s_i = 0 \tag{14}$$

Because any seven or more six-vectors s_i are linearly dependent, such small angles ε_i always exist if at least seven bonds $\hat{b}_i$ are used. We call such moves *strictly local wriggling*. The matrix B now is 6×7, each of its seven columns being a six-vector s_i, and the call is to

dgesvd('N', 'A', 6, 7, B, 6, S, U, 1, V, 7, WORK, 850, INF).

One might think (as we did) that strictly local wriggling is the ideal way to fold a protein. But although it is well suited to our third kinematic problem, the folding of a closed loop, it is very slow because it leaves the first and last backbone atoms unmoved to first order in the small angles ε_i. What does work well is the use of sevenfold wriggling and strictly local wriggling on alternate Monte Carlo sweeps along the protein. We call this technique the *sevenfold way*. It reduced the mean RMSD of 16PK to 0.38 ± 0.02 Å, as shown in Figure 1.

We also compared the sevenfold way to sevenfold thrashing on two other globular proteins—sperm-whale myoglobin (1A6M.pdb, 151 aa) and human muscle fatty-acid binding protein (1HMR.pdb, 131 aa). On these shorter proteins, we used the sevenfold way on all sets of seven contiguous bonds, fourfold wriggling on the three final sets of four contiguous bonds, and allowed arbitrary rotations about the last three bonds. For each protein, we made two denatured starting configurations, one that was a single long straight β-strand and one that was a single long straight α-helix. In 10 runs of 100,000 sweeps, sevenfold thrashing reduced the mean main-chain RMSD's of the long α-helix and of the long β-strand of 1A6M to 1.55 ± 0.01 and 0.89 ± 0.03 Å, respectively. The sevenfold-way did much better, respectively reducing the RMSD's to 0.12 ± 0.02 and 0.19 ± 0.05 Å, as shown in Figure 2. In the case of 1HMR, the mean RMSD's of the long α-helix and of the long β-strand, respectively, were reduced in 10 runs of 100,000 sweeps to 0.63 ± 0.02 and 0.63 ± 0.01 Å by sevenfold thrashing and to 0.44 ± 0.02 and 0.43 ± 0.03 Å by the sevenfold way, as shown in Figure 3.

Because our artificial energy function, the RMSD, is not directly related to an energy, our fixed-temperature simulations were carried out at zero temperature. We did however test the sevenfold way against thrashing in runs with simulated annealing. In these Monte Carlo simulations, the temperature dropped either linearly

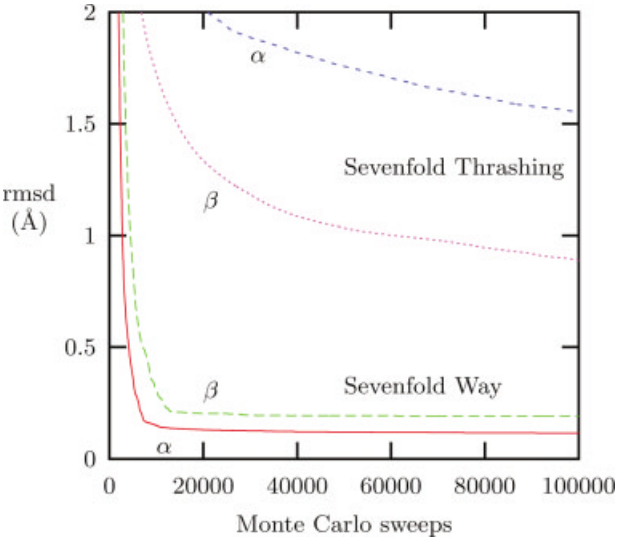

Figure 2. For the protein myoglobin, 1A6M, the lines trace the mean values of the RMSD for 10 runs starting from a long α-helix and from a long β-strand guided by sevenfold thrashing or the sevenfold way.

