Full cyclic coordinate descent: solving the protein loop closure problem in Cα space

Wouter Boomsma and Thomas Hamelryck*

Address: Bioinformatics center, Institute of Molecular Biology and Physiology, University of Copenhagen, Universitetsparken 15, Building 10, DK-2100 Copenhagen, Denmark
Email: Wouter Boomsma - wb@binf.ku.dk; Thomas Hamelryck* - thamelry@binf.ku.dk
* Corresponding author

Abstract

Background: Various forms of the so-called loop closure problem are crucial to protein structure prediction methods. Given an N- and a C-terminal end, the problem consists of finding a suitable segment of a certain length that bridges the ends seamlessly.

In homology modelling, the problem arises in predicting loop regions. In de novo protein structure prediction, the problem is encountered when implementing local moves for Markov Chain Monte Carlo simulations.

Most loop closure algorithms keep the bond angles fixed or semi-fixed, and only vary the dihedral angles. This is appropriate for a full-atom protein backbone, since the bond angles can be considered as fixed, while the (φ, ψ) dihedral angles are variable. However, many de novo structure prediction methods use protein models that only consist of Cα atoms, or otherwise do not make use of all backbone atoms. These methods require a method that alters both bond and dihedral angles, since the pseudo bond angle between three consecutive Cα atoms also varies considerably.

Results: Here we present a method that solves the loop closure problem for Cα only protein models. We developed a variant of Cyclic Coordinate Descent (CCD), an inverse kinematics method from the field of robotics, which was recently applied to the loop closure problem. Since the method alters both bond and dihedral angles, which is equivalent to applying a full rotation matrix, we call our method Full CCD (FCDD). FCCD replaces CCD’s vector-based optimization of a rotation around an axis with a singular value decomposition-based optimization of a general rotation matrix. The method is easy to implement and numerically stable.

Conclusion: We tested the method’s performance on sets of random protein Cα segments between 5 and 30 amino acids long, and a number of loops of length 4, 8 and 12. FCCD is fast, has a high success rate and readily generates conformations close to those of real loops. The presence of constraints on the angles only has a small effect on the performance. A reference implementation of FCCD in Python is available as supplementary information.
In between the end points of the template's loop. The modelled loop needs to bridge the gap between the protein being modelled and the template is often necessary to rebuild certain loops that differ in structure prediction contexts. In homology modelling, it is often necessary to rebuild certain loops that differ between the protein being modelled and the template protein [2]. The modelled loop needs to bridge the gap between the end points of the template's loop.

The loop closure problem arises in at least two different structure prediction contexts. In homology modelling, it is often necessary to rebuild certain loops that differ between the protein being modelled and the template protein [2]. The modelled loop needs to bridge the gap between the end points of the template's loop.

In de novo prediction, local resampling or local moves can be considered as a variant of the loop closure problem. Typically, the conformation of a protein segment needs to be changed without affecting the rest of the protein as a sampling step in a Markov chain Monte Carlo (MCMC) procedure [3]. In both homology and de novo structure prediction, the problem is however essentially the same.

The classic article by Go and Scheraga [1] describes an analytical solution to finding all possible solutions for a protein backbone of three residues. In this case, the degrees of freedom (DOF) comprise six dihedral angles, i.e. the backbone's (φ, ψ) angles. Another approach is to use a fragment library derived from the set of solved protein structures, and look for fragments or combinations of fragments that bridge the given fixed ends [4-6]. More recently, the loop closure problem has been tackled using algorithms borrowed from the field of robotics, in particular inverse kinematics methods [7-9]. Still other methods use various Monte Carlo chain perturbation approaches, often combined with analytical methods [10,11,3,12]. A good overview of loop closure methods and references can be found in Kolodny et al. (2005) [6].

Most methods assume that one is working with a full-atom protein backbone with fixed bond angles and bond lengths, so the DOF consist solely of the backbone's (φ, ψ) angles. However, in many cases not all the atoms of the protein backbone are present in the model. In particular, a large class of structure prediction, design and in silico folding methods makes use of drastically simplified models of protein structure [13,14].

A protein structure might for example be represented by a chain of Cα atoms or a chain of virtual atoms at the centers of mass of the side chain atoms [15]. In these models, there is obviously no full-atom model of the protein's backbone available.

In the case of Cα-only models, the structure can be described as a sequence of pseudo bonds, pseudo angles θ and pseudo dihedral angles τ [16]. Here, the term 'pseudo' indicates that the consecutive Cα's are not actually connected by chemical bonds. As in the case of the protein's backbone, the pseudo bond lengths can be considered fixed (typically 3.8 Å). In contrast, the pseudo bond angles between three consecutive Cα atoms are most definitely not fixed, but vary between 1.4 and 2.7 radians. Hence, a Cα-only model of N residues can be represented by a sequence of N - 2 pseudo bond angles θ and N - 3 pseudo dihedral angles τ (Figure 1).

