Comparing the Software and Algorithms Available for Clustering Molecular Dynamics Trajectories

Honours Literature Review

Robyn McKenzie
mckrob018@uct.ac.za
University of Cape Town
Cape Town, South Africa

ABSTRACT

As molecular dynamics simulations are increasingly used to analyse complex molecules, having a way of extracting the dominant molecule conformations from the trajectories is important. Clustering algorithms provide a way to extract the dominant conformations and analyse the large amounts of trajectory data that are produced. In this review, the clustering algorithms available from standard molecular dynamics analysis and simulation software and from other sources are evaluated. Factors including usability, customisability and quality of produced clusters are relevant when considering the overall usefulness of software in general, and as a possible tool for clustering trajectories of flexible molecules. In particular, clustering algorithms which are resistant to noise and outliers and can produce clusters of varying sizes are ideal. Additionally, it can be important to use algorithms which do not require user-specified parameters, as these can introduce bias.

KEYWORDS

molecular dynamics, clustering, conformation

1 INTRODUCTION

Molecular dynamics (MD) simulations can be used to provide powerful insights into the structure and properties of molecules in different states. It is common for the simulations, or trajectories, to consist of 50 000, 100 000 [16] or more than a million frames [15]. For these large quantities of data to be meaningfully interpreted, the dominant conformations of the molecule within the simulation must be extracted. This is particularly important for highly flexible molecules. A typical approach is to use a clustering algorithm to group the frames of the trajectory into several clusters. Ideally, the clusters are constructed so that there is maximal similarity between frames within the cluster and maximal dissimilarity between frames from different clusters.

Algorithms for clustering MD trajectories are available from a variety of sources, including standard packages for MD simulation, such as AMBER [17], CHARMM [3] and GROMACS [1], standard packages for MD analysis, such as UCSF Chimera [18] and VMD [12], as well as from stand-alone software packages which are released in conjunction with research. Many factors affect the accuracy and usefulness of the software in each of these groups. Different algorithms will have different computational complexities, in terms of time and memory requirements, will produce different results, and will have applications to which they are particularly suited. The software will also have differences in terms of its licensing, usability and ease of installation.

Due to the increasing application of clustering algorithms to MD, it is important to ensure that the algorithms being applied are suitable and produce useful clusters which do not contain unrelated frames and are not missing related frames. As shown by González-Alemán et al., there can be misinformation regarding what algorithm has been implemented within a package [10]. This misinformation can lead to inaccurate and non-reproducible results, as well as incorrect assumptions about the quality of the resulting clusters.

This review will look at the algorithms for clustering MD trajectories that are available today, either as part of standard MD software or from a separate source, and will consider their overall usefulness in clustering trajectories of flexible molecules.

2 CLUSTERING ALGORITHMS

The majority of clustering algorithms can be categorised as being either **distribution-based** – where the data is assumed to have been sampled from some set of probabilistic distributions, **density-based** – where clusters are created from regions of differing data densities – or **distance-based** – where the similarity between two data points is based on how close together they are [2]. This is one of a variety of ways of categorising clustering algorithms.

Distance-based clustering algorithms are widely used as they are easy to implement and only require that a distance function can be defined on the data. In the case of MD, the root-mean-square deviation (RMSD) is generally used to define the distance between frames [21]. Distance-based algorithms can be further sub-categorised as either flat or hierarchical [2]. In hierarchical distance-based models, the resulting clusters make up a tree or dendrogram where the leaves are individual data points and the root is a cluster containing all the data points. The dendrogram can either be built from the bottom, in bottom-up, agglomerative or linkage algorithms, or from the top, in top-down or divisive algorithms [21]. In flat distance-based models, such as k-means and its variants, clusters tend to be seeded randomly and are then iteratively reconstructed, with each generation of clusters improving upon the previous.

Density-based algorithms are also used in the context of MD. The DBSCAN algorithm is a popular density-based algorithm [2], and HDBSCAN [4, 5], which is a version of DBSCAN with a hierarchical component, has seen many different applications, including MD [16]. Additionally, the algorithm APLoD [15], which is based on the density peaks [19] algorithm, has been designed for clustering MD trajectories.