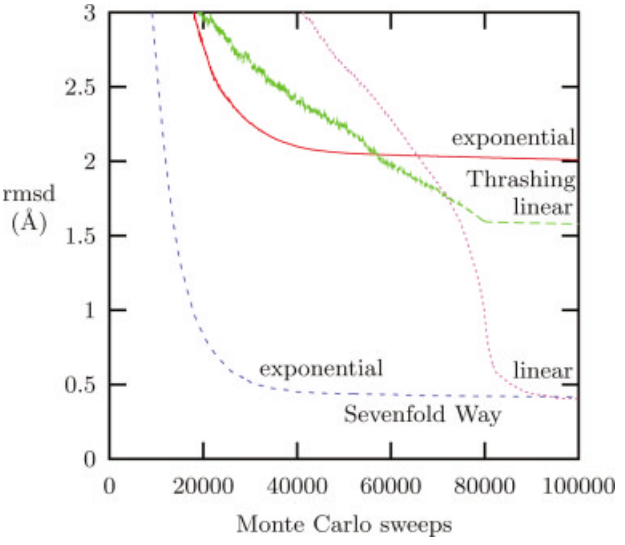

Figure 4. For the protein 16PK, the lines trace the mean values of the RMSD for 10 runs guided by algorithms that, respectively, use thrashing and the sevenfold way with linear and exponential cooling.

or exponentially with the sweep number from a very high T at the start of the simulation to $T = 0$ at 80,000 sweeps. These runs finished with 20,000 sweeps at $T = 0$. As shown in Figure 4, the sevenfold way reduced the mean RMSD of 16PK to 0.42 $\pm$ 0.01 Å with exponential cooling and to 0.40 $\pm$ 0.02 Å with linear cooling. Thrashing reduced the mean RMSD to 2.01 $\pm$ 0.03 Å with exponential cooling and to 1.58 $\pm$ 0.15 Å with linear cooling. We did not try entropic sampling.[20]

In these algorithms, the changes in the dihedral angles are small so that the small-angle approximations [eqs. (1–2)] are valid; atoms are displaced in each move by no more than about 0.05 Å. Such small moves might seem to imply a slow rate of convergence to minimum-energy states. But globular proteins typically are 50 Å or less in diameter, and so in 1000 moves an atom could traverse the protein. To test whether larger moves would give faster convergence, we made 10 thrashing runs with a cutoff 40 times bigger than our usual $\delta = 0.0125$. These runs with large-angle thrashing reduced the mean main chain RMSD in 100,000 sweeps to 3.69 $\pm$ 0.06 Å, which is to be compared to 2.40 $\pm$ 0.02 Å with $\delta = 0.0125$. So bigger moves may not mean faster convergence, perhaps because in a partially folded protein, smaller moves are more likely than larger ones to lower the energy.

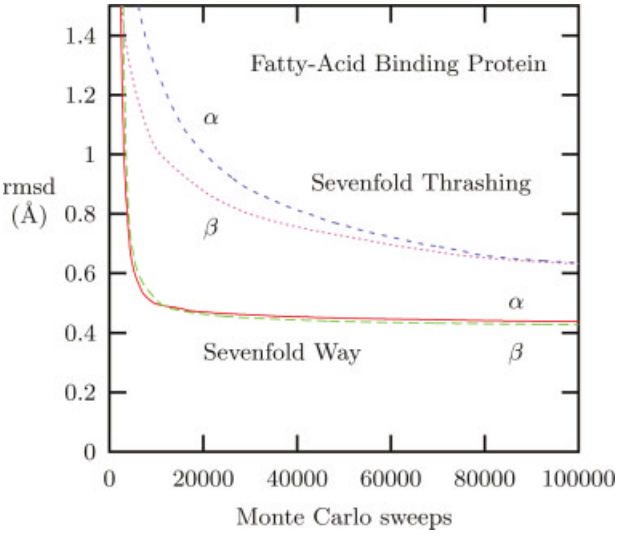

Figure 3. For the human muscle fatty-acid binding protein, 1HMR, the lines trace the mean values of the RMSD for 10 runs starting from a long α-helix and from a long β-strand guided by sevenfold thrashing or the sevenfold way.