Most inverse kinematics approaches assume that the DOF consist only of dihedral angles, and keep the bond angles fixed or semi-fixed. Hence, they cannot be readily applied to the Cα-only case without restricting the search space unnecessarily. In principle, fragment library based methods would apply, but here the problem of data sparsity arises [17,18]. Often, no suitable fragments can be found if the number of residues between the fixed ends becomes too high.

In order to solve the loop closure problem in Cα-space, we extend a particularly attractive approach that was recently introduced by Canutescu & Dunbrack [8]. The algorithm is called Cyclic Coordinate Descent (CCD), and like many other loop closure algorithms it derives from the field of robotics [19]. As pointed out by Canutescu & Dunbrack, the CCD algorithm is meant as a black box method that generates plausible protein segments that bridge two given, fixed endpoints. The final choice is typically made based upon the occurrence of steric clashes, applicable...
constraints (for example side chain conformations) and evaluation of the energy.

The CCD algorithm does not directly generate conformations that bridge a given gap, but alters the dihedral angles of a given starting segment that already overlaps at the N-terminus such that it also closes at the C-terminus. The starting segment can be generated in many ways, for example by using a fragment library derived from real structures or by constructing random artificial fragments with reasonable conformations. Surprisingly, most protein loops can be closed efficiently by CCD starting from artificial loops constructed with random \((\phi, \psi)\) dihedral angles [8].

The CCD algorithm alters the \((\phi, \psi)\) dihedral angles for every residue in the segment in an iterative way. In each step, the RMSD between the chain end and the overlap is minimized by optimizing one dihedral angle. Because only one dihedral angle is optimized at a time, the optimal rotation can be calculated efficiently using simple vector arithmetic.

The list of advantages of CCD is impressive: it is conceptually simple and easy to implement, computationally fast, very flexible (ie. capable of incorporating various restraints and/or constraints) and numerically stable. Therefore, we decided to adopt the CCD algorithm for use with \(C\alpha\)-only models. Here, we describe a new version of CCD that optimizes both dihedral angles and bond angles, while maintaining all the advantages of the CCD method. We call our method Full Cyclic Coordinate Descent (FCCD), where "Full" indicates that both dihedral angles and bond angles are optimized, while only the bond lengths remain fixed. At the heart of the FCCD method lies a procedure to superimpose point sets with minimal Root Mean Square Deviation (RMSD), based on singular value decomposition. As is the case for the CCD algorithm, FCCD is not a modelling method in itself. Rather, it can be used as a method to generate possible conformations that can be evaluated using some kind of energy function.

To test the algorithm, we selected random segments from a protein structure database, and evaluated the efficiency of closing the corresponding gaps starting from artificial segments with protein-like \((\theta, \tau)\) angles. We show that FCCD is both fast and successful in solving the loop closure problem, even in the presence of angle constraints. Conformations close to those of real protein loops are readily generated. Finally, we discuss possible applications of the FCCD algorithm, and mention some possible disadvantages.
Initially, the first three vectors of the moving loop coincide with the first three vectors of the fixed segment, while the last three vectors are conceivably reasonably close to the last three vectors of the fixed loop. This last condition is however not very critical. The moving segment can be obtained using any algorithm that generates plausible Cα fragments, including deriving them from real protein structures. The fixed segment is typically derived from a real protein of interest, or a model in an MCMC simulation.

The FCCD algorithm changes the pseudo bond angles and pseudo dihedral angles of the moving loop in such a way that the RMSD between the last three vectors of the moving loop \((m_{N-3}^N, \quad m_{N-2}^N, \quad m_{N-1}^N)\) and the last three vectors of the fixed loop \((f_{N-3}^N, \quad f_{N-2}^N, \quad f_{N-1}^N)\) is minimized, thereby seamlessly closing the gap.

Note that we assume that the last three vectors of the moving and fixed segments can be superimposed with an RMSD of 0.0 Å (see Figure 2). In other words, the first and last pseudo bond angles in both segments are equal. It is however perfectly possible to use segments with different pseudo bond angles at these positions. Since the final possible minimum RMSD will be obviously greater than 0 in this case, the RMSD threshold needs to be adjusted accordingly.

