BIOINFORMATICS ORIGINAL PAPER Vol. 28 no. 4 2012, pages 510–515
doi:10.1093/bioinformatics/btr692

Structural bioinformatics

Advance Access publication December 22, 2011

Fast large-scale clustering of protein structures using Gauss integrals

Tim Harder[1], Mikael Borg[1], Wouter Boomsma[2], Peter Røgen[3] and Thomas Hamelryck[1,*]

[1]The Bioinformatics Section, Department of Biology, University of Copenhagen, Copenhagen, [2]Department of Electrical Engineering and [3]Department of Mathematics, Technical University of Denmark, Lyngby, Denmark

Associate Editor: Anna Tramontano

ABSTRACT

Motivation: Clustering protein structures is an important task in structural bioinformatics. *De novo* structure prediction, for example, often involves a clustering step for finding the best prediction. Other applications include assigning proteins to fold families and analyzing molecular dynamics trajectories.

Results: We present Pleiades, a novel approach to clustering protein structures with a rigorous mathematical underpinning. The method approximates clustering based on the root mean square deviation by first mapping structures to Gauss integral vectors—which were introduced by Røgen and co-workers—and subsequently performing K-means clustering.

Conclusions: Compared to current methods, Pleiades dramatically improves on the time needed to perform clustering, and can cluster a significantly larger number of structures, while providing state-of-the-art results. The number of low energy structures generated in a typical folding study, which is in the order of 50 000 structures, can be clustered within seconds to minutes.

Contact: thamelry@binf.ku.dk; harder@binf.ku.dk

Supplementary Information: Supplementary data are available at *Bioinformatics* online.

Received on June 8, 2011; revised on December 5, 2011; accepted on December 9, 2011

1 INTRODUCTION

Predicting the 3D fold for a given amino acid sequence remains one of the greatest challenge in computational biology today. While many of the driving forces behind the folding process are understood in principle, their calculation in the context of a folding study is still challenging. Most *de novo* prediction methods rely on sampling a large number of decoy structures from an approximate force field. Due to the effects of configurational entropy and of the folding pathway, the native structure is not necessarily the structure with the lowest energy, even with a perfect physical force field.

Shortle *et al.* (1998) point out that decoy clustering is the correct way to select the final prediction, since the native structure is typically located near the bottom of a broad well in the energy landscape. They showed that selecting the centroid structure of the largest cluster readily results in structures that are closer to the native structure than the structure with the lowest energy. Here, the centroid is the structure with the lowest average distance to all other structures in the cluster. Modern computer hardware allows to generate tens of thousands of low energy decoys in a few hours. These developments underline the demand for software that is able to clusters tens of thousands of structures in seconds or minutes.

There are several clustering programs available in the public domain. They usually rely on an atomic representation of the decoys and subsequently cluster based on the pairwise root mean square deviation (RMSD), a standard measure of structure similarity (Kabsch, 1976, 1978; Theobald and Wuttke, 2006). The most popular programs, Calibur (Li and Ng, 2010), Durandal (Berenger *et al.*, 2011), Scud (Li and Zhou, 2005) and Spicker (Zhang and Skolnick, 2004) follow the Rosetta/iTasser procedure (Simons *et al.*, 1997; Wu *et al.*, 2007) and perform *exact clustering*. The structure with the most neighbors within a given cutoff is chosen as the first centroid. The selected structure, along with all neighbors, is then removed from the set and the procedure is repeated to find the next largest cluster. Calibur applies pruning to exclude a large part of the decoy set from the costly pairwise comparison using auxiliary grouping with upper and lower bounds. Durandal similarly avoids large amounts of the pairwise calculations in the initial step of filling the distance matrix, by *propagating* the information gained in the exact measurement: knowing the RMSD values of structure pairs (A, B) and (A, C) allows to infer information about the RMSD value of (B, C). Both methods are substantially faster than Spicker, which evaluates all pairwise distances. Scud also applies exact clustering, but uses a *reference root-mean-squared distance* to calculate the pairwise distances. Here, all structures are aligned to a randomly selected reference structure. The approximate RMSD between two structures is then calculated based on their respective RMSD values with that reference structure, thereby avoiding the cost of an explicit superposition.

With the exception of Spicker (Zhang and Skolnick, 2004), the number of structures that can be clustered is not limited by any of the programs. However, extensive memory consumption becomes a limiting factor with growing numbers of decoys, as reloading the structures from the hard disk leads to a significant drop in performance. Currently, with a common desktop computer, 10 000 to 30 000 structures can be clustered, depending on the amount of main memory available (Berenger *et al.*, 2011; Li and Ng, 2010).

