ON THE POWER AND LIMITS OF SEQUENCE SIMILARITY BASED CLUSTERING OF PROTEINS INTO FAMILIES

CHRISTIAN WIWIE and RICHARD RÖTTGER

Department of Mathematics and Computer Science, University of Southern Denmark, Odense, Fyn, Denmark E-mail: {wiwiec, roettger}@imada.sdu.dk

Over the last decades, we have observed an ongoing tremendous growth of available sequencing data fueled by the advancements in wet-lab technology. The sequencing information is only the beginning of the actual understanding of how organisms survive and prosper. It is, for instance, equally important to also unravel the proteomic repertoire of an organism. A classical computational approach for detecting protein families is a sequence-based similarity calculation coupled with a subsequent cluster analysis. In this work we have intensively analyzed various clustering tools on a large scale. We used the data to investigate the behavior of the tools' parameters underlining the diversity of the protein families. Furthermore, we trained regression models for predicting the expected performance of a clustering tool for an unknown data set and aimed to also suggest optimal parameters in an automated fashion. Our analysis demonstrates the benefits and limitations of the clustering of proteins with low sequence similarity indicating that each protein family requires its own distinct set of tools and parameters. All results, a tool prediction service, and additional supporting material is also available online under http://proteinclustering.compbio.sdu.dk.

Keywords: Protein Classification, Protein Evolution, Clustering

1. Introduction

With current wet-lab technology, we are producing a vast amount of genomic data at an ever increasing pace.¹ The knowledge of the very sequence of the organism is only one part of the complex puzzle of how organisms survive, reproduce and adopt to changing environmental conditions.² In order to benefit from the genomic data of an organism the data needs to be analyzed in an efficient and automated manner.

Of fundamental importance is the identification and classification of protein families fostering insights in the functional diversity of homologous proteins allowing to investigate the evolutionary history of the proteins.^{3,4} Several, hand-curated databases exist providing information on protein family classification, e.g., SCOP⁵ or PFAM.⁶ Even though these databases are impressive in size, the number of known protein families is still growing with every sequenced organism.⁷ Therefore, it is of importance to have reliable and automated means of classifying proteins in families, which can generally be separated into three groups:^{8,9} pairwise alignment algorithms, generative models, and discriminative classifiers. Here, we are focusing on the common approach of pairwise alignments using NCBI BLAST¹⁰ followed by a cluster analysis. There exists a myriad of clustering tools, all of them require different parameters and can only be used efficiently with a profound understanding of the underlying algorithm. Furthermore, as every clustering approach uses a different way of determining its optimal clustering, there is no universal best performer suiting all data sets equally well.¹¹

There have been several studies comparing the performance of various clustering approaches for this task, discussing the problem from various points of view. For example, the study of Chan *et al.*¹² compares the performance of two clustering tools on three different genomes in order to assess the sensitivity of these tools towards the C+G content. The main limitation of this study is the small number of data sets and tools utilized. In a different study by Bernardes *et al.*³ a larger-scale attempt was taken to compare the general performance of four different clustering approaches on data sets similar to our setting. The main focus of the paper was to demonstrate the limitations of sequence-based similarity functions compared to their novel profile based similarity function. Nevertheless, this work applied the tools in question only to the entire SCOP data set (with various levels of sequence identities) and clustered them into families and superfamilies. This approach neglects the variety within the protein families but gives a good overview of the general performance of the tested tools.

In contrast to previous works, we create several hundred data sets comprising smaller subsets of the SCOP data set in order to strategically assess the variance of the different protein families and their consequences to the different clustering tools. Further, we clustered each of our hundreds of data sets with extensive parameter training (1,000 parameters per data set per tool) using seven popular clustering approaches which have already demonstrated to work well on protein data sets.¹¹ This approach allows for a more detailed evaluation of the performances and limitations of the clustering tools. We further use the massive database of 100 of thousands clustering results generated during this work in order to conduct a meta learning approach, comparable to the work of De Souto *et al.*,¹³ for the prediction of the expected clustering performance and thus a tool ranking. We also suggest the parameter settings for the tools, as we can identify the most similar data set in our database together with the best parameters.

