

Supplementary

1. Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a technique that transforms the original set of correlated variables into a smaller set of uncorrelated variables known as principal components (PCs). These components are linear combinations of the original features, with the first component accounting for most of the variance seen in the data and the last component accounting for the least variance.

This analysis was performed using the ‘prcomp’ function from the ‘stats’ package in R, which standardizes the data before performing the decomposition (Venables & Ripley, 2002; R Core Team, 2024). Its base requirement is a numeric or complex matrix with the values called ‘x’. This data frame was scaled and centered by subtracting the mean and dividing by the standard deviation of each variable, ensuring all features contributed equally to the analysis. In the function this is handled by assigning ‘scale’ and ‘center’ to TRUE. The covariance matrix of the scaled data was then calculated to examine the relationships between the variables. This is different from the Pearson’s correlation coefficient which is also used separately in this research. PCA was performed through the eigen decomposition of this covariance matrix, where the eigenvectors correspond to the directions of the principal components and the eigenvalues represent their respective contributions to the overall variance.

The data was projected onto the first two principal components (see Fig. 5). With three components chosen based on the cumulative variance criterion, it was ensured that at least 95% of the total variance was retained (Table A1). To interpret the PCA results, scatter plots of the first few principal components were used to visually explore potential clusters in the transformed data space. A scree plot was also generated to evaluate the proportion of variance explained by each component, while a biplot provided insight into how the original variables contributed to the first two principal components (see Fig. 5). The principal component values which are used for the clustering can be calculated using the eigenvectors shown in Table A2, combined with the scaled and normalized values of the material characteristics.

Table A1: Importance of the principal components, including their standard deviation.

	<i>PC1</i>	<i>PC2</i>	<i>PC3</i>	<i>PC4</i>	<i>PC5</i>
<i>Standard Deviation</i>	2.0628	0.7058	0.4638	0.1596	0.0780
<i>Proportion of Variance</i>	0.851	0.0996	0.0430	0.0051	0.0012
<i>Cumulative Proportion</i>	0.851	0.9507	0.9937	0.9988	1.0000

Table A2: Eigenvalues of the material characteristics per principal component.

	<i>PC1</i>	<i>PC2</i>	<i>PC3</i>	<i>PC4</i>	<i>PC5</i>
<i>Water Content</i>	0.45362	-0.38972	0.42183	0.62854	-0.26334
<i>Wet Bulk Density</i>	-0.46509	-0.28225	0.40884	0.18347	0.70938

<i>Dry Bulk Density</i>	-0.46366	-0.23135	0.49495	-0.38475	-0.58179
<i>Particle Density</i>	-0.42726	-0.51899	-0.63258	0.32478	-0.20605
<i>Void Ratio</i>	0.42472	-0.66752	-0.09887	-0.56370	0.21564

2. Organic fraction calculations

For the calculation of organic fraction the particle density was needed, but values listed in the dataset were deemed to show ranges exceeding common end-member particle densities for peat and clay. This might be the consequence of measuring partially unsaturated samples, a result of sample storage, while assuming full saturation in the measurement and following calculations. The particle density of the samples ρ_s was recalculated using:

$$\rho_s = \frac{\rho_d}{1 - \frac{WC}{WC + 100}}$$

With ρ_d being the dry bulk density and WC being the water content. Initial water content and dry bulk density values were present in the dataset for all samples. Only for the oedometer test data the water content and dry bulk density after the end of the test were also available. Both the initial and final values result in a mostly linear pattern for the calculated particle density as a function of the dry bulk density, apart from samples with very low final dry bulk densities (Fig. A1). Particle densities calculated with the final dry bulk densities are consistently higher than their initial counterparts (Fig. A2). As samples are kept in saturated conditions during a test, the final particle density values are used for determination of the organic fraction.