The algorithm proceeds in an iterative way. In each iteration, a vector \(m_i\) in the moving segment is chosen that will serve as a center of rotation. This chosen center of rotation will be called the pivot throughout this article. Then, the rotation matrix that rotates \((m_{N-3}^N, \quad m_{N-2}^N, \quad m_{N-1}^N)\) on \((f_{N-3}^N, \quad f_{N-2}^N, \quad f_{N-1}^N)\) around the pivot and resulting in minimum RMSD is determined, and applied to all the vectors \(m_i\) downstream \(i\) (with \(i < N\)). In the next iteration, a new pivot is chosen, and the procedure is repeated. The vectors in the chain can be traversed linearly, or they can be chosen at random in each iteration. The difference between FCCD and CCD is that the latter applies a general rotation to the chain using an atom in the chain as a pivot, while the former only applies a rotation around a single axis. The process is stopped when the RMSD falls below a given threshold.

Finding the optimal (with respect to the RMSD) rotation matrix corresponds to finding one optimal pseudo bond angle and pseudo dihedral angle pair. We define \(\theta\) as the bond angle of the vectors \(m_i, \quad m_i, \quad m_{i+1}\) and \(\tau\) as the dihedral angle of the vectors \(m_i, \quad m_{i+1}, \quad m_{i+2}, \quad m_{i+3}\) (see Figure 1 and [16]). These definitions have the intuitive interpretation that altering \((\theta, \quad \tau)\) changes the positions of all Cα downstream from position \(i\). Conversely, using pivot \(m_i\) and applying a rotation matrix to all the positions downstream from position \(i\) corresponds to changing pseudo bond angle \(\theta\) and pseudo dihedral angle \(\tau\).

For a segment of \(N\) Cα's (with \(N > 3\)), the pseudo angles range from \(\theta_1\) to \(\theta_{N-2}\) and the pseudo dihedrals range from \(\tau_1\) to \(\tau_{N-2}\). Since the first and last bond angles of the moving segment are fixed, the pivot points range from position 2 to position \(N - 3\) (with \(N > 4\)). The pseudo bond angle and pseudo dihedral angle pairs thus range from \((\theta_2, \quad \tau_2)\) to \((\theta_{N-3}, \quad \tau_{N-3})\).

Finding the optimal rotation matrix with respect to the RMSD of the C-terminal overlaps can be efficiently solved using singular value decomposition, as described in detail in the following section.

**Finding the optimal rotation**

In this section we discuss solving the following subproblem arising in the FCCD algorithm: given a chosen pivot point \(i\) in the moving segment, find the optimal \((\theta, \quad \tau)\) pair that minimizes the RMSD between the last three Cα vectors in the moving segment and the last three Cα vectors in the fixed segment. Recall that the \((\theta, \quad \tau)\) pair at position \(i\) corresponds to the pseudo bond angles and pseudo dihedral angles defined by vectors \(m_{i-1}, \quad m_i, \quad m_{i+1}\) and \(m_{i+2}, \quad m_{i+3}, \quad m_{i+4}\) respectively.

Finding the optimal \((\theta, \quad \tau)\) pair simply corresponds to finding the optimal rotation matrix using Cα position \(i\) as the center of rotation (see Figure 2). This reformulated problem can be solved by a variant of a well known algorithm to superimpose two point sets with minimum RMSD which makes use of singular value decomposition [20,21]. Below, we describe this adapted version of the algorithm.

First, the C-terminal overlaps of the moving and the fixed segment need to be translated to the new origin that will be used as pivot for the optimal rotation. This new origin is the pivot vector \(m_i\) at Cα position \(i\) in the moving segment. The new vector coordinates of the moving and the fixed segments are put in two matrices (respectively \(M\) and \(F\)), with the coordinates of the vectors positioned column wise:

\[
M = [m_{N-3}^N \cdot m_i | m_{N-2}^N \cdot m_i | m_{N-1}^N \cdot m_i]
\]

\[
F = [f_{N-3}^N \cdot m_i | f_{N-2}^N \cdot m_i | f_{N-1}^N \cdot m_i]
\]

Then, the correlation matrix Σ is calculated using \(M\) and \(F\):

\[
Σ = FM^T
\]
Any real $n \times m$ matrix $A$ can be written as the product of an orthogonal $n \times n$ matrix $U$, a diagonal $n \times m$ matrix $D$ and an orthogonal $m \times m$ matrix $V^T$ [22]. Such a factorization is called a singular value decomposition of $A$. The positive diagonal elements of $D$ are called the singular values. Hence, $\Sigma$ can be written as:

$$\Sigma = UDV^T$$

The optimal rotation $\Gamma$ is then calculated as follows:

$$\Gamma = USV^T$$

The value of the diagonal $3 \times 3$ matrix $S$ is determined by the product $\det(U)\det(V^T)$, which is either 1 or -1. If this product is -1 then $S = \text{diag}(1, 1, -1)$, else $S$ is the $3 \times 3$ unit matrix. The matrix $S$ ensures that $\Gamma$ is always a pure rotation, and not a rotation-inversion [21].