To summarize, we present an in-depth analysis of protein clustering and the inherent variability of the data sets. We intensively investigated the performance of the tools on 202 different data sets with 1,000 different parameter settings each. We investigated the behavior of the tools and their parameters, reflecting the diversity of the different protein families. With a meta-learning approach we aim to predict the expected performance of the clustering tools on unseen data sets. We utilized intrinsic properties of the data sets (e.g., matrix rank or the cluster coefficient) and used them as features of a regression model for the prediction. We also provide the performance predictor as a web-service together with all results, the source code of the predictor, and additional information at http://proteinclustering.compbio.sdu.dk.

2. Materials

2.1. Data sets

We based our work on the Astral SCOPe 2.06 data set with less than 40% sequence identity.⁵ This scenario is very challenging for clustering tools as the alignment scores fall into the so-called twilight zone when the sequence identity drops below 35%.¹⁴ The data set provides a gold standard classification derived from the SCOP database which we utilize in order to assess the cluster quality. The Astral data set classifies each protein into a hierarchy of *class*, *fold*, *superfamily* and eventually *family*.

For our goal of predicting the expected performance of the clustering tools we require a multitude of data sets. Therefore, we have created sub-samples of the Astral data set by splitting it into classes and folds, i.e., we have created a single data set for each class, containing only the sequences of the one class and one data set for each fold in the same fashion. In the remainder we will refer to them as the *class data sets* and the *fold data sets*. This serves two purposes: (1) we received a sufficient number of data sets and (2) we were able to assess the diversity of the protein families and their impact on the clustering tools. We calculated pairwise BLAST¹⁰ hits (E-value cut-off 100) between all protein sequences and converted them into similarities using the "Coverage BeH" method by Wittkop *et al.*¹⁵ (coverage factor f = 20, cut-off 100,000).

Given these data sets, we cluster each of them into the corresponding families, leading to the following two *scenarios*: $Class \rightarrow Families$ and $Fold \rightarrow Families$. We performed a final filtering process by excluding all those data sets containing only one cluster, e.g., a fold containing only one family. We excluded them because they are trivial to cluster and would hugely distort the parameter prediction. After this final step we created seven class data sets and 195 fold data sets.

2.2. Clustering Tools

Table 1. Overview of the chosen clustering methods. We assign an abbreviation to each of the tools. We optimized the denoted parameters for each of the tools.

Abbreviation	Name	Optimized Parameter(s)
CDP	$Clusterdp^{16}$	Kernel radius $dc \in [\land, \lor]$
HC(linkage)	Hierarchical Clustering ¹⁷	Number of clusters $k \in [2, n]$
MCL	Markov Clustering ¹⁸	Inflation $I \in [1.1, 10]$
PAM	Partitioning Around Medoids ¹⁹	Number of clusters $k \in [2, n-1]$
TC	Transitivity Clustering ²⁰	$T \in [\wedge, \vee]$

We based our tool selection on the top performers (using the F1-score²¹) of a previous largescale performance comparison of various clustering approaches,¹¹ summarized in Table 1. The F1-score is defined as the harmonic mean of precision and recall when comparing a cluster result with a gold standard. Generally, external validity indices (i.e., measures comparing against a gold standard) evaluate a result with regard to the purity of individual clusters and the completeness of the clusters.^{11,21} In that context, the F1-score is a comprehensive measure that takes both of these into account by combining two external measures (precision and recall). The F1-score is the quasi-standard in clustering evaluation and has already proved useful in many biomedical contexts.^{11,21} All considered clustering tools performed very well with an average F1-score of over 0.7 in the original study. We excluded tools which return overlapping clusters, as the F1-Score is undefined for such clusterings. We treat hierarchical clustering as three tools, depending on the linkage function used (single, complete, average).

3. Methods

3.1. Data Statistics & Clustering

For each data set, we calculated 25 data statistics (see Table 2). We selected these statistics to reflect a wide variety of properties of the data sets. Note, that some of the statistics are

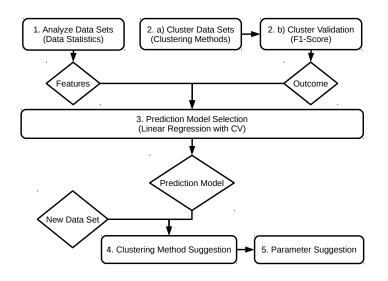


Fig. 1. Overview of the workflow of the presented method. (1) We calculate the features for the models, (2a) perform a clustering of all data sets and (2b) evaluate their quality. (1) and (2) are used to (3) train a regression model. (4) This model is used to predict the expected performance of each tool and suggests (5) the parameters.

correlated; this fact and the influence on the models is discussed in Section 3.2. The ranges of all statistics except *Minimal Similarity*, *Maximal Similarity* and *Number Samples* were normalized to [0, 1] to avoid biases in the trained regression models due to differences in the value ranges.