To Close a Loop

When modeling a homologous or mutant protein or an unresolved loop, one must configure a main chain between two fixed points. This problem also arises if one has a main chain with two cysteine residues, and one needs to make a disulfide bridge between them, forming a loop.

Let us consider the case of a loop with a disulfide bridge. Provided the strand of backbone is long enough, we may change the dihedral angles of the residues of the strand between the cysteines so as to move the β-carbon and the γ-sulfur of the second cysteine into the required positions opposite those of the first cysteine, which is held fixed. Let $\vec{C}_{\beta 0}$ and $\vec{S}_{\gamma 0}$ be the points to which the β-carbon and the γ-sulfur should be moved, and let $\vec{C}_\beta$ and $\vec{S}_\gamma$ be their present locations.

We have seen in eq. (2) that several small rotations amount to a net rotation and a net translation. We may choose the small angles of the rotations so as to correctly orient the $\vec{C}_\beta - \vec{S}_\gamma$ bond and to move it to the right position.

The required translation is

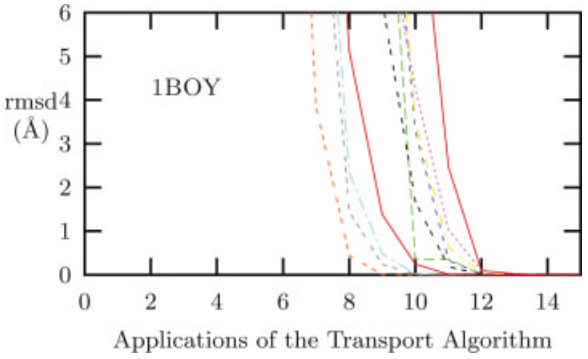

Figure 5. The transport algorithm closed the 44-bond loop of the protein 1BOY in 14 sweeps.

$$\vec{t} = \tfrac{1}{2}\,(\vec{S}_{\gamma 0} + \vec{C}_{\beta 0} - \vec{S}_{\gamma} - \vec{C}_{\beta}) \tag{15}$$

The axis of the required rotation is

$$\vec{x} = (\vec{S}_{\gamma} - \vec{C}_{\beta}) \times (\vec{S}_{\gamma 0} - \vec{C}_{\beta 0}) \tag{16}$$

and the tangent of its angle θ is the ratio of the length $\|\vec{x}\|$ of this cross-product to the corresponding scalar-product

$$d = (\vec{S}_{\gamma} - \vec{C}_{\beta}) \cdot (\vec{S}_{\gamma 0} - \vec{C}_{\beta 0}) \tag{17}$$

that is, $\tan\,\theta = \|\vec{x}\|/d$. So the required angle of rotation is

$$\vec{\theta} = \text{atan2}(\|\vec{x}\|, d)\hat{x} \tag{18}$$

Thus we must perform $n \geq 6$ small rotations, each of angle ε_i about bond $\hat{b}_i$, so that the net rotation is

$$\vec{\theta} = \sum_{i=1}^{n} \varepsilon_i \hat{b}_i \tag{19}$$

and the net translation is

$$\vec{t} = \sum_{i=1}^{n} \varepsilon_i \hat{b}_i \times (\vec{a} - \vec{c}_i) \tag{20}$$

These two conditions may be written in terms of the 6-vectors

$$s_i = \begin{pmatrix} \hat{b}_i \\ \hat{b}_i \times (\vec{a} - \vec{c}_i) \end{pmatrix} \tag{21}$$

as

$$\sum_{i=1}^{n} \varepsilon_i s_i = \begin{pmatrix} \vec{\theta} \\ \vec{t} \end{pmatrix} \tag{22}$$

We form a matrix B whose first n columns are the 6-vectors s_i and whose last column is the 6-vector $(-\vec{\theta}, -\vec{t})$. A call to dgesvd returns the matrix V, and the small angles ε_i are given by a suitably safe version of $\varepsilon_i = V(j, i)/V(j, n + 1)$, where $j > 6$ is the row index with the largest value of $|V(j, n + 1)|$. In our applications of this transport algorithm, we set n equal to the number of rotatable main-chain bonds of the loop.