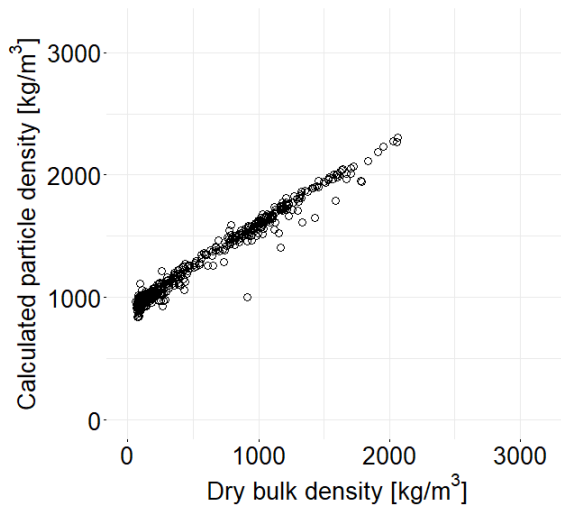


Fig. A1a: Calculated particle density as a function of the initial dry sample density reported for the entire dataset.

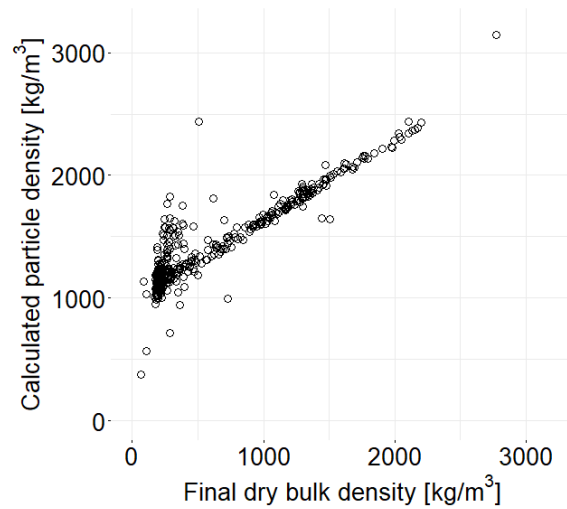


Fig. A1b: Calculated particle density as a function of the final dry sample density reported for the oedometer test samples.

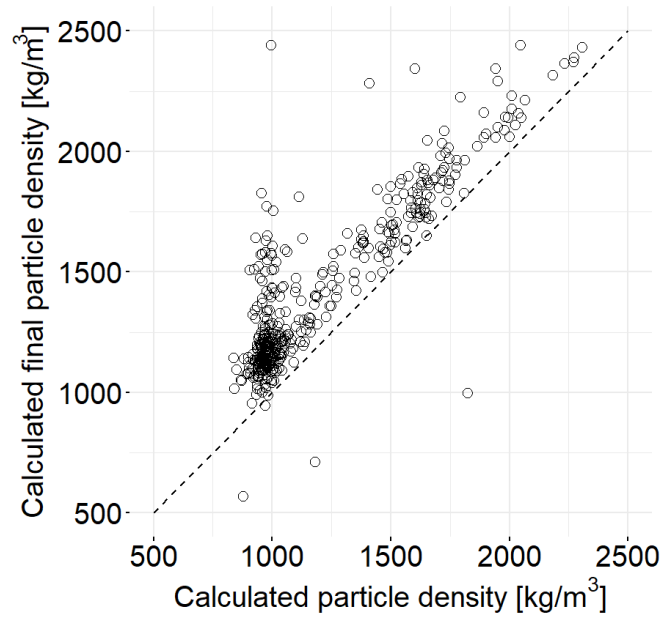


Figure A2: Calculated initial particle density vs calculated final particle density.

3. Clustering methods

3.1 K-Means Clustering

K-Means is a widely used partition-based clustering algorithm that groups data points into a predefined number of clusters, K , based on the shortest distance between the data point in question and the nearest cluster mean. The algorithm begins by randomly initializing K centroids and iteratively refines them by assigning each data point to the nearest centroid. The centroids are then updated by computing the mean of all data points in each cluster, also called the mean of the Voronoi sets. This process continues until the centroids no longer change significantly.