We utilized ClustEval¹¹ to execute each clustering tool with 1,000 different parameter sets as indicated in Table 1 and validated the results using the F1-Score. The maximal execution time of any tool per clustering was limited to 15 minutes as we occasionally observed degenerated execution times depending on the used parameters.

3.2. Regression & Feature Selection

For each clustering tool we selected an ordinary, Lasso and Ridge regression model. We used the R functions lm, glmnet ($\alpha = 1$) and glmnet ($\alpha = 0$) to train ordinary, Lasso and Ridge regression models respectively. The data set statistics used as features for the regression models are potentially correlated and thus might be troublesome for regression models. For this reason, we perform a feature selection for the ordinary linear regression. Lasso and Ridge regression already have an intrinsic feature selection, thus they were not subject to an additional feature selection.

We trained each of the three regression models per tool using the data statistics as feature variables. The outcome variables are either the best achieved F1-Score of each tool on each data set, or the parameter leading to the best result; depending on whether we want to predict the F1-Scores or the parameters. To assess the quality of the prediction, we used the mean absolute error (MAE) to measure error rates: $MAE(\hat{y}, y) = \frac{1}{N} \sum_{i=1}^{N} |\hat{y}_i - y_i|$ where \hat{y}_i denotes the prediction, y_i the real value for data set i, and N the total number of data sets. Using MAE

Table 2. Overview of the calculated data statistics. The *Absolute Z-Score*, *Assortativity* and *Similarity Percentiles* are parameterized, i.e., we calculate the same statistic multiple times for different parameters. The brackets behind the statistic name denotes the number of parameters used.

Description
The fraction of all object pairs having a similarity within $\{1,2,3,4\}$ stan-
dard deviations from the mean.
The preference for vertices with same degree to connect to each other in
the similarity graph.
The ratio of fully connected triplets of nodes to connected triplets of
nodes in the similarity graph.
The number of edges to remove such that the similarity graph falls into
several connected components.
The ratio of the number of edges and the number of possible edges in
the similarity graph.
The average scaled Shannon entropy of the weights of the incident edges
on each vertex in the similarity graph.
The sum of edge weights to remove such that the similarity graph falls
into several connected components.
The number of independent rows in the similarity matrix.
The largest similarity in the similarity matrix.
The smallest similarity in the similarity matrix.
The number of objects in the input data set.
The fraction of all object pairs having a similarity within the $\{[0-10], [10-10], $
$20], \ldots, [90-100]$ similarity percentile.

allows for easy interpretation of the error-rate compared to other measures such as the root mean squared error (RMSE).²⁷

3.2.1. Cross Validation

In order to estimate prediction errors for a trained model we utilize a 10-fold cross validation. We repeated the cross validations 100 times with different folds to minimize the influence of a single fold. Note that the Astral data set has only seven classes, thus when only using the class data sets, a Leave-one-out cross validation (LOOCV) was performed instead.

3.2.2. Feature Selection for Ordinary Regression Models

We utilized a greedy forward feature selection approach coupled with 10-fold cross validations to select features and thus models with small prediction error while trying to avoid overfitting. In each step of the process, we successively added that feature to the model which lead to the smallest cross validation prediction error estimate.