Here we present Pleiades, a novel approach to protein structure clustering that uses Gauss integrals (Røgen and Bohr, 2003; Røgen and Fain, 2003; Røgen, 2005) to represent a protein's 3D structure. Gauss integrals allow to represent structures by 31-dimensional vectors, whose pairwise Euclidean distances correlate well with the RMSD of the corresponding structures (Lindorff-Larsen *et al.*, 2005). Using this efficient representation, Pleiades is able to quickly

*To whom correspondence should be addressed.

cluster a large number of protein structures, improving drastically on existing methods.

2 IMPLEMENTATION

2.1 Representing proteins using Gauss integrals

In order to describe the overall fold of a protein structure, it is usually sufficient to consider the so-called C_α trace, which consists of the segments that connect the C_α carbon atoms of the protein backbone. More detailed information such as the exact side chain conformations are of less importance for describing the fold.

Røgen *et al.* (Lindorff-Larsen *et al.*, 2005; Røgen and Bohr, 2003) introduced a simplified description of the 3D fold by interpreting the protein's C_α trace as an oriented open curve in space, and subsequently calculating a series of generalized Gauss integrals. In a planar projection of a curve, one can count the number of crossings and, using a handedness rule, also the *signed* number of crossings. The two simplest Gauss integrals, called *writhe* and *average crossing* number, are the signed and unsigned number of crossings averaged over all planar projections of the curve. The writhe and average crossing number are natural measures of how 'entangled' a curve is. The higher order Gauss integrals measure specific patterns of entanglement along the backbone; for almost planar curves they count how often a particular pattern of two or more crossings occurs (Røgen, 2005). A protein's C_α trace can thus be represented as a vector of Gauss integrals. Røgen *et al.* (Røgen, 2005) showed that for two given proteins, the Euclidean distance between their Gauss integral vectors correlates well with the RMSD.

Pleiades uses an enhanced version of the measure called the *tuned Gauss integral* (GIT). The inspection of 24 000 high-resolution domains from the CATH database (Orengo *et al.*, 1997) revealed both a correlation between different Gauss integrals as well as a dependency on the length of the amino acid chain. This analysis led to an empirical correction factor, reducing the length dependency, that is applied to each Gauss integral. Moreover, some Gauss integrals can be predicted from lower order integrals and the length of the protein. In these cases, the estimated value is removed to limit the internal correlation. Lastly, calculating a smoothed curve from the otherwise rugged C_α trace further decreases the correlation between different Gauss integrals and improves the signal to noise ratio. Overall, the use of the GIT measure leads to a significantly better correlation with the RMSD, especially when the compared proteins have different lengths (Røgen, 2005).

The result is a 31-dimensional GIT vector that describes a protein's fold. The distance between two structures is simply the Euclidean distance between the two vectors $\bar{x}$ and $\bar{y}$:

$$\mathrm{d}(\bar{x},\bar{y}) = \sqrt{\sum_{i=1}^{31}(x_i - y_i)^2} \tag{2.1}$$

where $\bar{x}$ and $\bar{y}$ are the GIT vectors calculated from structures X and Y, respectively; x_i and y_i is the i-th component of $\bar{x}$ and $\bar{y}$, respectively; and the sum runs over all 31 components of the vector.

While the GIT measure accurately describes a protein's fold, it is not possible to derive a unique 3D structure from a GIT vector: different structures may map to the same GIT vector. Formally, the set of proteins and the GIT vector distance form a pseudometric space. Unlike in a metric space, in a pseudometric space the distance between two different points can be zero. However, the strong correlation between the two measures—especially at low RMSD values—indicates that the GIT distance adequately approximates the RMSD distance for our purposes.

2.2 *K*-means clustering

The K-means algorithm was first introduced by Lloyd (1982) and has since then been used in a variety of applications (Jain, 2010; Steinley, 2006). The algorithm itself is straightforward and has proven to be especially useful in high-dimensional spaces due to its speed.

The distance matrix required for *exact clustering* is an $N \times N$ matrix where N is the number of points in the dataset. For performance reasons, it is desirable to store at least the upper triangle of the matrix in main memory, as this allows fast access. For large datasets, however, the size of the matrix easily exceeds the amount of available memory. For K-means clustering, it is not necessary to store this distance matrix, allowing the clustering of significantly larger datasets.