In order to apply to all the vectors that are downstream from the pivot point $i$, these vectors are first translated to the origin of the rotation (ie. pivot point $m_i$), left multiplied by $\Gamma$ and finally translated back to the original origin:

$$m'_i = \Gamma(m_j - m_i) + m_i$$

where $i < j < N$.

Adding angle constraints to FCCD

It is straightforward to constrain the $(\theta, \tau)$ angles to a given probability distribution. For each rotation matrix $\Gamma$, the resulting new pseudo bond angles and dihedral angles can easily be calculated. The new angles can for example be accepted or rejected using a simple rejection sampling Monte Carlo scheme, comparing the probabilities of the previous pair $(\theta^{prev}, \tau^{prev})$ with that of the next pair $(\theta^{next}, \tau^{next})$. If $P(\theta^{next}, \tau^{next}) > P(\theta^{prev}, \tau^{prev})$ the change is accepted, otherwise it is accepted with a chance proportional to $P(\theta^{next}, \tau^{next}) / P(\theta^{prev}, \tau^{prev})$. A similar approach was used by Canutescu & Dunbrack [8], and we describe its performance in combination with FCCD in the following section.

More advanced methods could take the probability of the sequence of angles into account as well, for example using a Hidden Markov Model of the backbone [23]. The pseudo code in Table 3 illustrates accepting/rejecting rotations using an unspecified 'accept' function, whose details will depend on the application.

**FCCD's performance**

In order to evaluate the general efficiency of the method, we selected random fragments of various sizes from a representative database of protein structures, and used these fragments as fixed segments. Hence, the evaluation described below is not limited to loops, but extends to random protein segments. This is a relevant test, since local moves in a typical MCMC simulation are indeed performed on random segments.

The fixed segments were sampled from a dataset of fold representatives (see Methods). First we selected a random fold representative, and subsequently extracted a random continuous fragment of suitable length. The lengths varied from 10 to 30 with a step size of 5. It should be noted that the length of the segment here refers to the number of $\alpha$-atoms between the ends that need to be bridged.

The moving segments were generated using random dihedral and bond angles in regions accessible to proteins (see previous section). This was done by sampling the $(\theta, \tau)$ pairs according to a probability distribution derived from a set of representative protein structures (see Methods). The bond length was fixed at 3.8 Å, in tune with the consensus $\alpha$-$\alpha$ distance in protein structures. The last bond angle in the moving segment was chosen equal to the last bond angle in the fixed loop to make a final RMSD of 0.0 Å possible. The RMSD threshold was 0.1 Å. The maximum number of iterations was set to 1000, where one iteration is a sweep over all positions. We ran the FCCD program on 1000 different fixed segments. Table 1 summarizes the results.

| Segment length | Average time (ms) | Average iterations | % Closed |
|----------------|-------------------|--------------------|----------|
| 5              | 4.5/51.7          | 14.0/27.0          | 99.90/86.50 |
| 10             | 5.2/28.3          | 10.5/16.8          | 99.40/98.20 |
| 15             | 5.6/28.6          | 7.8/12.1           | 99.60/99.40 |
| 20             | 6.2/27.1          | 6.3/9.0            | 99.80/99.40 |
| 25             | 7.6/31.7          | 5.5/7.6            | 99.00/99.90 |
| 30             | 7.1/31.0          | 4.4/6.3            | 99.70/99.40 |
A first observation is the effect of the angle constraints. These slow down FCCD with a factor of 10 for small segments (5 residues) and roughly a factor of 5 for larger segments (10 residues or more). Nonetheless FCCD including constraints remains quite speed efficient: small five residue segments are on average closed in about 50 ms, while larger segments (from 10 to 30 residues) are closed considerably faster (on average in about 30 ms). The explanation for this is of course that it is easier to close large segments because they have more DOF. Hence, FCCD, like CCD, is fast and easily handles large segments efficiently.

Overall, the success rate of FCCD is excellent, and very little affected by constraints. For 5 residue segments, adding constraints diminishes the number of successfully closed segments from 99.9% to 86.5%. This effect is however much less pronounced for larger segments: more than 98% percent of the moving/fixed segment pairs can be successfully closed. In short, FCCD is both speed efficient and has a high success rate, even in the presence of constraints.

**Evaluation of FCCD’s sampling space**

Does FCCD potentially generate realistic protein conformations? FCCD could be used to propose possible conformations that are subsequently evaluated by an energy function. In this context, it is of course imperative to generate realistic conformations. To answer this question, we evaluate FCCD’s ability to generate closed segments that are close to real protein loops. We used 30 real loops with lengths of 4, 8 and 12 residues as fixed segments. The loop length refers to the number of residues between the N- and C-terminal overlaps.