During this feature selection procedure, we generate models of increasing complexity, i.e., using more features. Thus, both training and testing errors of the cross-validation will decrease in the beginning. However, with increasing number of features, the model will overfit the training data which is indicated by a growing prediction error. The moment we observe a growing prediction error, we stop adding features and report the current model as the final model. A similar feature selection procedure was previously published in Pahikkala *et al.*²⁸

4. Results & Discussion

4.1. Data Statistics

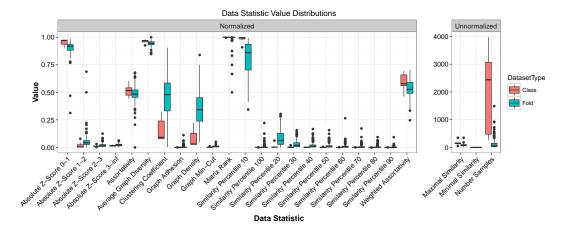


Fig. 2. The distributions of data statistic values for the class and fold data sets. We normalized data statistics using the theoretical maxima where available. The statistics *Minimal Similarity*, *Maximal Similarity* and *Number Samples* are not normalized.

Figure 2 summarizes the calculated data statistics for both class and fold data sets. Generally, some statistics such as *Graph Min-Cut*, *Graph Density* or *Clustering Coefficient* emphasize the sparsity of the pair-wise similarity matrix of the protein sequences. This is due to the fact that the proteins in the Astral data set do not have large sequence similarities resulting in many protein pairs without any significant BLAST hit. Further, we want to highlight two interesting observations:

(1) There is a clear difference in the statistical properties between the class and fold data sets. Again, this is due to the many protein pairs without any BLAST hit. The ratio of these pairs is larger in the class data sets which contain even more distantly related proteins. This is most clearly seen on Statistics such as Average Graph Diversity, Clustering Coefficient, Graph Density, Similarity Percentile 10/20 and Absolute Z-Score 0-1/1-2 which are very sensitive to this proportion.

(2) Even data sets of the same type (i.e., fold or class) vary hugely. This demonstrates the variety of the different protein families. This is even more pronounced in the fold data sets as they contain fewer families and thus are more susceptible to "outlier" families whereas in the class data sets, the variety of the different statistics is generally more balanced.

4.2. Clustering Tool Performances

We clustered all data sets into protein families using the clustering tools summarized in Table 1 to all previously mentioned class and fold data sets. The resulting F1-Scores are depicted in Figure 3. Generally, the selected clustering methods perform well on the data sets. HC(complete), MCL and PAM perform on average slightly worse than their competitors. The

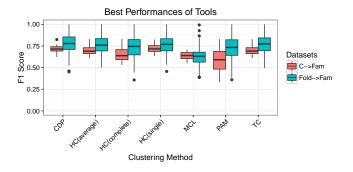


Fig. 3. The best tool performances as F1-Scores for the two scenarios: Clustering class (C) or fold (Fold) into families (Fam).

performance of PAM on class data sets might be due to our execution time limit of 15 minutes per clustering. For k-parameter values close to the real number of clusters in the classes, the algorithm does not finish in time. On the other hand, we only have seven of those data sets in this study, so the effect on the performance should be limited. None of the other methods were affected by the time limit. The general trend is that fold data sets can be clustered better (on average) than class data sets which can be explained by the fact that the class data sets are sparser. When ranking the tools by their F1-Score performance for each data set it shows that there is no best performer across all data sets, as expected. Rather, several tools alternate in taking the top ranks. The lack of a universal best performer and the variance in the rankings emphasize that performances and rankings are highly data set dependent. This further motivates the demand of a predictor based on data statistics.

4.3. Clustering Tool Parameters

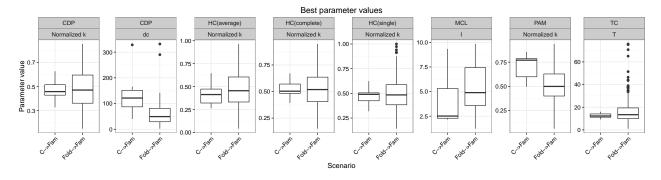


Fig. 4. The best performing parameter values of all tools. If a tool performed best with multiple parameter values we took the mean. Each tool was executed using 1,000 different parameter values. Because CDP has two main parameters we assessed both of them with an equal number of values: 32 * 32 = 1024. Note that the parameter k is normalized by the number of objects.

We compared the best parameters of each tool for the two scenarios. Figure 4 summarizes our findings. Clearly, when clustering a fold data set we can observe a considerably larger variety for all tools. Parameters directly reflecting the desired number of clusters, i.e. k, have been normalized with the number of objects in the data set. Please note, that we cannot use the mean k parameter as a general "rule-of-thumb" as this value entirely depends on the average family size in the data set which is determined by the way we created the data sets. Nevertheless, the variance in the k parameters certainly demonstrate the variance in protein families. The only outlier with respect to the k parameter is PAM, again likely due to the runtime restriction.