The algorithm consists of the following steps: (i) choose k initial cluster centers at random from the entire dataset. (ii) Assign each data point to its nearest cluster. (iii) Compute new cluster centroids for all clusters, given their current members. (iv) Repeat (ii) and (iii) until convergence.

The total energy of a cluster is then defined as

$$\mathrm{E}(C) = \frac{\sum_j \mathrm{d}(\bar{c}_j, \bar{c}_c)}{|C| - 1}$$

where C is the cluster, $\bar{c}_j$ is the j-th member of C, $\bar{c}_c$ is the centroid of cluster C, $|C|$ is the cluster size, d is given by Equation (2.1) and the sum runs over all members of cluster C.

2.3 *K*-means++

The K-means algorithm is non-deterministic; the result depends on the initial seed of cluster centroids, which are selected randomly. Both performance and convergence time can be improved by selecting the initial seeds more carefully (Arthur and Vassilvitskii, 2007). Arthur *et al.* (Arthur and Vassilvitskii, 2007) suggest to select the initial cluster centroids by sampling seeds dependent on the distance to all seeds selected so far.

The initial seeding procedure is modified as follows: (i) select one initial cluster randomly from the entire dataset; (ii) sample the next centroid, as explained below; (iii) repeat step (ii) until k cluster centroids have been selected. After this step, the algorithm is identical to the regular K-means procedure as described in the previous section. In our case, the next centroid is sampled with probability:

$$\mathrm{P}(\bar{c}_i) = \frac{\mathrm{d}(\bar{c}_i, \bar{c}_{c,i})}{\sum_j \mathrm{d}(\bar{c}_j, \bar{c}_{c,j})^2}$$

where $\bar{c}_i$ is the GIT vector representing the i-th structure; $\mathrm{d}(\bar{c}_i, \bar{c}_{c,i})$ is the distance between the i-th structure and its nearest cluster centroid $\bar{c}_{c,i}$; and the sum runs over all structures. In other words, the probability of selecting a certain structure as another seed is proportional to the distance to the current nearest cluster center.

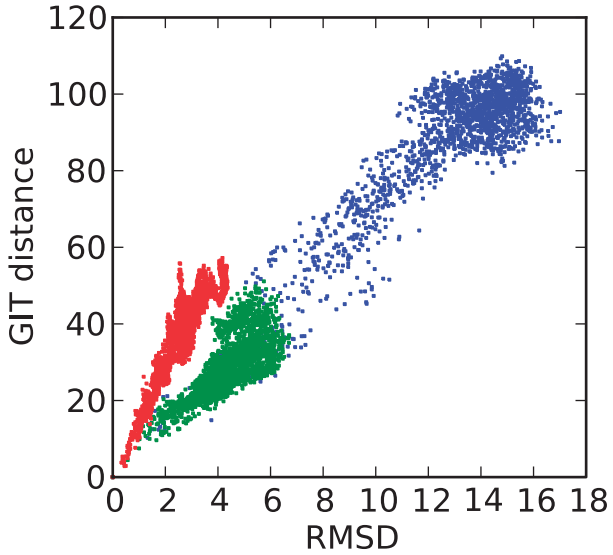

Fig. 1. Correlation between RMSD and GIT distances. The native ensembles of *Candida antarctica* Lipase B (CalB) and adenylate kinase are shown in red and green, respectively. For each decoy structure, the GIT distance versus the RMSD to the native structure is plotted. The Pearson's correlation coefficient is 0.90 for the CalB ensemble and 0.74 for the adenylate kinase ensemble. The unfolding ensemble of adenylate kinase is shown in blue. The Pearson's correlation coefficient is 0.90.

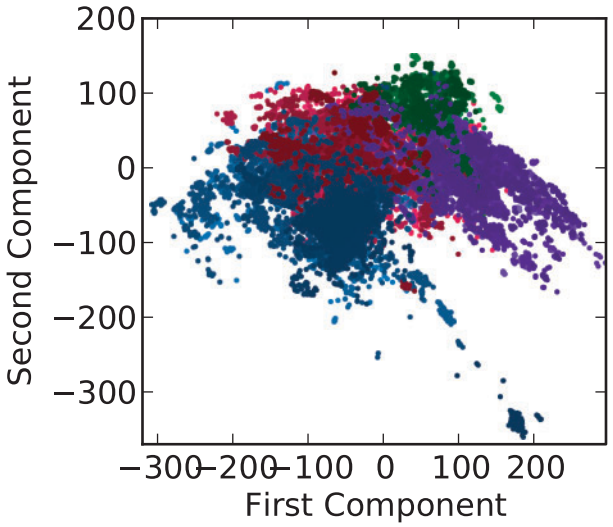

Fig. 2. The SCOP database as GIT vector projection. For each structure, we plot the projections of the first and second component of a principal component analysis. The colors encode the different fold classes; all α folds in shades of green; all β folds in shades of blue; α/β folds in shades of purple; and $\alpha+\beta$ folds in shades of red. The darkness of the shades increases with the number of structures in the family.