FCCD was applied using \((\theta, \tau)\) constraints and an RMSD threshold of 0.1 Å. The maximum number of iterations was set to 1000. For each loop, we attempted to generate closed segments from 1000 random moving segments within the allowed number of iterations. The moving segments were generated as described in the previous section. For all 30 loop cases, we then identified the closed segment that resembled the input loop best as judged by the RMSD. For the calculation of the RMSD, we included the N-and C-terminal overlaps. The results are shown in Table 2, and the best fitting loops for each loop size are shown in Figure 3.

It is clear that FCCD readily generates closed segments that are reasonably close to the real loops, with an average RMSD of about 0.6, 2.2 and 3.0 Å for loops of 4, 8 and 12 residues, respectively. The highest minimum RMSD values for these loop lengths are 0.76, 2.42 and 3.37 Å, respectively, indicating that FCCD in general can come up with a reasonably close conformation. Using more initial moving segments will obviously increase the chance of encountering a close conformation. Additionally, one can also expect an even better performance with a more refined way to constrain the \((\theta, \tau)\) angles.

**Conclusion**

In this article, we introduce an algorithm that solves the loop closure problem for \(\text{Ca}\) only protein models. The method is conceptually similar to the CCD loop closure method introduced by Canutescu and Dunbrack [8], but optimizes dihedral and bond angles simultaneously, while the former method only optimizes one angle at a time. At the heart of the method lies a modified algorithm to superimpose point sets with minimum RMSD, based on singular value decomposition [20,21].

The algorithm is fast, numerically stable and leads to a solution for the great majority of loop closure problems studied here. Importantly, the method remains efficient even in the presence of constraints on the dihedral and bond angles. FCCD readily handles large gaps, and potentially generates realistic conformations. Compared to other loop closure methods, FCCD is surprisingly easy to implement provided a function is available to calculate the singular value decomposition of a matrix.

A possible disadvantage is that FCCD has a tendency to induce large changes to the pseudo angles at the start of the moving segment while angles near the end are less affected, which is also the case for CCD [8]. This can for example be avoided by selecting the pivot points in a random fashion, or by limiting the allowed change in the angles per iteration. Occasionally the method gets stuck, which can be avoided by incorporating stochastic changes away from the encountered local minimum. One can also simply try again with a new random moving segment. We believe that CCD and FCCD despite these disadvantages are among the most efficient loop closure algorithms currently available.

The FCCD algorithm proposed here has great potential for use in structure prediction methods that only make use of \(\text{Ca}\) atoms, or that otherwise do not include all backbone atoms [15,13,14]. FCCD could be used for example to implement local moves in a MCMC procedure. The moving segments could be derived from a fragment database or generated from a probabilistic model of the protein backbone. The latter model could range from a primitive probability distribution over allowed \((\theta, \tau)\) angle pairs like we used here to a Hidden Markov Model that also models the sequence of \((\theta, \tau)\) angle pairs.

We are planning to use the FCCD algorithm in combination with a sophisticated probabilistic model of the protein’s backbone, which will steer both the generation of
Loops generated by FCCD (blue) that are close to real protein loops (green). The loops with lowest RMSD to a given loop of length 4 (top), 8 and 12 (bottom) are shown (loops 1qnr, A, 195–198, 3chb, D, 51–58 and 1ctq, A, 26–37). The N-terminus is at the left hand side.
Table 2: Minimum RMSD (out of 1000 tries) between a fixed segment derived from a protein structure and a closed segment generated by FCCD. The length of the loops is shown between parentheses in the upper row.

| Loop (4)  | RMSD | Loop (8)  | RMSD | Loop (12) | RMSD |
|-----------|------|-----------|------|-----------|------|
| 1dvj, A, 20–23 | 0.59 | 1cru, A, 85–92 | 2.31 | 1cru, A, 358–369 | 3.37 |
| 1dys, A, 47–50 | 0.67 | 1ctq, A, 144–151 | 2.22 | 1ctq, A, 26–37 | 2.40 |
| 1egu, A, 404–407 | 0.61 | 1dbw, A, 334–341 | 2.04 | 1d4o, A, 88–99 | 3.20 |
| 1ej0, A, 74–77 | 0.61 | 1ds1, A, 20–27 | 2.20 | 1d8w, A, 43–54 | 2.74 |
| 1i0h, A, 123–126 | 0.73 | 1glk, A, 122–129 | 2.20 | 1ds1, A, 282–293 | 3.16 |
| 1id0, A, 195–198 | 0.54 | 1lxh, A, 106–113 | 1.98 | 1egu, A, 508–519 | 3.06 |
| 1tca, A, 95–98 | 0.76 | 1qop, B, 14–21 | 2.17 | 1q1w, A, 31–42 | 3.04 |
| 1thf, D, 121–124 | 0.56 | 3chb, D, 51–58 | 1.97 | 1qop, A, 175–186 | 2.97 |