Distribution-based algorithms, also known as probabilistic or generative models, are not common in terms of applications to MD. However, many distribution-based algorithms can be shown to reduce to distance-based algorithms [2]. As distance-based algorithms are easier to understand and implement, it is logical that they would be used over their corresponding distribution-based algorithms.

While some clustering algorithms do fit neatly into the described categories, others do not. One of the most commonly used clustering algorithms for MD is the Quality Threshold (QT) algorithm, and while it is distance-based, it cannot be classified as either flat or hierarchical. There are also more complex clustering techniques which use combinations of algorithms from different categories to achieve improved clustering [16], and therefore cannot be categorised as others can.

Each clustering algorithm will deliver varying results and have different shortcomings and strengths. Different implementations of a single clustering algorithm may also have subtle or significant differences. One important factor is the degree of outlier sensitivity. Some clustering algorithms are also biased towards creating spherical, convex clusters, clusters of similar sizes or very small clusters [21]. An ideal clustering algorithm should be able to find concave clusters and clusters of irregular shapes, different sizes and different diameters where required. It should also ideally have some way of handling outliers and noise in the data.

Another important aspect of each clustering algorithm is whether it requires or allows user-specified parameters and if these parameters have a default option available. Common parameters are the desired cluster count [21], the cut-off value for hierarchical algorithms, and the similarity threshold for QT algorithms [10]. Particularly if the trajectory analysis is exploratory, specifying these parameters could introduce bias and it is preferable for the algorithm to be able to select suitable default parameters without user input [16]. On the other hand, it is also sometimes useful for the user to be able to adjust the clustering degree by changing a parameter [23].

3 CLUSTERING FOR MOLECULAR DYNAMICS

3.1 Main MD Algorithms

3.1.1 Quality Threshold. The Quality Threshold (QT) algorithm, from Heyer et al. [11] in 1999, is one of the most commonly used clustering algorithms for MD. It generally produces clusters which are well-formed as they have a similarity quality guarantee [10]. This means that any two frames which are placed in the same cluster are guaranteed to have a user-specified level of similarity at minimum. It does not require the user to make any assumptions about cluster count, rather, the user specifies the similarity threshold. However, the similarity threshold does affect the number and sizes of the resulting clusters.

The algorithm works by first placing each frame into its own candidate cluster. For each candidate cluster, each of the frames is considered and is added if the RMSD between it and every other frame in the cluster is below the similarity threshold. Once the candidate clusters are fully constructed, the largest is selected and its frames are removed from the remaining clusters. This process is repeated until each frame is in a cluster or is deemed impossible to cluster.

There is an algorithm by Daura et al. [8] which is similar to the QT algorithm, with a key difference that it does not provide the same similarity quality guarantee which makes QT an effective algorithm [10]. The algorithm from Daura et al. only requires that frames added to a cluster are similar to the seed of the cluster, while QT requires similarity to *every* frame in the cluster. On multiple occasions, it has been stated that QT clustering has been implemented, when actually the algorithm from Daura et al. has been implemented [10].

The QT algorithm has been applied to the trajectory of a protein, and the results compared to those from the algorithm by Daura et al. when applied to the same trajectory [10]. It is shown that frames within the clusters produced by the algorithm from Daura et al. often have an RMSD value between them which exceeds the given similarity threshold, while this is not the case for the clusters produces by the QT algorithm. The result of this is that the QT clusters each have visibly correlated frames, while the clusters produced by the algorithm from Daura et al. include unrelated frames our combine unrelated clusters.

3.1.2 Hierarchical Algorithms. Hierarchical algorithms are relatively fast and are easy to implement and understand. As previously outlined, there are two classes of hierarchical algorithms, agglomerative and divisive, with the difference between the two being the direction in which the hierarchical dendrogram is constructed [21]. In agglomerative algorithms, each data point is initially assigned to its own cluster, and a pair of clusters are merged at each step. In divisive algorithms, all of the data points are initially assigned to one large cluster, and a cluster is split at each step. The way in which the clusters are selected for merging or splitting varies between implementations and affects the quality of the resulting clusters.