3 DISCUSSION

3.1 Properties of the GIT measure

We start with two tests to illustrate that GIT vectors are an appropriate measure of structural similarity. In order to illustrate the correlation between RMSD and the GIT distance, we analyzed structures from three ensembles involving two different proteins (Fig. 1). The ensembles were obtained by starting from the native structure and re-sampling parts of the structure guided by probabilistic models of backbone and side chain conformations (Boomsma *et al.*, 2008; Harder *et al.*, 2010). The first protein, adenylate kinase (PDB 4AKE) (Müller *et al.*, 1996), catalyzes the AMP + ATP $\Leftrightarrow$ 2ADP reaction and undergoes significant conformational changes during its catalytic cycle, making it a prime target for protein dynamics simulations. The protein is 214 residues long and has an α/β fold. The second protein, *Candida antarctica* lipase B (CalB), is an enzyme with industrial applications, (PDB: 1TCA) (Uppenberg *et al.*, 1994). The CalB protein also has an α/β fold, but is significantly longer with 317 residues. For two ensembles, the compactness and the hydrogen bond network of the native state were enforced using Gaussian restraints, resulting in a 'native ensemble'. For the third ensemble, no such restraints were imposed, resulting in an 'unfolding ensemble'. All simulations were performed using the PHAISTOS package (Borg *et al.*, 2009) and generated 1000 decoys each. Details on the simulation setup can be found in the Supplementary Material. Figure 1 illustrates the high correlation between the GIT distances and the corresponding RMSD values, especially for highly similar structures. The Pearson's correlation coefficient is 0.90 for the adenylate kinase unfolding ensemble, 0.74 for the native ensemble of the same protein and 0.90 for the native ensemble of CalB.

In a second test, we investigate whether GIT can detect structural similarities in a diverse set of protein structures, in terms of both fold and protein length. First, we converted the entire Structural Classification of Proteins (SCOP) (Murzin *et al.*, 1995) database into GIT vectors. We then removed all families with 30 or less members. The final dataset contained GIT vectors calculated from 52 876 structures. Figure 2 shows a projection of the SCOP dataset of GIT vectors after principal component analysis (Bishop, 2006). As expected, the all α and all β fold classes are indeed well separated, with the mixed α and β classes located in between the two.

3.2 Rebuilding the SCOP hierarchy

The SCOP database organizes known protein structures into a hierarchy describing the folds (Murzin *et al.*, 1995). The SCOP database is organized in four hierarchy levels—class, fold, superfamily and family—and contains a total of 110 800 domains from 38 221 Protein Data Bank (PDB) (Berman *et al.*, 2000) entries in version v1.75 from June 2009.

In order to illustrate the capabilities of Pleiades, we start with an evaluation that involves a large number of protein structures and a clear biologically relevant goal: the automated detection of protein folds. More specifically, we compared the results of our clustering method with the fold classification in the SCOP database (Murzin *et al.*, 1995).

We extracted and converted all domains present in the current release of SCOP into GIT vectors. For the clustering test, we limited ourselves to domains from the main SCOP classes all α, all β, $\alpha+\beta$ and α/β, since multichain complexes as well as small peptides are beyond the scope of this algorithm. We further removed very small families, with less than five members, and structures with problems in the conversion, for example due to missing atoms. In total, we included 63 864 domains from 1436 families and 823 superfamilies.