This method is applied by the ‘kmeans’ function in Rstudio and can be found in the stats package (R Core Team, 2024). It requires several input variables, of which the following have been used: a numeric data matrix ‘x’, a provided number of clusters or set of initial cluster centres called ‘centers’, the maximum number of iterations ‘iter.max’, and the amount of random initializations for the centres of clusters ‘nstart’, which is used when ‘centers’ is a number instead of a position matrix. The used data matrix is specified in the methods, as well as the number of clusters. The maximum amount of iterations was left to the default setting of 10, while the amount of initializations was set to 25. By default the algorithm of Hartigan and Wong is used (Hartigan & Wong, 1979), though other algorithms can be chosen.

One of the main advantages of using K-Means to cluster data points is the relative simplicity and straightforward way it works compared to other methods, which also leads to computational efficiency, especially for large datasets. However, the requirement to specify ‘centers’ in advance and its sensitivity to the initial placement of centroids can be seen as downsides of the method. Because of the sensitivity to initial placement, a significant amount of random starts is chosen to account for this potential problem. Additionally, K-Means assumes that clusters are spherical and of similar size, which may not apply to all datasets. The combination of these two limitations may result in unreliable clustering of the data in comparison to other methods (Fig. A3).

3.2 Gaussian Mixture Model (GMM)

Gaussian Mixture Modelling (GMM) is a probabilistic clustering method that assumes data points are generated from a mixture of Gaussian distributions. Unlike K-Means, which assigns each data point to a single cluster, GMM provides so-called soft assignments, meaning each data point has a probability of belonging to multiple clusters. Furthermore, GMM is more flexible than K-Means as it can model elliptical clusters with different sizes and orientations. GMM uses the Expectation-Maximization (EM) algorithm to iteratively reach convergence of the parameters of the Gaussian components, maximizing the likelihood of the observed data (Dempster et al., 1977; Moon, 1996). The convergence limit also implies that there is no maximum amount of iterations used as user-defined number.

In Rstudio the 'mclust' function, found in the package with the same name, is used to compute the GMM clusters (Scrucca et al., 2023). The following input variables have been used for this research, though more input variables can be inserted: a numeric vector, matrix or data frame of observations called 'data', an integer or integer vector specifying the amount of desired output clusters 'G', and 'initialization' defining the starting positions of the cluster centres. The output classifications of K-Means for a given amount of clusters serves as the starting positions for the GMM method. The combination of the two methods can lead to improved clustering compared to output of the individual use of the methods (Fig. A3).