Average RMSD 0.63 Average RMSD 2.17 Average RMSD 3.00

Table 3

maxit = maximum number of iterations
moving = N × 3 matrix of Cα positions in moving segment
fixed = N × 3 matrix of Cα positions in fixed segment
threshold = desired minimum RMSD
N = length of the segments
M = 3 × 3 matrix (centered coordinates along columns)
F = 3 × 3 matrix (centered coordinates along columns)
S = diag(1, 1, -1)

repeat maxit:

# Start iteration over pivots
for i from 2 to N-3:
    pivot = moving[i,:]
    # Make pivot point origin
    for j from 0 to 2:
        M[,] = moving[N-3+j,:]-pivot
        F[,] = fixed[N-3+j,:]-pivot
        # Find the rotation Γ that minimizes RMSD
        Σ = FM*
        U, D, Vt = svd(Σ)
        # Check for reflection
        if det(U)det(Vt)<0:
            U = US
            Γ = UVt
            # Evaluate and apply rotation
            if accept(Γ):
                # Apply the rotation to the moving segment
                for j from i+1 to N-1:
                    moving jugg = Γ (moving jugg-pivot)+pivot
                rmsd = calc_rmsd(moving [N-3,:], fixed [N-3,:])
                # Stop if RMSD below threshold
                if rmsd<threshold:
                    return moving, rmsd
            # Failed: RMSD threshold not reached before maxit
        return 0

The accept function rejects or accepts the proposed rotation, based on the resulting (θ, τ) pair. The svd function performs singular value decomposition, and calc_rmsd calculates the RMSD between two lists of vectors.
the initial moving loop and the acceptance/rejection of the angles. The performance of FCCD in this context will be the subject of a future publication.

**Methods**

**Implementation**

The FCCD algorithm was implemented in C, using the LAPACK [24] function dgesvd for the calculation of the singular value decomposition. Handling PDB files and calculating the \((\theta, \tau)\) angles [16] was done using Biopython’s Bio.PDB module [25]. We used a 2.5 GHz Pentium processor to calculate the benchmarks. A reference implementation of FCCD in Python is available as supplementary information.

**Structure databases**

For the calculation of the \((\theta, \tau)\) probability distribution and the generation of random protein fragments, we used the SABMark 1.63 Twilight Zone database [26]. SABMark Twilight Zone contains 2230 high quality protein structures, divided over 236 different folds. All protein pairs have a BLAST E-value below 1, and thus presumably belong to different superfamilies. A dataset of fold representatives was generated by selecting a single structure at random for each fold (see Table 4).

The loops used to evaluate FCCD’s sampling space were derived from Canutescu & Dunbrack [8]. We shifted two loops (1d8w, A, 46–57 and 1qop, A, 178–189) by three residues to ensure that all loops had three flanking residues on each side.

**Calculation of the \((\theta, \tau)\) probability distribution**

The bond angle \(\theta\) was subdivided in 18 bins and the dihedral angle \(\tau\) in 36 bins, in both cases starting at 0 degrees and with a bin width of 10 degrees. All \((\theta, \tau)\) angles were extracted from all structures in the SABMark Twilight Zone database that consisted of a polypeptide chain without breaks. In total, 257534 angle pairs were extracted. Each such \((\theta, \tau)\) angle pair was assigned to a bin pair, and the number of angle pairs assigned to each bin pair was stored in a 18 × 36 count matrix. Finally, the normalized count matrix was used to assign a probability to any given \((\theta, \tau)\) angle pair.

**List of abbreviations**

- CCD: Cyclic Coordinate Descent
- DOF: Degrees Of Freedom
- FCCD: Full Cyclic Coordinate Descent
- MCMC: Markov Chain Monte Carlo
- RMSD: Root Mean Square Deviation