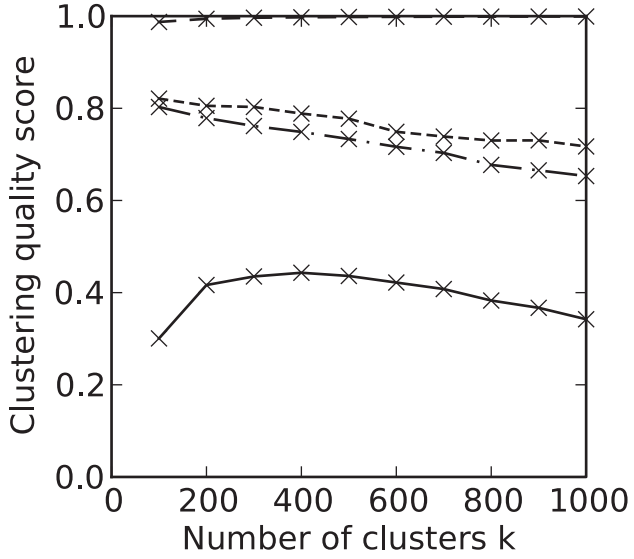

Fig. 3. Rebuilding the SCOP database: The figure shows—from top to bottom—the specificity (medium dashed line), the correct classification rate (small dashed line), the sensitivity (dash-dotted line) and the adjusted Rand score (black line) as a function of the number of clusters. The dataset contained GIT vectors from 1436 families and 823 superfamilies.

In order to investigate the quality of the clustering, we calculated the sensitivity, specificity, correct classification rate (CCR) and the adjusted Rand index. The sensitivity and the specificity are defined as the ratio of the detected true positives and the entire positive set, and the ratio of the detected true negatives and the entire negative set, respectively:

$$\text{sensitivity} = \frac{\text{TP}}{\text{TP}+\text{FN}}$$

$$\text{specificity} = \frac{\text{TN}}{\text{TN}+\text{FP}}$$

where TP is the number of true positives or correctly clustered structures, FN is the number of false negatives or incorrectly clustered structures, TN is the number of true negatives or correctly discriminated structures and FP is the number of false positives or incorrectly discriminated structures. For a perfect clustering, both values are one.

The CCR is the ratio of correctly classified structures over all structures, which has been used to evaluate the results of protein structure clustering before (Chi *et al.*, 2009). We considered a structure as correctly clustered when it was assigned to the same cluster as the majority of the members of the same family.

The adjusted Rand index (ARI) (Hubert and Arabie, 1985) is a measure of the similarity of two clustered sets. Unlike the classic Rand measure (Rand, 1971), the ARI takes the chance of a correct or incorrect random assignment into account, which is especially important in cases of large datasets with numerous clusters.

The sensitivity (Fig. 3, dash-dotted line) decreases as the number of clusters increases, because more structures belonging to the same SCOP family end up in different clusters. The specificity in our tests is very close to one (Fig. 3, medium dashed line), since there is a large number of structures that are correctly discriminated; the number of structures that are incorrectly clustered together becomes almost insignificant. This indicates that the GIT measure is able to discriminate different structures well (high specificity), but sometimes struggles with detecting similar structures (lower sensitivity). This test is challenging because members of the same fold family can have different chain lengths, which is a known difficulty for the GIT measure (Røgen, 2005).

According to the CCR (Fig. 3, small dashed line), the number of correctly assigned structures increases as k decreases. When setting the number of clusters equal to the number of families (1436) in the dataset, the CCR is 67.2%; when setting it to the number of superfamilies in the set (823), 74% of the structures are assigned to the correct cluster (Fig. 3, small dashed line).

In our test, the classic Rand measure followed the specificity and was very close to one (not depicted). We then turned to the ARI (Fig. 3, black line), which indicates that optimal clustering occurs around $k = 400$.

The large number of clusters and the often relatively small cluster sizes, many with 10 or less members, form a difficult task for the algorithm. From $k = 600$ and upwards, there is an excess of clusters available, which results in empty clusters. However, the test shows that clustering such a large number of structures into a large number of clusters is indeed feasible and yields meaningful results.

3.3 Clustering improves decoy selection

In the following test, we show the applicability of Pleiades in the context of protein structure prediction. We also investigate the applicability of K-means clustering in the context of protein structure clustering in general. The clustering task here is somewhat different since all the structures are structural variations of the same amino acid sequence, and thus all have the same length. Additionally, we are only interested in a relatively small number of well-populated clusters. Also, the suitable number of clusters k is usually unknown. Many structure prediction procedures use a Markov chain Monte Carlo approach and therefore generate decoys that are not fully independent. This often leads to clusters that are interconnected, which renders the evaluation more difficult, since there is not necessarily a clearly *correct* or *incorrect* cluster assignment possible for each decoy.

The purpose of clustering decoy structures is to improve the decoy selection for the final refinement (Shortle *et al.*, 1998). Thus, we evaluate the performance of Pleiades by comparing the cluster centroids obtained from Pleiades, and those from the state-of-the art clustering tool Calibur, to the native structure. Additionally, we also investigate whether the K-means clustering procedure in general is applicable in the context of protein structure clustering. Therefore, as a baseline, we implemented an RMSD-based K-means clustering procedure.