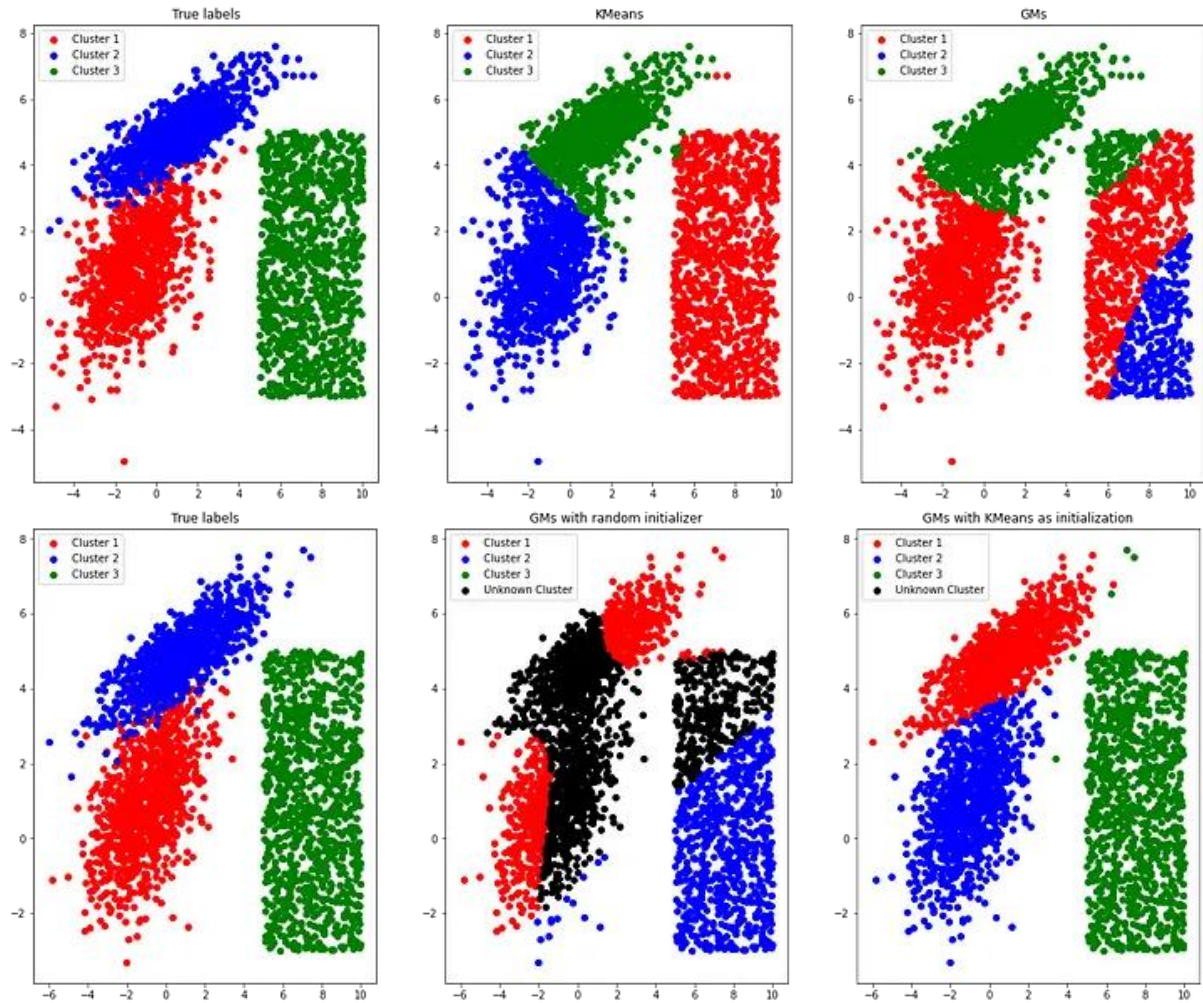


Figure A3: Differences between K-means and Gaussian Mixture Modelling clustering methods and a combination of the two when applied to a random dataset (Kubara, 2020). R-code showing the workflow and origin of the data is provided in Kubara (2020).

3.3 DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

DBSCAN is a density-based clustering algorithm that groups data points based on their density in a given region (Ester et al., 1996). Unlike K-Means and GMM, DBSCAN does not require a desired number of clusters beforehand. Instead, it relies on two parameters: ϵ (the maximum distance between data points in a cluster) and minPts (the minimum number of data points required to form a dense region). Data points are classified as core data points when within a radius ϵ the specified minimum number of data points is located. Border data points are located within radius ϵ of a core data point, but do not have enough data points within their own radius. Noise is defined as the data points outside of the radius ϵ of any core data point. Next, clusters are created of a network of core and border data points with inter-distance of radius ϵ or smaller. Core data points of clusters are therefore separated by at least radius ϵ .

The 'dbscan' function in Rstudio is part of the 'dbscan' package (Hahsler et al., 2019). It uses three main input variables: data matrix 'x', radius ϵ called 'eps', and the minimum amount of data points required 'minPts'. The last two are user-defined. For 'minPts' it is suggested to start with a value of the number of dimensions in the

dataset plus one ($d + 1$). Next, ‘eps’ can be found by using the function ‘kNNdistplot’ from the same package, which shows the distance to the closest neighbour. The elbow point indicating a sharp increase in distance is usually seen as the boundary with noise data points and the distance is taken for ‘eps’.

The key advantage of DBSCAN is its ability to detect clusters of arbitrary shapes and filter out noise in comparison to for example K-Means (Fig. A4). However, DBSCAN struggles with varying density clusters and can be sensitive to the choice of ϵ and the minimum amount of data points.