The extracellular domain of human tissue factor (1BOY.pdb, 219 aa) has two disulfide bonds. The one between residues 186 and 209 closes a loop that has 44 rotatable main-chain bonds. The transport algorithm with $n = 44$ closed this loop to less than 0.0001 Å in 14 sweeps, as shown in Figure 5.

To Fold a Loop

When the transport algorithm closes a loop, the loop may well be of quite high energy with steric conflicts. It is therefore necessary to vary the conformation of the loop without breaking it. Strictly local wriggling moves are well suited to this task. But we have found that the sevenfold way combined with the transport algorithm does a better job. We used the sevenfold way on all sets of seven contiguous bonds and applied the transport algorithm after every 200,000 sweeps of the sevenfold way. As an energy function, we used the all-atom RMSD between the atoms of the folding loop and those of the pdb file. In 10 runs of 2,000,000 sweeps from fully extended coils of the 44-bond loop of 1BOY, this process reduced the loop's mean all-atom RMSD to 0.85 ± 0.14 Å and its mean main-chain RMSD to 0.42 ± 0.06 Å, as shown in Figure 6.

The sevenfold way needed a 1,000,000 sweeps to reduce the mean all-atom RMSD of the 44-bond loop in 1BOY to less than 1 Å. One may do better by letting the program randomly choose how many bonds, $7 \leq n \leq 44$, to fold and which row, $7 \leq j \leq n$, of

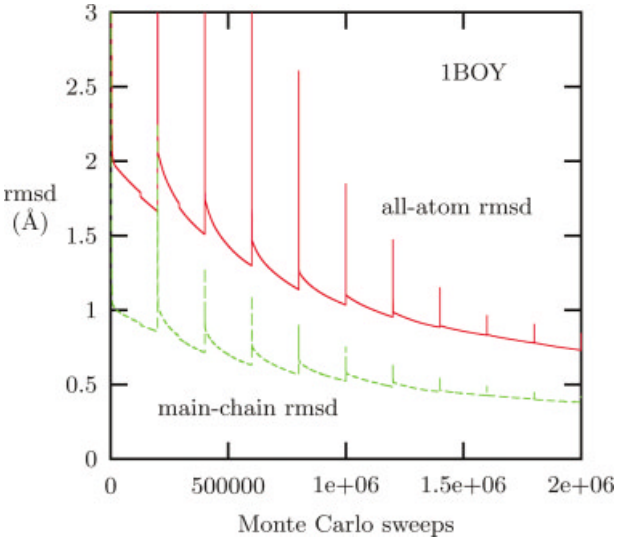

Figure 6. The sevenfold way needed about a million sweeps to reduce the main chain and all-atom RMSD's of the loop of 44 rotatable bonds in 1BOY below 1 Å. The spikes are caused by the application of the transport algorithm, which closes the loop after every 200,000 sweeps.

the matrix V of the singular-value decomposition [eq. (5)] to use for the angles $\varepsilon_i = xV_{ji}$. A further advantage is gained by allowing suitable moves of the last six bonds, as was done for the proteins 1A6M and 1HMR. In 10 runs of 20,000 sweeps, this algorithm reduced the mean all-atom RMSD of the 44-bond loop of 1BOY to 0.66 ± 0.22 Å. The large standard error arose from the fact that the RMSD got stuck at 1.326 Å in five of the runs, but went to less than 0.02 Å in the other five.