Agglomerative hierarchical algorithms seem to be more popular than divisive for clustering MD trajectories as they are more commonly implemented [21, 23]. However, an example implementation of a divisive algorithm chooses to split the cluster with the greatest diameter, which it defines as the maximal distance between any pair of points in the cluster [21]. The point that the cluster is split at is then based off of this maximal distance. Agglomerative algorithms are also known as linkage algorithms. Four implementations are single-linkage, complete-linkage, average-linkage and centroid-linkage. In each of these algorithms, the pair of clusters that have the smallest distance between them are merged, but the algorithms differ in how they define the distance between clusters. The distance between a pair of clusters is defined as the minimal intercluster point-to-point distance in single-linkage, the maximal intercluster point-to-point distance in complete-linkage, the average intercluster point-to-point distance in average-linkage and the distance between the cluster centroids in centroid-linkage.

Implementations vary based on parameter requirements as well. Some implementations will require that the user supply a desired cluster count [21] or a cut-off value[23], which is the RMSD distance at which cluster merging or splitting stops. Other algorithms will perform an optimisation to pick a suitable number of clusters [13, 23]. The complete cluster hierarchy can also be output in its entirety and, because this gives a visualisation of the clusters, it can be used to better understand the data. This also means that a cluster count or cut-off value could be selected after the algorithm is complete, rather than having to run the algorithm multiple times with different parameters [21].

A number of hierarchical algorithms were applied to real MD trajectory by Shao et al. [21] in 2007. The simulations used were of a 10-mer polyadenine DNA single strand, the interaction of a drug with DNA and a protein. Two artificial trajectories were also created by sampling independent trajectories of the polyadenine DNA single strand, with the goal of testing if the algorithms could identify five equally sized clusters in the first artificial trajectory and five substantially differently sized sized clusters in the other. Single-linkage, average-linkage and centroid-linkage successfully located the differently sized clusters. However, as single-linkage is particularly sensitive to the position of points near the boundary of a cluster, because they can continually be merged into the cluster even if they are unrelated, single-linkage performed poorly with the real MD data. Average-linkage and centroid-linkage both performed we on the real MD data, and were able to produce clusters of varying size and shape

Some agglomerative algorithms tend to produce very small or singleton clusters around outliers, as the algorithms begin with each frame as its own cluster [21]. Divisive algorithms are similarly susceptible to outliers as they have a direct effect on where the clusters are split, however, they seem to be faster than agglomerative algorithms for low cluster counts. Shao et al. [21] also note the results of hierarchical algorithms depend heavily on the data, and the user-selected cluster count. Their implementation required a user-specified cluster count, and so they ran the algorithms with a variety of counts to find the optimal value.

3.1.3 *HDBSCAN.*

Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [4, 5], is a density-based algorithm, which outputs a hierarchical tree and has the ability to label frames as noise. The clusters which are output are the areas of high density which are separated by areas of low density or noise [16].

The algorithm makes use of the the idea of neighbourhoods in the data, each neighbourhood consists of a core point and the k nearest data points, or neighbours [16]. The core distance for each neighbourhood is the distance from the core point to the furthest point in the neighbourhood. The algorithm works by first creating a network of all the points, where the weight of the edge between each pair of points is the mutual reachability distance between them. The mutual reachability distance of two points is the maximum of three values: the distance between the two points and the core distances of each of the points. Once the network has been constructed, the edges are removed, beginning with those with the greatest weights, until removing any more edges would result in a disjoint partition of the data. At this point, a hierarchical dendrogram is created via the single-linkage algorithm. Points with the lowest edges between them are merged to create clusters until one large cluster is formed. The dendrogram must then be cut to produce the final clusters. To do this, the user-specified minimum cluster size is used with a cluster stability metric to obtain the final clustering

Although it takes two parameters – minimum cluster size and minimum neighbourhood size – for exploratory purposes HDBSCAN can be used so that is effectively non-parametric. The authors of the algorithm recommend setting both parameters to the default value, one, for this purpose [4].

In 2016, Melvin et al. [16] implemented and applied HDBSCAN to a set of proteins and nucleic acids and found it to be effective in serveral ways. For generally stable molecules, it is able to cluster large-scale conformational changes. Although it produced many small clusters for unstable systems, it was still able to capture the intermediate conformations of the molecules. The fact that it can categorise frames as noise means that it is still able to capture conformations in disordered systems, where other algorithms may produce unusable clustering. Melvin et al. recommend it for intrinsically disordered proteins. It was also successful in identifying clusters of varying shape and density. Finally, due to the ability of HDBSCAN to be used non-parametrically, it is ideal for initial investigation of a system, as its behaviour is a good indicator of the overall stability of the system.