Interestingly, the parameters of CDP and MCL have different means when clustering classes compared to clustering folds. This has practical implications, as for an unknown data set it is impossible to determine whether it is comprised of a class, a fold or a mixture. The threshold T of TC remains stable regardless of the data set type, with a larger variance for the fold data sets, including some significant outliers. Overall, this indicates that a naive parameter suggestion for arbitrary protein data sets is not feasible at least it does not do justice to the variety present in different protein families.

4.4. Predicting Tool Performance

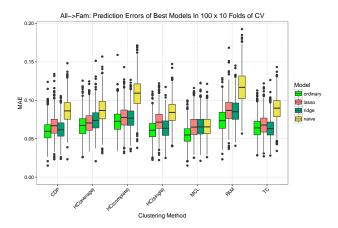


Fig. 5. The tool performance prediction errors for the final models of each tool when trained on all data sets. The prediction errors were estimated with 100×10 -fold cross validations. The yellow boxes represent the performance of the naive model.

Figure 5 compares the tool performance prediction errors of the final models for all tools when trained on all data sets. We also calculate a naive predictor serving as baseline which predicts the average performance of each tool over all training data sets.

Generally, our final models outperform the naive models for all clustering tools except MCL (difference in MAE of ≥ 0.025). Note that prediction errors are relatively low for both kinds of models as all clustering tools performed well on the selected data sets. On average, the predictions of the naive models have an MAE ≈ 0.1 , while those of our final models show an MAE ≈ 0.075 . Ordinary models generally outperform Lasso and Ridge regression models in terms of MAE. The general trend is MAE(ordinary) < MAE(lasso) < MAE(ridge). However, the differences between ordinary, Lasso and Ridge regression are very small.

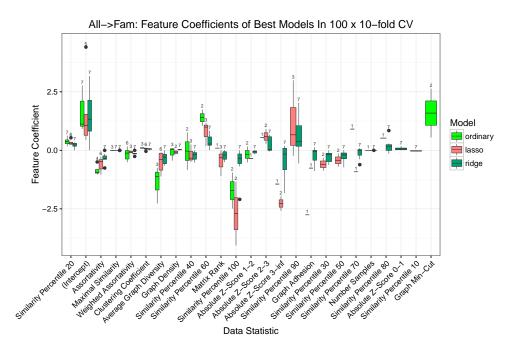


Fig. 6. This figure depicts the features and their average value in the different regression models for tool performance prediction. The features are sorted according to how often they have been selected by all models. We treated features with zero coefficient as not being in the model. The small number above each box indicates how often the feature was selected by the model represented by the box. Please note that the feature *Minimal Similarity* was never selected and thus is omitted from the figure.

Here, we want to point out the limitations of the models presented. A meaningful prediction is only possible in case the features of the unknown data set are in the same range as the features of the training data sets. We have chosen the ASTRAL data set with only up to 40% sequence similarity as we expected to observe here the most extreme feature distributions compared to data sets with higher sequence similarity.

Therefore, we have also tested the performance of the prediction with data sets not used for training. For that we have used the SCOP data set with proteins having 95% or less sequence identity; we proceeded as with the original data set and separated it also into the different classes. The error for the predicted F1 score with 0.083 for Lasso and 0.084 for Ridge regression was still remarkably small. Only the ordinary regression model showed a clear drop in performance with an average error of 0.191. This indicates that the ordinary regression is the most sensitive model with respect to unseen feature values. We will constantly update the model with new clustering results in order to further improve the quality and robustness of the models over time. To this point, the presented models should rather be regarded as a proof-of-concept.

Furthermore, we compared which data statistics have been chosen as features in the different types of models (see Figure 6). Features that have been chosen by all models clearly have predictive power for the tool performances. Examples for such features are the [10, 20]-Similarity Percentile, Assortativity, Maximal Similarity and Weighted Assortativity. The coefficients of the maximal similarity are very small compared to the other features, as this feature is not normalized and thus takes large values across the data sets.