| Table 4: SABMark identifiers of the 236 structures used as fold representatives |
|---------------------------------|---------------------------------|
| 1ew6a_ 1ail_ 1l1la_ 1kid_ 1n8ycl_ 1ghbl_ 1ge5da_ 1ep3b2_ 1lihoa_ 1m0wa1 |
| 1dhs_ 1gqua2_ 2lefa_ 1nstya_ 1leaf_ 1li6a_ 1d5ra2_ 1foha3_ 1gqua3_ 1crza2 |
| 3pvia_ 1i6pa_ 1e4tf1_ 1xk5d_ 2pth_ 1lu9a2_ 1dkia_ 1f6ga_ 1m2o3a_ 2dpmia_ |
| 1ajsa_ 1fsooa_ 3xgl_ 1x64a_ 1mtyg_ 1dugv2_ 1lqopb_ 1lata_ 1k2y2s2_ 1f0ka_ |
| 1ayl_ 1tboa_ 8ahp_ 1n8h8a_ 1bi5a2_ 2mhr_ 1a2pa_ 3ltz_ 1dki_ 167ia2 |
| 1b4fa_ 1bb8_ 1ikpf_ 1mnu5a2_ 1lfxa_ 1gpea2_ 1lqca_ 1a2vxa2_ 1lfma_ 1i77a2 |
| 1jcxa1_ 1lo7a_ 1fmo0e_ 1fbsb2_ 1o0wa2_ 1ldjia_ 1ik0ra3_ 1levs3_ 1lpdx2_ 1qda1 |
| 1d5ya3_ 1h3sa2_ 1qo03_ 1tig_ 1lxxaa_ 1ck9a_ 1lgxas_ 1e5q2a_ 1lvxa2_ 1qbea |
| 3g3s_ 1f08a_ 1c7ka_ 1lkka_ 1dq3a3_ 1uox_ 1zasa_ 1bop_ 1m4ja_ 1d5v5a |
| 1f5ma_ 1k2ea2_ 1e1a2_ 1ldw_ 1ln1a_ 2pola2_ 1f0lai_ 1r16a_ 1fviates_ 1i77a |
| 1is2a1_ 1e8g2a_ 1qroa1_ 2dnja_ 1kuua_ 1qh5a_ 1li7a_ 1bip2a_ 1j7ma3_ 1hcha3 |
| 1f003_ 1lgrl_ 1nkzd_ 1mwx3a_ 1lp4a_ 1ih7a2_ 1lue2a_ 1glna_ 1lma_ 2por |
| 4htci_ 1es7b_ 1tcr_ 1d1la_ 1fd3a_ 1i8na_ 1h8pa1_ 4sgbi_ 1f1tv_ 1quba1 |
| 1d4va3_ 1tpg_ 2_ 1iuua_ 1f5va_ 1mdya_ 1lmc1_ 1fgn_ 1leska_ 1li502_ 1bfa4 |
| 1edm_ 1es3a_ 1ezvb1_ 1jqa1_ 1k3ea_ 1rec_ 1lm5a_ 1k82a1_ 1laja_ 1lm0ka |
| 1cova_ 1kgfc_ 1ocroc_ 1h67a_ 2cpga_ 1ljra_ 1lbwa1_ 1h5a_ 2cb2a_ 1ljm2a |
| 1hyp_ 1cuk2_ 1ecwa_ 1h9la_ 1g7da_ 1ljkw_ 1ldglg_ 1lipsa_ 1pa2a_ 1ico9a1 |
| 1f1za1_ 1ks9a1_ 2sqa2_ 1d2ta_ 1h3a_ 1lwer_ 1b3ua_ 1n1ba2_ 1poc_ 1e79i |
| 1mlqga_ 1lenwa_ 14m4a1_ 1e5ba_ 1qhoa2_ 1vk7a2_ 1hi4a2_ 1chda_ 1lam1_ 1ca1_2 |
| 1phm_ 2_ 1d7pm_ 1jcb2_ 1fcia_ 1gr3a_ 1mjia_ 1ria7d_ 1il8s2_ 1fita_ 1jboe |
| 1h2a_ 1lcya1_ 1mgqa_ 1hcia_ 1lb3qa2_ 1ljka_ 1ldar_ 1leaxma2_ 1lejea_ 1lag |
| 1e79d2_ 2hrpa_ 1ho0a1_ 1gr1a_ 2erl_ 1bta_ 1f7ha_ 1jmxsa_ 1lcrsa_ 1lm1xa4 |
| 1hxoa1_ 1g0a1_ 1ciy_ 2_ 1ldba_ 3tcl_ 1gg3a1_ 1lpmi_ 1bdo_ 1h3s2a_ 1gppa |
| 1f9ya_ 1k6wa1_ 1ljna_ 1lu9a_ 1m6a1_ 1o94a3_ 1o94a3_ 1o94a3_ 1o94a3_ 1o94a3 |
Authors' contributions
TH conceived the FCCD algorithm. WB implemented FCCD in the C language, and introduced various refinements and optimizations. Both authors read and approved the article.

Additional material

Additional File 1
The file FCCD.py contains an implementation of the FCCD algorithm. The program was implemented in the interpreted, object oriented language Python http://www.python.org. The Numeric Python package http://[numeric scipy.org], a Python module that implements many advanced mathematical operations efficiently in C and FORTRAN, provided implementations of singular value decomposition and various matrix operations. In addition, the Biopython toolkit, a set of Bioinformatics modules implemented in Python, was used to represent atomic coordinates as vector objects [25]. The core of the FCCD implementation comprises only 50 lines of Python code. Numeric Python and Biopython (version 1.4b) are needed to execute the sample code.