3.1.4 *K-Means and iMWK-Means.*

K-means is one of the simplest clustering algorithms available [2]. The algorithm works by first randomly seeding the centroids of k clusters, where k is the is the user-specified desired cluster count, and assigning each data point to the cluster whose centroid it is nearest to. The centroid of each cluster is then recalculated, by computing the average of all the points in the cluster and the data points are reassigned. This process continues until there are no changes when recalculating the centroids [21].

The fact that the centroids are randomly seeded means that the results are not deterministic [21] and even with multiple runs it is not guaranteed that ideal clustering will be produced. The resulting clusters are also obviously dependent on the choice of k, which is not ideal if the structure of the data is not known before applying the clustering. K-means was applied to MD trajectories in 2007 by Shao et al. [21], and although it performed reasonably well, it was unable to produce cluster of different sizes or irregular shapes.

Intelligent Minkowski Weighted K-Means (iMWK-Means) [9], is a variant of k-means which does not require that the user specify the number of clusters. It only has one parameter which is essentially a choice of distance metric [16]. iMWK-Means works by first overestimating the number of clusters and then performing a round of k-means. After this, each frame is rescaled based on its distance from the centroid of the cluster that it is in. The process of performing k-means and rescaling is repeated until the rescaling results in no changes, after which point, k-means is applied for a final time.

The process of rescaling increases the likelihood that the denser areas of the data will be clustered together, while the noisy frames will receive a low weighting and will affect the clustering to a lessor degree [16].

iMWK-Means was applied by Melvin et al. [16] to a set of MD trajectories of nucleic acids and proteins. It is able to identify clusters of varying shape and density and was found to be primarily effective for generally stable systems, where it tends to produce clusters which show subtle changes in the secondary structures of the molecule. Melvin et al. hypothesise that iMWK-Means may prove additionally useful if it is applied to the frames within a single cluster from another algorithm, for example, HDBSCAN, as it would be able to pick out the more subtle details within the cluster.

3.1.5 APLoD. In 2017, Lui et al. [15] presented the clustering algorithm, Adaptive Partitioning by Local Density-Peaks (APLoD), which, when compared to the density peaks [19] algorithm, is highly efficient in terms of time costs and memory usage. The algorithm uses a density-based search and calculates local densities, rather than global densities as is done in the density peaks algorithm. The number of clusters produced by APLoD is dependent on the choice of localisation parameter, k. A small value for k results in more clusters than a large value for k, additionally, the number of clusters in the high density regions of the data is affected by k to a greater degree than the sparse regions. Lui et al. suggest varying the value of k in order to achieve reasonable clustering.

The APLoD algorithm was applied to a series of extremely large MD data sets and demonstrated its ability to cluster large amounts of data quickly [15]. It is able to create clusters of different sizes based on the density of the different areas of the data due to the fact that it adapts to local densities.

3.2 Software Implementations

3.2.1 Standard Packages. In 2007, Shao et al. [21] developed a freely available C library, PTRAJ, of algorithms for clustering MD trajectories. The algorithms included, among others, are k-means, divisive hierarchical clustering, agglomerative hierarchical clustering – with single-linkage, centroid-linkage, average-linkage and complete-linkage variants.

From 2013, PTRAJ and its successor CPPTRAJ [20] were both distributed as part of the freely available simulation package, AMBER [6, 17]. The PTRAJ code was reworked and rewritten in C++ to become CPPTRAJ. Compared to PTRAJ, CPPTRAJ has substantially improved computation times and a more maintainable codebase [20]. At the time, CPPTRAJ contained three of the algorithms from PTRAJ – the single, average and complete linkage agglomerative hierarchical algorithms – with plans for the remaining algorithms to be added.

CHARMM (Chemistry at HARvard Molecular Mechanics) [3] can be used to cluster trajectories. It contains a k-means implementation. It is also possible to use scripts to run other clustering algorithms with CHARMM [3, 14]. A version without high-performance functionality, charmm, is available for free. Otherwise, the full version of CHARMM can be licensed for a fee [7].

GROMACS [1] is freely available program for MD. A major goal of the software is to be highly efficient on computers with widely varying specifications and to be easily extendable by outside developers. The most commonly used algorithm available within GROMACS is the *gromos* method [10] which is an implementation of the algorithm by Daura et al. [8].