The *Graph Diversity* measures whether a node in the similarity graph is very similar to only few other nodes (low diversity) or is equally similar to many nodes (high diversity). All model types chose this statistic as a predictor with negative impact on the tool performance. This might be explained by the fact that a very high diversity implies equal similarities between all nodes, leading to the lack of an actual cluster structure.

Interestingly, the selected *Similarity Percentile* statistics indicate that details of the similarity distribution have a large predictive power for the tool performance. For example, many pairwise similarities between the [10 - 20]-*Similarity Percentile* indicate a better tool performance while fewer pairwise similarities between the [90 - 100]-*Similarity Percentile* have the opposite effect.

Surprisingly, the *Clustering Coefficient* does not enter many models with a large coefficient. Equally surprising, given the performance difference between the class and fold data sets, is that the data set size is only very rarely chosen as a feature.

4.5. Predicting Tool Parameters

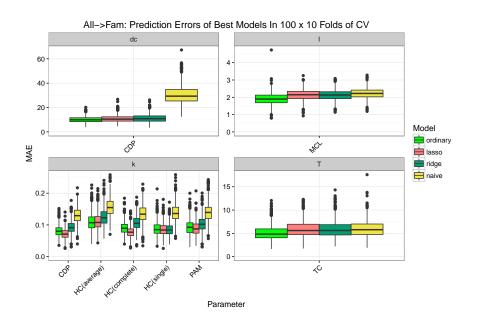


Fig. 7. The prediction performances for clustering tool parameters when trained on all data sets. Note that the various k parameters are summarized in one common plot and are normalized by the data set size.

As already previously discussed, a simple parameter suggestion valid for all data sets is not feasible due to the large variance in the protein families. Therefore, we applied the same pipeline as for the quality prediction to the parameters of the tools as well.

The results are summarized in Figure 7 and show a more mixed quality. We do not outperform the naive predictor for the threshold parameter T of TC and the Inflation parameter of MCL. We clearly outperform the naive predictor in the case of the dc parameter of CDP as well as the k parameters of all tools using such a parameter. Nevertheless, as discussed earlier, the k parameter is highly dependent on the way we have sampled the data sets, thus the predictive power has to be taken into account with care. Overall, the results indicate that an automated parameter prediction is not reliably possible with the presented simple models and may require more test data and more sophisticated models. In practice, the user has to resort to other methods for finding suitable parameters.²⁹

5. Conclusion

With this work, we have thoroughly investigated the performance of seven well-known and established clustering tools and have particularly investigated the behavior of the tools' parameters. We have observed that all tools perform quite well on these data sets. Nevertheless, the good performance can only be reached when exhaustive parameter finding by means of a comparison against a gold standard is performed. In practice, such gold standards are not available and consequently the parameters need to be retrieved by different means. When investigating the behavior of the parameters, we cannot suggest the user a single parameter for all data sets due to the high variance of the protein families. Only TC shows a consistent behavior of a parameter which is not directly dependent on the number of clusters. Overall, a single fixed parameter cannot account for the potential variety in the data sets. Even though the k parameter also shows a consistent behavior, it is not suitable for any recommendations as this behavior results from the way we have sampled our data sets which cannot be expected in practice.

Given this massive repository of clustering results at hand, we utilized it for learning regression models for predicting the expected performance of the investigated tools on previously unseen data sets. The presented model does outperform the naive model. Especially when considering that all clustering tools performed constantly well, the achieved prediction accuracy is notable. We also tested the models on data sets which have not been part of the training process. This can be seen as a strong indicator that it is generally possible to identify data sets suitable for a particular tool in an automated fashion. We have created a web-service where the user can upload a data set and receive the expected performance of the different tools. Please be advised that the model might fail when presented with data sets whose feature values are outside of the range of values the model was trained on. The web service is available under http://proteinclustering.compbio.sdu.dk. We will constantly enhance the model with additional data in order to cover a broader variety of data set features and thus creating more reliable predictions.

More generally speaking, the study shows that state-of-the-art clustering tools, when presented only with sequence similarities, have limitations with capturing the high diversity of protein families and require a specific parameter for every data set which cannot be easily provided in practice. Nevertheless, the performance achieved by the tools is certainly good enough to render this approach a viable one; probably the biggest limitation is due to the rather simple similarity function only using sequence data. Fed with more sophisticated similarity functions, these tools might be able to capture the nature of the data set even better.