Hoping to send the RMSD of every run to zero, we introduced a new Monte Carlo move. When a protein has folded to a globular form close to the native structure, a net translation may disrupt the folding protein as much as or more than a net rotation. So we added a new move in which the net translation, but not the net rotation, is set to zero:

$$\vec{t} = \sum_{i=1}^{n} \varepsilon_i \hat{b}_i \times (\vec{a} - \vec{c}_i) = 0 \qquad (23)$$

One may enforce this condition by forming a matrix B whose $n \geq 4$ columns are the vectors $\hat{b}_i \times (\vec{a} - \vec{c}_i)$. The resulting *n-fold way* on successive sweeps sets to zero net translations, net rotations, and both the net translations and the net rotations. It reduced the RMSD of the 44-bond loop of 1BOY to zero in 20 out of 20 runs in fewer than 25,000 sweeps, as shown in Figure 7.

The aspartyl protease human progastricsin (1HTR.pdb, 329 aa) has three disulfide bonds. The one between residues 251 and 284 closes a loop that has 60 rotatable bonds. We denatured and stretched that loop and then used the n-fold way to fold it. By limiting the number n of simultaneously rotated bonds to 44, which by default was true in the case of 1BOY, we found that we could drive the RMSD to zero in 7 out of 10 runs, as shown in

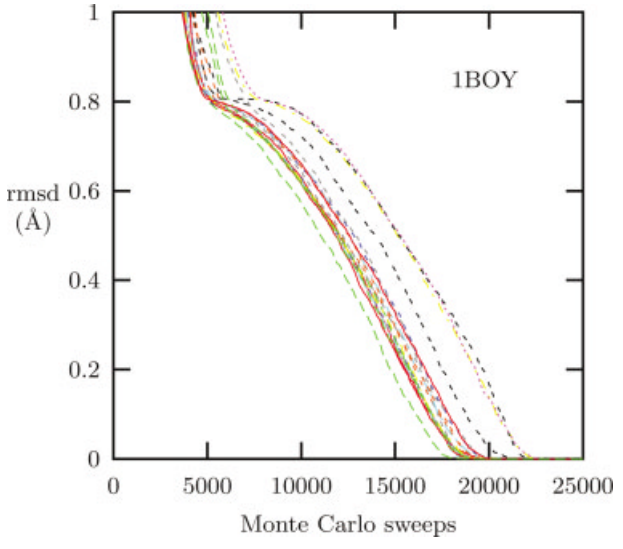

Figure 7. The n-fold way reduced to zero the all-atom RMSD of the loop of 44 rotatable bonds in 1BOY in all 20 runs of 25,000 Monte Carlo sweeps.

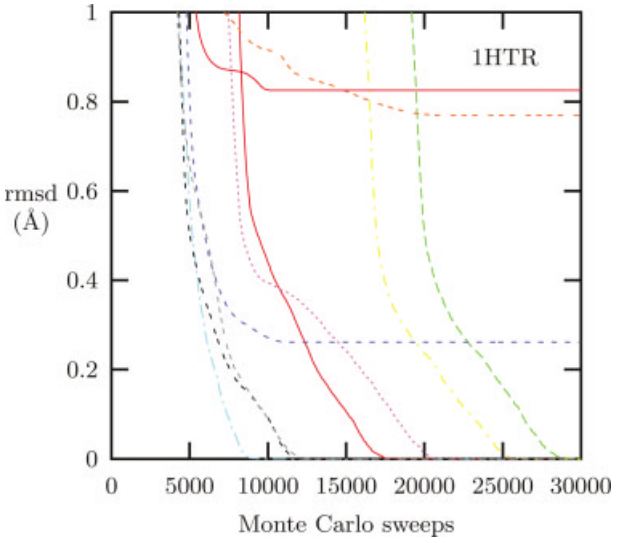

Figure 8. The n-fold way reduced to zero the all-atom RMSD of the loop of 60 rotatable bonds in human progastricsin, 1HTR.pdb, in 7 of 10 runs of 30,000 Monte Carlo sweeps.

Figure 8. In simulations with n set equal to 20, 30, 40, and 50, we found that the RMSD went to zero in four or five out of 10 runs.