UCSF Chimera [18] has been designed to be extensible. The software is freely available, and is primarily written in Python, with a C++ core layer. Its structure makes it simple for outside developers and researchers to add additional functionality. It uses the NMRCLUST [13] algorithm to cluster. This algorithm is an average-linkage hierarchical algorithm which does not require the user to specify a cut-off value or desired cluster count. It instead uses a penalty function to choose a cut-off value which minimises the number of clusters as well as the spread of each cluster. This maximises the likelihood that the frames within a cluster are similar and that there are no frames that should be included in the cluster that are missing. However, as it is an agglomerative hierarchical algorithm it will sometimes produce many small clusters [23].

The VMD (Visual Molecular Dynamics) [12] *measure cluster* command can also be used to cluster MD trajectories. The clustering algorithm is described as being based on the QT algorithm [10]. However, in 2019 González-Alemán et al. showed that the clusters produced by *measure cluster* do not posses the similarity quality guarantee which makes QT a popular algorithm. Instead the resulting clusters provably contain pairs of frames which are more dissimilar than should be allowed by the similarity threshold. Based on this, it seems that the *measure cluster* command instead implements the algorithm by Daura et al. [8]. VMD is also distributed freely with its C++ source code, and has plugin system which makes it highly extensible [22].

In 2018, Tubiana et al. [23] presented TTClust, an easy-to-use and freely available clustering program for MD. It contains an agglomerative hierarchical algorithm with a number of variants including single-linkage, complete-linkage, average-linkage and centroid-linkage. The user can supply a cut-off value or desired cluster count, alternatively the TTClust implementation can compute an optimal cluster count. Because the TTClust algorithm saves the RMSD between each pair of frames during its initial run, these values do not need to be recalculated on subsequent runs unless parameters are changed. This is useful if repeated runs with different parameters are required.

3.2.2 Stand-alone Packages. González-Alemán et al. [10] provided a freely available Python implementation of QT clustering in 2019. The clusters produced with this algorithm were shown to have a similarity quality guarantee, unlike the implementation of the algorithm by Daura et al. [8] in the GROMACS [1] *gromos* and VMD [12] *measure cluster* commands. González-Alemán et al. additionally point out that, as of 2019, a freely available and correct implementation of QT clustering did not exist.

A Python implementation of HDBSCAN and iMWK-Means for MD is available from Melvin et al.[16]. The package was made available in 2016 and makes use of the implementations of each the algorithms which were made available by their respective authors.

The authors of the APLoD [15] algorithm have made their Python implementation available for free. The implementation is from 2017.

4 DISCUSSION

4.1 Algorithms

Table 1 shows a summary of the algorithms investigated in this review. Each has different strengths and weaknesses which are important in considering them as useful candidates for clustering MD trajectories of flexible molecules. As there will never be a single algorithm which always delivers superior results, a variety of approaches should be selected as candidates [21].

It is reasonable to assume that clustering methods which are able to produce clusters of varying shapes and sizes would be useful in clustering flexible molecules. HDBSCAN, iMWK-Means and the

Table 1: Comparison of Clustering Algorithms

Algorithm	Implemented In	Parameters	Notes
Quality Threshold [11]	Stand alone software [10]	User-specified similarity threshold	
Algorithm from Daura et al. [8]	GROMACS [1] *gromos*	Unknown	
	VMD [12] *measure cluster*	Unknown	
Hierarchical – Divisive	PTRAJ [21], CPPTRAJ [20] (AMBER)	User-specified cluster count	
Hierarchical – Agglomerative	PTRAJ [21], CPPTRAJ [20] (AMBER)	User-specified cluster count	
	TTClust [23]	Optional user-specified cluster count or cut-off value	Optimal cluster count is computed if no parameters are given
Hierarchical – Agglomerative (NMRCLUST) [13]	UCSF Chimera [18]	None	Penalty function used to choose optimal cluster count
HDBSCAN [4, 5]	Stand alone software [16]	Minimum cluster size, minimum neighbourhood size	Suitable default values available – can be used non-parametrically
iMWK-Means [9]	Stand alone software [16]	Choice of distance metric	Suitable default values available – can be used non-parametrically
K-Means	CHARMM [16]	User-specified cluster count	
	PTRAJ [21], CPPTRAJ [20] (AMBER)	User-specified cluster count	
APLoD [15]	Stand alone software [15]	User-specified localisation parameter	