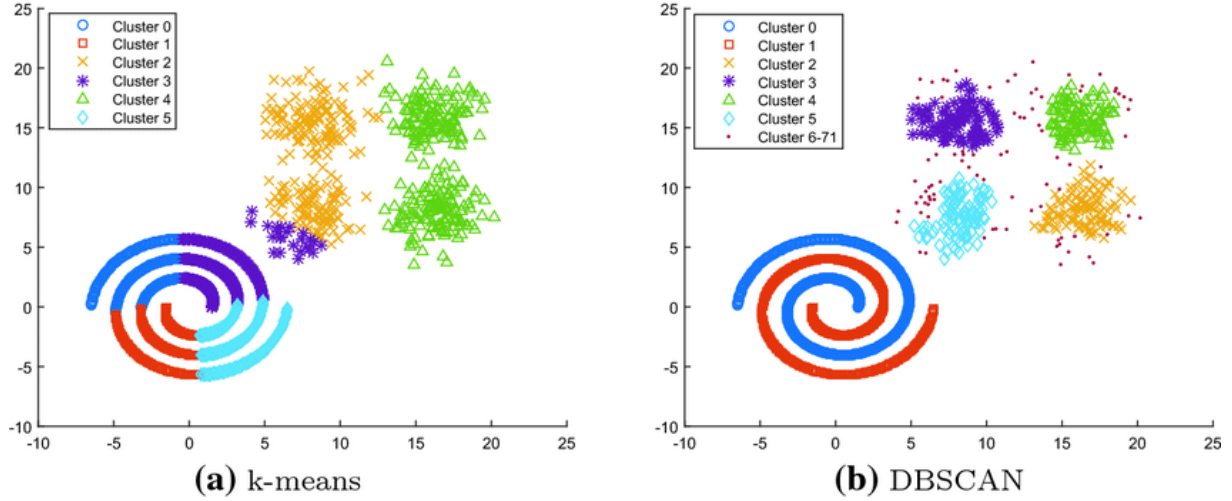


Figure A4: K-Means and DBSCAN clustering results on Spiralsquare data set (Liang et al., 2021). Note that the smaller red dots are considered outliers.

4. Cluster evaluation techniques

4.1 Silhouette score

The Silhouette Score is a metric used to evaluate the quality of clustering or the optimal number of clusters by measuring how similar each data point is to its own cluster compared to other clusters. For a given data point, the silhouette score s is defined as:

$$s = \frac{b - a}{\max(a, b)}$$

where a is the average distance to all data points within its cluster and b is the average distance from all data points in the closest neighboring cluster. The overall Silhouette Score for a clustering solution is obtained by averaging the individual s values of all data points; this mean value is used to compare the quality of different cluster configurations. The silhouette score ranges from -1 to 1, where values close to 1 indicate well-separated clusters, values near 0 suggest overlapping clusters, and negative values indicate misclassified data points. The Silhouette Score may not work well for clusters of varying densities and shapes.

Using the format function ‘silhouette’ from the ‘cluster’ package in Rstudio it is possible to calculate the Silhouette Score (Rousseeuw, 1987; Maechler et al., 2023). This function requires two input variables: a vector in which the cluster codes are stored called ‘x’, and a distance matrix ‘dist’ following from the command ‘dist’ found in the ‘stats’ package applied to the used dataset (R Core Team, 2024). The cluster output of the GMM was taken as input for the ‘x’ and ‘dist’ followed from the method described above.

The Silhouette Score for our dataset was calculated for GMM results implementing two to ten clusters. The progression of the score can be seen in Fig. A5. It shows that the best separation of cluster is achieved when three clusters are used. Furthermore it is visible that using four clusters is better than using more clusters or using 2 clusters.