All results are given in Å RMSD to the native structure. Due to excessive computational demands, Pleiades RMSD was only applied to decoy sets with 12 500 or less decoys.

We used a decoy set generated by Wu *et al.* (2007) (http://zhanglab.ccmb.med.umich.edu/decoys/), which we will call the iTasser set, that has been used to evaluate protein structure clustering as before (Li and Ng, 2010). The decoy set consists of 56 folding simulations with a varying number of decoy structures generated per simulation.

Table 1 shows the clustering results on each decoy set. We applied Pleiades clustering with the number of clusters k set to 10. All results

Table 1. Clustering results: application of Pleiades, Pleiades with smart seeding and Pleiades RMSD to the iTasser decoy set, for $k=10$. All results are given in Å RMSD to the native structure. Due to excessive computational demands, Pleiades RMSD was only applied to decoy sets with 12500 or less decoys

PDB ID	Pleiades smart seeding	Pleiades RMSD	Pleiades	Calibur
1abv_	12.31	12.28	11.05	11.97
1af7_	4.42		4.42	4.45
1ah9_	2.99		3.33	2.85
1aoy_	4.42		4.17	4.76
1b4bA	5.43	5.60	5.32	5.75
1b72A	2.71	2.73	2.75	3.10
1bm8_	8.15		6.99	7.07
1bq9A	5.73		6.69	7.50
1cewI	3.35		3.54	3.68
1cqkA	1.73		1.73	1.68
1csp_	2.36	2.31	2.36	2.38
1cy5A	1.67		1.55	1.62
1dcjA_	9.93		10.07	9.96
1di2A_	2.41		2.43	2.62
1dtjA_	1.91		1.91	2.12
1egxA	2.56		2.56	2.60
1fadA	3.62	3.52	3.53	3.66
1fo5A	3.84		3.70	3.77
1g1cA	2.60		2.65	2.65
1gjxA	7.59	7.11	7.59	8.18
1gnuA	7.11		8.29	8.37
1gpt_	4.19		4.24	4.64
1gyvA	3.37		3.36	3.44
1hbkA	3.64		3.63	3.48
1itpA	7.65	7.87	7.95	7.87
1jnuA	2.61		2.72	2.68
1kjs_	5.26		5.26	5.89
1kviA	2.05		2.05	2.10
1mkyA3	4.95	5.06	5.17	5.33
1mla_2	2.79	2.81	3.01	2.82
1mn8A	8.08	7.95	8.13	7.08
1n0uA4	4.26	4.29	4.26	4.53
1ne3A	3.96	4.73	5.07	4.07
1no5A	10.18	10.43	10.33	10.70
1npsA	2.23		2.06	2.28
1o2fB_	5.90	5.14	5.72	6.07
1of9A	3.53		3.54	3.61
1ogwA_	2.41		2.52	1.28
1orgA	2.50		2.52	2.66
1pgx_	3.44		3.25	3.01
1r69_	1.93		1.90	1.92
1sfp_	5.03		5.23	5.21
1shfA	1.51		1.47	1.46
1sro_	3.56		3.43	3.54
1ten_	1.71		2.00	1.84
1tfi_	4.96		4.57	5.08
1thx_	2.13		2.32	2.26
1tif_	7.18	7.36	7.18	6.95
1tig_	4.13	3.58	4.13	3.58
1vcc_	8.04		8.28	6.43
256bA	3.24		3.46	2.98
2a0b_	2.23		2.23	2.78
2cr7A	4.45	3.00	2.82	7.48
2f3nA	1.90		1.82	1.94
2pcy_	4.48		4.70	4.57
2reb_2	6.00	5.20	5.96	5.90
Average	4.33		4.34	4.43

are given in Å RMSD to the native structure for the best centroid of the first five clusters reported (Li and Ng, 2010; Zhang and Skolnick, 2004). On average, the Calibur and Pleiades perform equally well, with a difference in the average result of <0.1 Å. The differences between the clustering programs are within 0.5 Å of each other, with few exceptions.

The dependency of the clustering result on the initial seed is often stated as a problem of K-means clustering. Therefore, we repeated the previous exercise, but chose the initial seeds using the K-means++ algorithm. Table 1 shows the results for clustering with $k=10$. The smarter seeding leads to equal or better results in 66% of the cases compared with the regular K-means approach.

The Supplementary Tables ST1 through ST4 show the clustering result for varying values of k. The choice of the number of clusters appears to have little effect; the average difference between $k=5$ and $k=20$ is within 0.1 Å. Note that for the final selection of the decoys, only the five highest ranked cluster centroids were considered, regardless of the value of k.