agglomerative hierarchical average-linkage and centroid-linkage variants are known to be capable of this, although others may also be. These algorithms will still perform well even if some of the dominant conformations of the molecule are visited less frequently in the trajectory than others, and this is likely to be the case for at less some flexible molecules. Additionally, HDBSCAN and iMWK-Means have the ability to ignore noise in the data [16] while average-linkage and centroid-linkage may be sensitive to outliers. HDBSCAN, in particular, was able to find dominant conformations even in disordered systems with high levels of noise. This may be useful in clustering flexible molecules as many of the frames may not depict any of the dominant structures visited by the molecule.

As the structure of the trajectories is unlikely to be known in advance, algorithms that are non-parametric, or have suitable default values for parameters, would be ideal as well. Having to specify a cluster count requires assumptions to be made and can introduce a degree of bias [23]. HDBSCAN and iMWK-Means are both suitable in this case as they can easily be used with default parameters which effectively make them non-parametric. Additionally, neither the NMRCLUST implementation of agglomerative hierarchical clustering nor the TTClust implementation requires parameters from the user.

The QT algorithm is also an attractive option for clustering, as the resulting clusters will always have a similarity quality guarantee and hence will be highly correlated. Although it does require a parameter, the similarity threshold, this requires less of a direct assumption about the data than a cluster count, for example. QT clustering is also superior to the algorithm from Daura et al. as this algorithm does not possess the same quality guarantee. The fact that no unrelated frames can be added to a cluster also means that the degree to which outliers can affect the final clustering is limited.

Although APLoD seems like a promising algorithm, its use case is primarily focused on creating Markov State Models for proteins[15]. Because of this, it may not be an ideal candidate for clustering general flexible molecules where the goal is to study the dominant conformations of the molecule.

4.2 Software Usability

Algorithms should be selected so as to prioritise both the quality of the cluster that are produced, as well as the ease of installation, customisation and use of the software in which the algorithm is implemented. The software which are implemented completely or partially in Python are possibly more useful than the software which are not. Python is simpler and easier to customise than C, C++ and Fortran. The powerful tools available in the Python ecosystem can also be leveraged more easily through software that is already implemented in Python.

PTRAJ and its successor CPPTRAJ are written in C and C++ respectively. It is unclear whether or not the PTRAJ library is still distributed with AMBER or available at all. It was implemented in 2007 [21], and plans to replace it with CPPTRAJ were in motion by 2013 [20]. Regardless, the way that the PTRAJ code-base was structured meant that it was not easily extendable and customising it or using it as it is would be complicated. CPPTRAJ is more easily extendable that PTRAJ, and might be expected to be straightforward to use as it is distributed with AMBER [20]. However, it is complicated to use for those not familiar with AMBER [23]. If all of the algorithms from PTRAJ have now been implemented in CPPTRAJ and are hence available in AMBER, CPPTRAJ may be an attractive option for clustering flexible molecules due to the variety of algorithms it includes.

GROMACS, UCSF Chimera and VMD all put an emphasis on the fact that their software is intended to be extended by outside researchers, and they have endeavoured to structure their source code so that the customisation process is straightforward [1, 12, 18]. This same emphasis is seemingly absent from CHARMM and AMBER. UCSF Chimera particularly lends itself to customisation, as its structure is divided into a C++ and Python core which implements key functionality, while more specific functionality is all added through extensions which can either be entirely in Python or a mixture with C++ [18].

The algorithms implemented by González-Alemán et al. [10], Melvin et al. [16] and Liu et al. [15] are not part of standard MD software and therefore there may be some difficulty in using them as they may not have much, if any, documentation. Similar consideration must be made for TTClust [23]. Although it is a complete system for clustering MD trajectories, it is not as widely used as the other software discussed and there may be unforeseen issues. However, these concerns are somewhat offset by the fact that these are all Python libraries, which adds to their usability.

5 CONCLUSIONS

In this review, we have evaluated the algorithms available for clustering MD trajectories. These algorithms are available from standard MD simulation and analysis software, as well as from separate packages made available with research. Standard MD software may have better documentation and be more stable. More caution must be exercised when using non-standard software, but it should not be completely ignored as it may provide algorithms which produce better clustering.