Summary

We have presented simple solutions to three kinematic problems that occur in the folding of proteins. We have shown how to construct suitably local elementary Monte Carlo moves, how to close a loop, and how to fold a loop without breaking any of its bonds.

In future work, we intend to determine whether these kinematic algorithms improve the efficiency of finite-temperature Monte Carlo searches guided by realistic energy functions with implicit solvation and excluded volume.

Acknowledgments

We have benefited from talking with S. Atlas, D. Baker, B. Brooks, G. Cahill, P. Cahill, E. Coutsias, L. deEcheandia, K. Dill, D. Dix, S. Fellini, H. Frauenfelder, K. Frost, N. van Gulick, U. Hansmann, G. Herling, D. Lanar, A. Parsegian, W. Saslow, C. Schwieters, C. Seok, P. J. Steinbach, C. Strauss, and R. Venable. Some of this work was done while one of us (K.C.) was on sabbatical at the Center for Molecular Modeling of the Center for Information Technology of the National Institutes of Health. We should like to thank Ken Dill and Peter J. Steinbach, respectively, for the hospitality they each extended to S.C. at UCSF and K.C. at NIH. The UNM RAC provided a grant for software. This study used the high-performance computational capabilities of the Biowulf/LoBoS3 cluster at the National Institutes of Health, Bethesda, MD. Some of our computations

were done by the computers of the Albuquerque High-Performance Computing Center.

References

1. Gō, N.; Scherage, H. A. Macromolecules 1970, 3, 178.
2. Burkert, U.; Allinger, N. L. Molecular Mechanics; ACS Monographs; American Chemical Society: Washington, DC, 1982.
3. Bruccoleri, R.; Karplus, M. Macromolecules 1985, 18, 2767.
4. Palmer, K. A.; Scheraga, H. A. J Comput Chem 1991, 12, 505.
5. Elofsson, A.; Le Grand, S. M.; Eisenber, D. Proteins: Struct, Funct, Genet 1995, 23, 73.
6. Schatzki, T. F. Polym Prepr (Am Chem Soc, Div Polym Chem) 1965, 6, 646.
7. Helfand, E. J Chem Phys 1971, 54, 4651.
8. Skolnick, J.; Helfand, E. J Chem Phys 1980, 72, 5489.
9. Helfand, E.; Wasserman, Z. R.; Weber, T. A. Macromolecules 1980, 13, 526.
10. Helfand, E.; Wasserman, Z. R.; Weber, T. A. Polym Prepr (Am Chem Soc, Div Polym Chem) 1981, 22, 279.
11. Helfand, E.; Wasserman, Z. R.; Weber, T. A.; Skolnick, J.; Runnels, J. H. J Chem Phys 1981, 75, 4441.
12. Weber, T. A.; Helfand, E.; Wasserman, Z. R. Simulation of Polyethylene; Advances in Chemistry Series 204 (molecular-based study of fluids 20); American Chemical Society: Washington, DC, 1983; p 487.
13. Helfand, E. Science 1984, 226, 647.
14. Dodd, L. R.; Boone, T. D.; Theodorou, D. N. Mol Phys 1993, 78, 961.
15. Leontidis, E.; de Pablo, J. J.; Laso, M.; Suter, U. W. Adv Polym Sci 1994, 116, 283.
16. Dinner, A. R. J Comput Chem 2000, 21, 1132.
17. Kolossvàry, I.; Keserû, G. M. J Comput Chem 2001, 22, 21.
18. Anderson, E.; Bai, Z.; Bischof, C.; Blackford, S.; Demmel, J.; Dongarra, J.; Du Croz, J.; Greenbaum, A.; Hammarling, S.; McKenney, A.; et al. LAPACK Users' Guide, 3rd Ed.; SIAM: Philadelphia, 1999. Available on-line at http://www.netlib.org/lapack/lug/lapack_lug.html.
19. Cahill, M.; Cahill, S.; Cahill, K. Biophys J 2002, 82, 2665. cond-mat/0108218.
20. Lee, J. Phys Rev Lett 1993, 71, 211.