References

- 1. M. L. Metzker, Nature reviews genetics 11, 31 (2010).
- 2. J. Baumbach, A. Tauch and S. Rahmann, Briefings in bioinformatics 10, 75 (2009).
- J. S. Bernardes, F. R. Vieira, L. M. Costa and G. Zaverucha, BMC bioinformatics 16, p. 1 (2015).
- 4. S. Whelan and N. Goldman, Molecular biology and evolution 18, 691 (2001).
- 5. N. K. Fox, S. E. Brenner and J.-M. Chandonia, Nucleic Acids Research 42, D304 (dec 2013).
- R. D. Finn, P. Coggill, R. Y. Eberhardt, S. R. Eddy, J. Mistry, A. L. Mitchell, S. C. Potter, M. Punta, M. Qureshi, A. Sangrador-Vegas *et al.*, *Nucleic acids research* 44, D279 (2016).
- V. Kunin, I. Cases, A. J. Enright, V. de Lorenzo and C. A. Ouzounis, *Genome biology* 4, p. 1 (2003).
- 8. J. Chen, B. Liu and D. Huang, BioMed Research International 2016 (2016).
- 9. L. Liao and W. S. Noble, Journal of computational biology 10, 857 (2003).
- S. F. Altschul, W. Gish, W. Miller, E. W. Myers and D. J. Lipman, *Journal of Molecular Biology* 215, 403 (oct 1990).
- 11. C. Wiwie, J. Baumbach and R. Röttger, *Nature Methods* 12, 1033 (sep 2015).
- 12. C. X. Chan, M. Mahbob and M. A. Ragan, BMC bioinformatics 14, p. 1 (2013).
- 13. M. C. De Souto, R. B. Prudencio, R. G. Soares, D. S. De Araujo, I. G. Costa, T. B. Ludermir and A. Schliep, Ranking and selecting clustering algorithms using a meta-learning approach, in 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), 2008.
- 14. B. Rost, Protein engineering **12**, 85 (1999).
- 15. T. Wittkop, J. Baumbach, F. P. Lobo and S. Rahmann, BMC Bioinformatics 8, p. 396 (2007).
- 16. A. Rodriguez and A. Laio, *Science* **344**, 1492 (jun 2014).
- 17. R Core Team, R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, (2014).
- 18. S. Dongen, A Cluster Algorithm for Graphs, tech. rep. (Amsterdam, The Netherlands, The Netherlands, 2000).
- M. Maechler, P. Rousseeuw, A. Struyf, M. Hubert and K. Hornik, *cluster: Cluster Analysis Basics and Extensions*, (2016). R package version 2.0.4 For new features, see the 'Changelog' file (in the package source).
- T. Wittkop, D. Emig, S. Lange, S. Rahmann, M. Albrecht, J. H. Morris, S. Böcker, J. Stoye and J. Baumbach, *Nature Methods* 7, 419 (jun 2010).
- 21. J. Handl, J. Knowles and D. B. Kell, *Bioinformatics* **21**, 3201 (may 2005).
- 22. M. E. J. Newman, *Physical Review E* 67 (feb 2003).
- A. Barrat, M. Barthlemy, R. Pastor-Satorras and A. Vespignani, Proc Natl Acad Sci U S A 101, 3747 (Mar 2004).
- 24. D. R. White and F. Harary, Sociological Methodology 31, 305 (2001).
- 25. R. Diestel, Graph Theory (Graduate Texts in Mathematics) (Springer, 2006).
- 26. N. Eagle, M. Macy and R. Claxton, Science 328, 1029 (2010).
- 27. C. J. Willmott and K. Matsuura, *Climate research* **30**, 79 (2005).
- T. Pahikkala, A. Airola and T. Salakoski, Speeding up greedy forward selection for regularized least-squares, in 2010 Ninth International Conference on Machine Learning and Applications, (Institute of Electrical & Electronics Engineers (IEEE), dec 2010).
- R. Röttger, P. Kalaghatgi, P. Sun, S. de Castro Soares, V. Azevedo, T. Wittkop and J. Baumbach, Bioinformatics, p. bts653 (2012).