Trajectories of flexible molecules are likely to be successfully clustered by algorithms that can handle noise and outliers, can produce clusters of varying shapes and sizes and do not require too many assumptions about the data to be made by the user. It is also advisable that a variety of algorithms be applied to any one flexible molecule, to ensure that odds of achieving useful clustering are maximised.

HDBSCAN, iMWK-Means, NMRCLUST from UCSF Chimera are ideal in that they require no parameters. UCSF Chimera has the added benefit of being suited to customisation if this is required and NMRCLUST implements average-linkage hierarchical clustering which is able to produce clusters of varying shapes and sizes, as HDBSCAN and iMWK-Means are.

HBDSCAN and iMWK-Means also seem to be the least vulnerable to noise and outliers out of all the algorithms investigated. HBDSCAN in particular is able pick the dominant conformations out of disordered systems where a large number of the frames are categorised as noise.

Finally, QT clustering is a popular approach for clustering MD trajectories and González-Alemán et al. [10] provide a true implementation of it. QT produces clusters with a quality guarantee which ensures that no unrelated frames, or outliers, are included within the resulting clusters.

REFERENCES

[1] Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Páll, Jeremy C Smith, Berk Hess, and Erik Lindahl. 2015. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX* 1-2, C (2015), 19–25.

[2] Charu C. Aggarwal and Chandan K. Reddy. 2013. *DATA CLUSTERING Algorithms and Applications.* CRC Press/Taylor and Francis Group, University of Minnesota, Minneapolis, Minnesota.

[3] B. R. Brooks, C. L. Brooks, A. D. Mackerell Jr., L. Nilsson, R. J. Petrella, B. Roux, Y. Won, G. Archontis, C. Bartels, S. Boresch, A. Caflisch, L. Caves, Q. Cui, A. R. Dinner, M. Feig, S. Fischer, J. Gao, M. Hodoscek, W. Im, K. Kuczera, T. Lazaridis, J. Ma, V. Ovchinnikov, E. Paci, R. W. Pastor, C. B. Post, J. Z. Pu, M. Schaefer, B. Tidor, R. M. Venable, H. L. Woodcock, X. Wu, W. Yang, D. M. York, and M. Karplus. 2009. CHARMM: The biomolecular simulation program. *Journal of Computational Chemistry* 30, 10 (2009), 1545–1614. https://doi.org/10.1002/jcc.21287

[4] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. 2013. Density-Based Clustering Based on Hierarchical Density Estimates. In *Advances in Knowledge Discovery and Data Mining*, Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 160–172.

[5] Ricardo J. G. B Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander. 2015. Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 10, 1 (2015), 1–51.

[6] D.A. Case, I.Y. Ben-Shalom, S.R. Brozell, D.S. Cerutti, III T.E. Cheatham, V.W.D. Cruzeiro, T.A. Darden, R.E. Duke, D. Ghoreishi, M.K. Gilson, H. Gohlke, A.W. Goetz, D. Greene, R Harris, N. Homeyer, S. Izadi, A. Kovalenko, T. Kurtzman, T.S. Lee, S. LeGrand, P. Li, C. Lin, J. Liu, T. Luchko, R. Luo, D.J. Mermelstein, K.M. Merz, Y. Miao, G. Monard, C. Nguyen, H. Nguyen, I. Omelyan, A. Onufriev, F. Pan, R. Qi, D.R. Roe, A. Roitberg, C. Sagui, S. Schott-Verdugo, J. Shen, C.L. Simmerling, J. Smith, R. Salomon-Ferrer, J. Swails, R.C. Walker, J. Wang, H. Wei, R.M. Wolf, X. Wu, L. Xiao, D.M. York, and P.A. Kollman. 2018. AMBER 2018.

[7] CHARMM. 2020. CHARMM. https://www.charmm.org// Retrieved 12 May 2020.

[8] Xavier Daura, Karl Gademann, Bernhard Jaun, Dieter Seebach, Wilfred F. Van Gunsteren, and Alan E. Mark. 1999. Peptide Folding: When Simulation Meets Experiment. *Angewandte Chemie International Edition* 38, 1-2 (1999), 236–240.