We implemented a K-means clustering procedure that uses the standard distance measure between protein structures, the RMSD. The exact description of the approach can be found in the Supplementary Material. Since the RMSD evaluation is computationally expensive, we ran this test only for the smaller decoys sets. Table 1 lists the results of these tests which are on par with both Pleiades and Calibur, showing that the K-means approach using the RMSD as exact distance measure provides state-of-the-art clustering results. This finding also suggest that the K-means approach is capable of compensating for potential loss of information due to the approximative GIT description.

3.4 Computational performance

Using GIT vectors instead of the detailed atomic positions in combination with K-means clustering allows Pleiades to be fast, even for large datasets. All the tests for this study were performed on a common desktop PC (2.7 GHz, AMD Dual Core 5200+) with 4 GB of main memory. Pleiades relies on the precalculation of the GIT vectors, which takes some additional time. We argue, however, that in a typical scenario of a folding study, this does not pose a problem, as the GIT vectors can be calculated in parallel with the actual simulation. Secondly, once converted, different clustering experiments—for example, using different values for k—can be quickly conducted.

The calculation of the GIT vectors can easily be parallelized on multicore systems, for example by using tools like GNU Parallel (http://www.gnu.org/software/parallel/). Software for the convenient conversion of PDB files to GIT vectors is included in the program package (see Section 5).

Table 2 lists run-times for a number of randomly selected decoy sets with different sizes and protein lengths from the iTasser set. All run-times were measured in seconds for a single core process. The table shows that Pleiades is at least one order of magnitude faster than Calibur. Even when including the conversion time, Pleiades is still faster in three cases, comparable in one and only significantly slower for the smallest sets.

All times are measured in seconds and all calculations were performed on a single CPU core on an otherwise idle machine.

Table 2. Run-times: this table lists some of the run-times for randomly selected decoy sets with different sizes. All times are measured in seconds and all calculations were performed on a single CPU core on an otherwise idle machine

PDB ID	No. of decoys/length	GIT conversion	Pleiades (k = 10)	Calibur
1abv_	12 500/103	707	11	114
1tig_	12 500/88	592	12	167
1orgA	20 000/118	1499	19	2388
1vcc_	20 000/76	819	20	716
1gpt_	32 000/47	297	31	1954
1cy5A	32 000/92	1190	31	5116

4 CONCLUSION

In this article, we present Pleiades, a novel approach to protein structure clustering using the GIT representation. Pleiades is able to quickly cluster a very large number of protein structures, a task that is highly relevant in the context of protein folding simulations. Using the efficient GIT representation, we were able to successfully cluster up to 1 million decoys resulting from protein folding studies on a common desktop computer in a few hours.

One of the main advantages of Pleiades is its computational speed. The program is able to cluster the output of a regular folding simulation, up to 50 000 structures, in a matter of seconds run-time, while maintaining an accuracy comparable to state-of-the-art programs.

In order to use the convenient GIT representation of the protein fold, some preprocessing is necessary; all structures must first be converted to GIT vectors. Folding studies usually run for extended periods of time, occasionally saving structures to disk. The conversion of a single structure takes fractions of a second and can be performed in parallel with the actual folding simulation.

The benefits of Pleiades are 2-fold: the increased efficiency and reduced memory consumption. Pleiades redefines the limits of how many protein structures can be clustered, routinely dealing with several hundred thousand structures in minutes run-time. At the same time, conventional clustering tasks involving only tens of thousands of structures can be performed in a few seconds, while maintaining an accuracy on par with state-of-the-art clustering methods. Because of these features and the availability of an implementation under an open-source license, Pleiades timely addresses a current bottleneck in structural bioinformatics.

5 AVAILABILITY

Pleiades was developed in C++ as part of the Phaistos package (Borg *et al.*, 2009) and is freely available from sourceforge (http://sourceforge.net/projects/phaistos/) under the GNU General Public License.

ACKNOWLEDGEMENTS

We thank our colleagues at the Bioinformatics Center (University of Copenhagen), Jesper Ferkinghoff-Borg (DTU Elektro, Technical University of Denmark), Thomas Poulsen (Novozymes) and Leonardo De Maria (Novozymes) for valuable comments and suggestions.

Funding: Danish Council for Strategic Research (NABIIT) (Grant 2106-06-0009 to Ti.H and M.B); Danish Council for Independent Research (FNU) (Grant 272-08-0315 to W.B.).

Conflict of Interest: none declared.