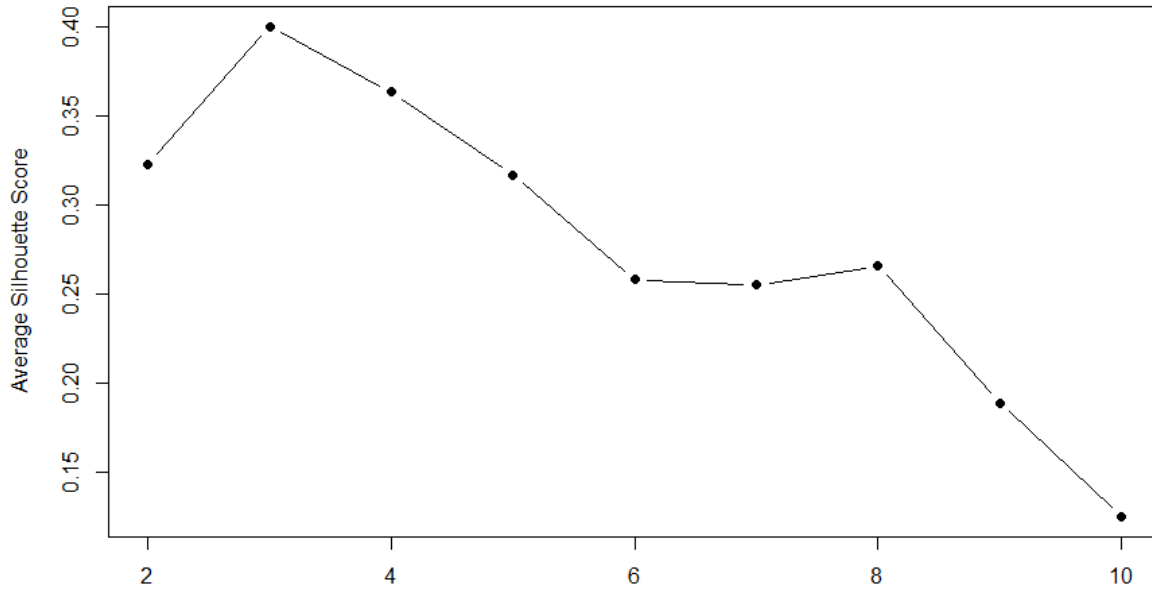


Figure A5: Average Silhouette Score for Gaussian Mixture Model cluster per number of clusters.

4.2 Bayesian Information Criterion (BIC) Score

The Bayesian Information Criterion (BIC) is a model selection criterion used to compare different clustering outputs of K-Means & GMM. It balances model complexity and goodness of fit, penalizing excessive parameters to avoid overfitting. The BIC score is computed as:

$$BIC = k \log n - 2 \log L$$

where k is the number of parameters in the model, n is the number of data points, and L is the likelihood of the data given the model (Wit et al., 2012). A lower BIC score indicates a better model, making it useful for selecting the optimal number of clusters. While effective, BIC assumes that data follow an underlying probabilistic distribution, which may not always be the case. Additionally, BIC tends to favor simpler models, sometimes underestimating the true number of clusters (Wit et al., 2012).

The BIC score is automatically calculated by the 'mclust' function described in the GMM paragraph. It can be accessed as one of the output list elements. Fig. A6 shows the BIC Score (a) and the BIC Slope (b) per number of clusters. The BIC score decreases over the entirety of the evaluated number of clusters, apart from the situation with 10 clusters. This means that using nine clusters would be the best option. However, as the aim is to keep the clustering simple, it is also possible to look at the BIC slope (Fig. A6.b) to examine the effect of adding an additional cluster. This method shows that increasing the number of clusters from two to three adds the most value.

Further increasing the number of clusters is beneficial up to five clusters, after which the added value of each additional cluster is <100.