[9] Renato Cordeiro de Amorim and Christian Hennig. 2015. Recovering the number of clusters in data sets with noise features using feature rescaling factors. *Information Sciences* 324 (2015), 126–145. https://doi.org/10.1016/j.ins.2015.06.039

[10] Roy González-Alemán, David Hernández-Castillo, Julio Caballero, and Luis A. Montero-Cabrera. 2019. Quality Threshold Clustering of Molecular Dynamics: A Word of Caution. *Journal of Chemical Information and Modeling* 60, 2 (2019), 467–472. https://doi.org/10.1021/acs.jcim.9b00558

[11] L J Heyer, S Kruglyak, and S Yooseph. 1999. Exploring Expression Data: Identification and Analysis of Coexpressed Genes. *Genome research* 9, 11 (1999), 1106–1115.

[12] William Humphrey, Andrew Dalke, and Klaus Schulten. 1996. VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics* 14 (1996), 33–38.

[13] Lawrence A. Kelley, Stephen P. Gardner, and Michael J. Sutcliffe. 1996. An automated approach for clustering an ensemble of NMR-derived protein structures into conformationally related subfamilies. *"Protein Engineering, Design and Selection"* 9, 11 (1996), 1063–1065.

[14] Sergei V Krivov and Martin Karplus. 2004. Hidden complexity of free energy surfaces for peptide (protein) folding. *Proceedings of the National Academy of Sciences of the United States of America* 101, 41 (2004), 14766–14770.

[15] Song Liu, Lizhe Zhu, Fu Kit Sheong, Wei Wang, and Xuhui Huang. 2017. Adaptive partitioning by local density-peaks: An efficient density-based clustering algorithm for analyzing molecular dynamics trajectories. *Journal of Computational Chemistry* 38, 3 (2017), 152–160. https://doi.org/10.1002/jcc.24664

[16] Ryan L. Melvin, Ryan C. Godwin, Jiajie Xiao, William G. Thompson, Kenneth S. Berenhaut, and Freddie R. Salsbury. 2016. Uncovering Large-Scale Conformational Change in Molecular Dynamics without Prior Knowledge. *Journal of Chemical Theory and Computation* 12, 12 (2016), 6130–6146. https://doi.org/10.1021/acs.jctc.6b00757

[17] David A Pearlman, David A Case, James W Caldwell, Wilson S Ross, Thomas E Cheatham, Steve Debolt, David Ferguson, George Seibel, and Peter Kollman. 1995. AMBER, a package of computer programs for applying molecular mechanics, normal mode analysis, molecular dynamics and free energy calculations to simulate the structural and energetic properties of molecules. *Computer Physics Communications* 91, 1-3 (1995), 1–41.

[18] Eric F Pettersen, Thomas D Goddard, Conrad C Huang, Gregory S Couch, Daniel M Greenblatt, Elaine C Meng, and Thomas E Ferrin. 2004. UCSF Chimera–a visualization system for exploratory research and analysis. *Journal of computational chemistry* 25, 13 (2004), 1605–1612.

[19] Alex Rodriguez and Alessandro Laio. 2014. Clustering by fast search and find of density peaks. *Science* 344, 6191 (2014), 1492–1496.

[20] Daniel R Roe and Thomas E Cheatham. 2013. PTRAJ and CPPTRAJ: Software for Processing and Analysis of Molecular Dynamics Trajectory Data. *Journal of chemical theory and computation* 9, 7 (2013), 3084–3095.

[21] Jianyin Shao, Stephen W. Tanner, Nephi Thompson, and Thomas E. Cheatham. 2007. Clustering Molecular Dynamics Trajectories: 1. Characterizing the Performance of Different Clustering Algorithms. *Journal of Chemical Theory and Computation* 3, 6 (2007), 2312–2334. https://doi.org/10.1021/ct700119m

[22] Theoretical and Computational Biophysics Group. 2020. VMD – Visual Molecular Dynamics. https://www.ks.uiuc.edu/Research/vmd/ Retrieved 12 May 2020.

[23] Thibault Tubiana, Jean-Charles Carvaillo, Yves Boulard, and Stéphane Bressanelli. 2018. TTClust: A Versatile Molecular Simulation Trajectory Clustering Program with Graphical Summaries. *Journal of Chemical Information and Modeling* 58, 11 (2018), 2178–2182. https://doi.org/10.1021/acs.jcim.8b00512 PMID: 30351057.