REFERENCES

Arthur,D. and Vassilvitskii,S. (2007) k-means++: the advantages of careful seeding. In *Proceedings of the 18th Annual ACM-SIAM Symposium*, Discrete Algorithms (SODA), pp. 1027–1035. New Orleans, LA, USA.

Berenger,F. *et al.* (2011) Entropy-accelerated exact clustering of protein decoys. *Bioinformatics*, **27**, 939–945.

Berman,H.M. *et al.* (2000) The protein data bank. *Nucleic Acids Res.*, **28**, 235–242.

Bishop,C.M. (2006) *Pattern Recognition and Machine Learning*. Springer, New York.

Boomsma,W. *et al.* (2008) A generative, probabilistic model of local protein structure. *Proc Natl Acad Sci USA*, **105**, 8932–8937.

Borg,M. *et al.* (2009) A probabilistic approach to protein structure prediction: PHAISTOS in CASP9. In *LASR*, pp. 65–70. Leeds University Press, Leeds, UK.

Chi,P.-H. *et al.* (2009) Efficient SCOP-fold classification and retrieval using index-based protein substructure alignments. *Bioinformatics*, **25**, 2559–2565.

Harder,T. *et al.* (2010) Beyond rotamers: a generative, probabilistic model of side chains in proteins. *BMC Bioinformatics*, **11**, 306.

Hubert,L. and Arabie,P. (1985) Comparing partitions. *J. Class.*, **2**, 193–218.

Jain,A.K. (2010) Data clustering: 50 years beyond K-means. *Pattern Recognit. Lett.*, **31**, 651–666.

Kabsch,W. (1976) A solution for the best rotation to relate two sets of vectors. *Acta Crystallogr. A*, **32**, 922–923.

Kabsch,W. (1978) A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallogr. A*, **34**, 827–828.

Lindorff-Larsen,K. *et al.* (2005) Protein folding and the organization of the protein topology universe. *Trends Biochem. Sci.*, **30**, 13–19.

Li,H. and Zhou,Y. (2005) SCUD: fast structure clustering of decoys using reference state to remove overall rotation. *J. Comput. Chem.*, **26**, 1189–1192.

Li,S.C. and Ng,Y.K. (2010) Calibur: a tool for clustering large numbers of protein decoys. *BMC Bioinformatics*, **11**, 25.

Lloyd,S.P. (1982) Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, **28**, 129–137.

Müller,C.W. *et al.* (1996) Adenylate kinase motions during catalysis: an energetic counterweight balancing substrate binding. *Structure*, **4**, 147–156.

Murzin,A.G. *et al.* (1995) SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, **247**, 536–540.

Orengo,C.A. *et al.* (1997) CATH: a hierarchic classification of protein domain structures. *Structure*, **5**, 1093–1108.

Rand,W.M. (1971) Objective criteria for the evaluation of clustering methods. *J. Am. Stat. Assoc.*, **66**, 846–850.

Røgen,P. and Bohr,H. (2003) A new family of global protein shape descriptors. *Math. Biosci.*, **182**, 167–181.

Røgen,P. and Fain,B. (2003) Automatic classification of protein structure by using Gauss integrals. *Proc. Natl Acad. Sci. USA*, **100**, 119–124.

Røgen,P. (2005) Evaluating protein structure descriptors and tuning Gauss integral based descriptors. *J. Phys. Condens. Matter*, **17**, 1523–1538.

Shortle,D. *et al.* (1998) Clustering of low-energy conformations near the native structures of small proteins. *Proc. Natl Acad. Sci. USA*, **95**, 11158–11162.

Simons,K.T. *et al.* (1997) Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and bayesian scoring functions. *J. Mol. Biol.*, **268**, 209–225.

Steinley,D. (2006) K-means clustering: a half-century synthesis. *Br. J. Math. Stat. Psychol.*, **59**, 1–34.

Theobald,D.L. and Wuttke,D.S. (2006) THESEUS: maximum likelihood superpositioning and analysis of macromolecular structures. *Bioinformatics*, **22**, 2171–2172.

Uppenberg,J. *et al.* (1994) The sequence, crystal structure determination and refinement of two crystal forms of lipase B from Candida antarctica. *Structure*, **2**, 293–308.

Wu,S. *et al.* (2007) Ab initio modeling of small proteins by iterative TASSER simulations. *BMC Biol.*, **5**, 17.

Zhang,Y. and Skolnick,J. (2004) SPICKER: a clustering approach to identify near-native protein folds. *J. Comput. Chem.*, **25**, 865–871.