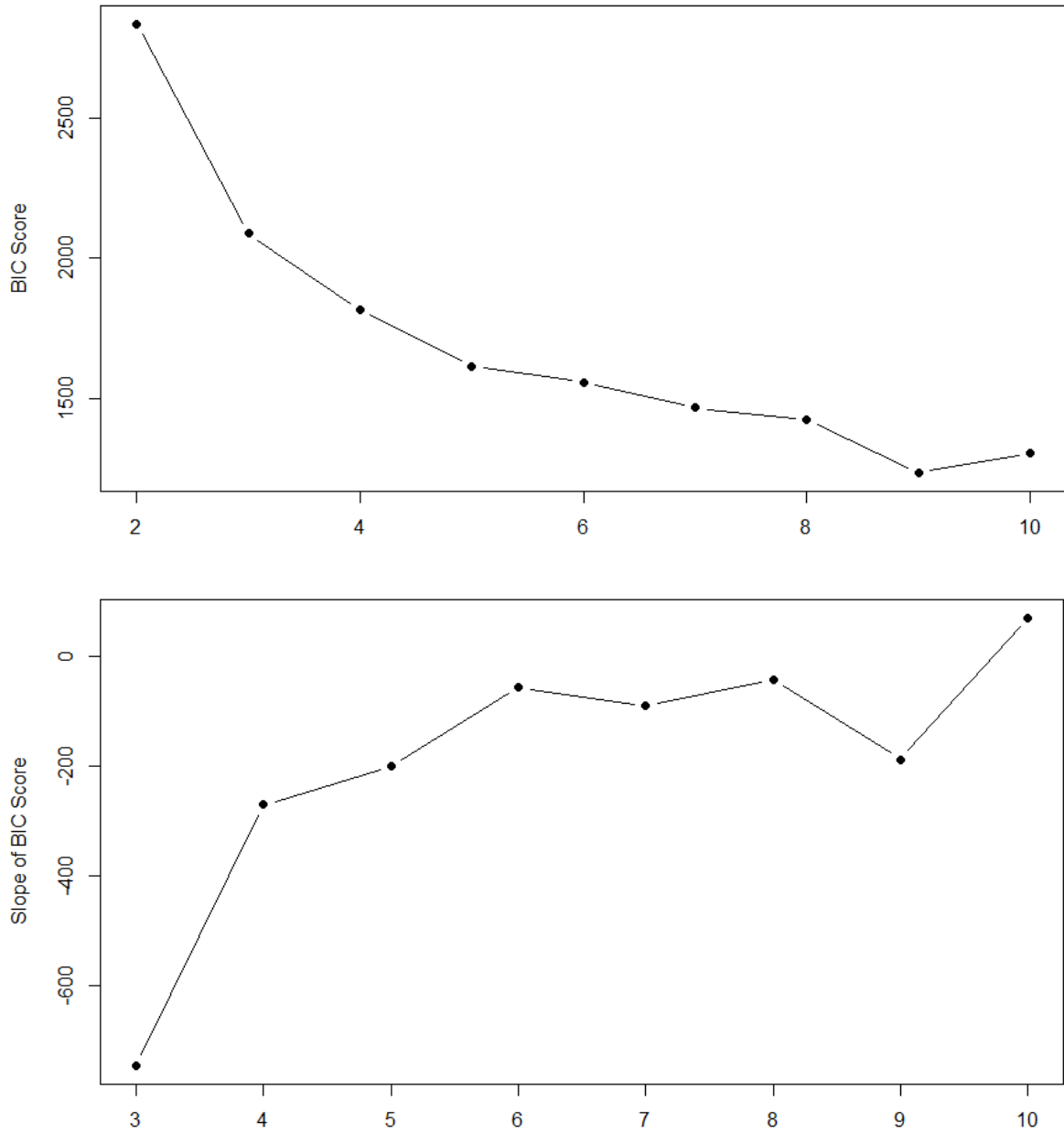


Figure A6: (a) BIC Score per number of clusters and (b) BIC Score Slope per number of clusters

4.3 Jensen-Shannon Divergence

Jensen-Shannon Divergence (JSD) is a measure of similarity between probability distributions and is used to assess clustering quality by comparing the distributions of different clusters. It is based on Kullback-Leibler (KL) divergence but is symmetric and always finite, making it more stable. The JSD between two distributions P and Q is given by:

161

$$JSD(P \parallel Q) = \frac{1}{2} D_{KL}(P \parallel M) + \frac{1}{2} D_{KL}(Q \parallel M)$$

162

where $M = \frac{1}{2}(P + Q)$ and D_{KL} is the KL divergence (Endres & Schindelin, 2003). Given that $P(x)$ and $Q(x)$ span

163

the same domain, the KL divergence from Q to P is defined as (Kullback & Leibler, 1951):

164

$$D_{KL}(P \parallel Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

165

A lower JSD value indicates that the clusters are more distinct. JSD is useful in applications where data is represented as probability distributions, which is why it has been chosen to assess the quality of the GMM results.