Click here for file [http://www.biomedcentral.com/content/supplementary/1471-2105-6-159-S1.py]

Acknowledgements
Wouter Boomsma is supported by the Lundbeckfond http://www.lundbeckfonden.dk/. Thomas Hamelryck is supported by a Marie Curie Intra-European Fellowship within the 6th European Community Framework Programme. We acknowledge encouragement and support from Prof. Anders Krog, Bioinformatics Center, Institute of Molecular Biology and Physiology, University of Copenhagen.

References
1. Go N, Scheraga H: Ring closure and local conformational deformations of chain molecules. Macromolecules 1970, 3:178-187.
2. Koehler P, Delarue M: A self consistent mean field approach to simultaneous gap closure and side-chain positioning in homology modelling. Nat Struct Biol 1995, 2:163-70.
3. da Silva R, Degreve L, Caliri A: LMProt: an efficient algorithm for Monte Carlo sampling of protein conformational space. Biophys J 2004, 87:1567-77.
4. Jones T, Thirup S: Using known substructures in protein model building and crystallography. EMBO J 1986, 5:19-22.
5. Rohl C, Strauss C, Chivian D, Baker D: Modeling structurally variable regions in homologous proteins with rosetta. Proteins 2004, 55:656-77.
6. Kolodny R, Guibas L, Levitt M, Koehler P: Inverse kinematics in biology: The protein loop closure problem. Int J Robotics Research 2005, 24:151-163.
7. Manecha D, Canby J: Efficient inverse kinematics for general 6R manipulators. IEEE Trans Rob Act 1994, 10:648-657.
8. Canutescu A, Dunbrack R Jr: Cyclic coordinate descent: A robotics algorithm for protein loop closure. Protein Sci 2003, 12:963-72.
9. Coutsias E, Seok C, Jacobson M, Dill K: A kinematic view of loop closure. J Comput Chem 2004, 25:510-28.
10. Favrin G, Irbäck A, Sjunnesson F: Monte carlo update for chain molecules: Biased Gaussian steps in torsional space. J Chem Phys 2001, 114:8154-8158.
11. Cahill M, Cahill S, Cahill K: Proteins wriggle. Biophys J 2002, 82:2665-70.
12. Singh R, Bergert B: Chaintweak: sampling from the neighbourhood of a protein conformation. Pac Symp Biocomput 2005 [http://pacbio-web.stanford.edu/pac05/singh.pdf].
13. Buchete N, Straub J, Thirumalai D: Development of novel statistical potentials for protein fold recognition. Curr Opin Struct Biol 2004, 14:225-32.
14. Tozzini V: Coarse-grained models for proteins. Curr Opin Struct Biol 2005, 15:144-50.
15. Khaira D, Lu H, Kolinski A, Skolnick J: TOUCHSTONE: an ab initio protein structure prediction method that uses threading-based tertiary restraints. Proc Natl Acad Sci USA 2001, 98:10125-30.
16. Oldfield T, Hubbard R: Analysis of C alpha geometry in protein structures. Proteins 1994, 18:324-37.
17. Fidelis K, Stern P, Bacon D, Moult J: Comparison of systematic search and database methods for constructing segments of protein structure. Protein Eng 1994, 7:953-60.
18. van Vlijmen H, Karpplus M: PDB-based protein loop prediction: parameters for selection and methods for optimization. J Mol Biol 1997, 267:975-1001.
19. Wang L, Chen C: A combined optimization method for solving the inverse kinematics problem of mechanical manipulators. IEEE Trans Rob Aut 1991, 7:489-499.
20. Kabsch W: A discussion of the solution for the best rotation to relate two sets of vectors. Acta Cryst 1978, A34:827-828.
21. Umeyama S: Least squares estimation of transformation parameters between two point patterns. IEEE Trans Pattern Anal Mach Intell 1991, 13:376-80.
22. Golub GH, Loan CFV: Matrix Computations 3rd edition. Baltimore, Maryland: Johns Hopkins University Press; 1996.
23. Bystruff C, Thorsson V, Baker D: HMMSTR: a hidden Markov model for local sequence-structure correlations in proteins. J Mol Biol 2000, 301:173-90.
24. Anderson E, Bai Z, Bischof C, Demmel J, Dongarra J, Croz JD, Greenbaum A, Hammarling S, McKenney A, Ostrouchov S, Sorensen D: LAPACK’s user’s guide 2nd edition. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics; 1995.
25. Hamelryck T, Manderick B: SABmark-a benchmark for sequence alignment that covers the entire known fold space. Bioinformatics 2005, 21:1267-8.