166

167

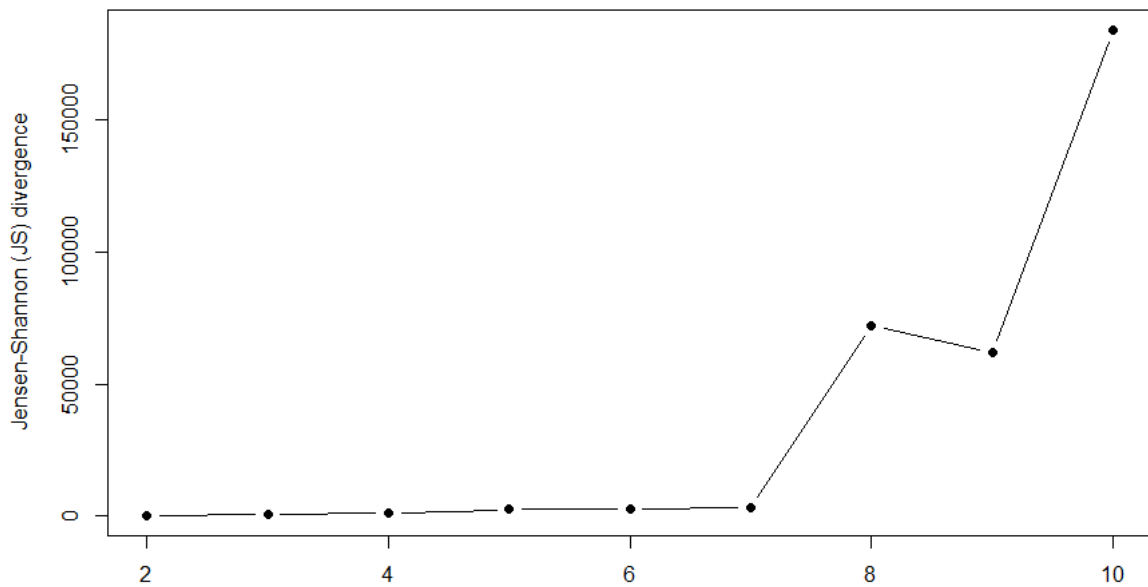
The JS Divergence scores per number of clusters can be seen in Fig. A7. Scores are relatively low up to seven

168

clusters, indicating distinct clusters. A relatively small increase in the JSD score can be seen from four to five

169

clusters, which shows a small preference for four clusters or less.



170

171

Figure A7: Average Jensen-Shannon Divergence Score, calculated using the Kullback-Leibler Divergence, for the entire dataset

172

per number of clusters.

173

5. Cluster robustness

174

Insight in the robustness of individual clusters can be achieved when reviewing the clustering results with three

175

and five clusters. When three clusters are used, the Clay and Peat clusters are largely retained, while the Clayey

176

peat and Decomposed peat clusters are merged, indicating that these clusters have most similar material

177

characteristics algorithmically. When five clusters are used, all clusters are retained in their current extent, except

178

for the Clayey peat cluster, which is separated at approximately 1100 kg/m³ for the wet bulk density and 240% for

179

the water content. This review of clusters points out that Clayey peat cluster is most variable, but is material

180

characteristic-wise more aligned with peats than clays. The adaptability of Clayey peat is reasonable, given the

181

cluster covers the shift between end-members clay and peat. The fact that Clayey peat merges with Decomposed

182

peat when three clusters are used, highlights that its material characteristics mostly resemble end-member peat and

183

thus that organic matter is controlling characteristics even at relatively low contents.

185

References

- 186 Dempster, A.P., Laird, N.M., Rubin, D.B. (1977). "Maximum Likelihood from Incomplete Data via the EM
187 Algorithm". Journal of the Royal Statistical Society, Series B. 39 (1): 1–
188 38. CiteSeerX 10.1.1.163.7580. doi:10.1111/j.2517-6161.1977.tb01600.x. JSTOR 2984875.
- 189 Endres, D.M.; Schindelin J.E. (2003). "A new metric for probability distributions" (PDF). IEEE Trans. Inf.
190 Theory. 49 (7): 1858–1860. doi:10.1109/TIT.2003.813506. hdl:10023/1591. S2CID 14437777
- 191 Ester, M., Kriegel, H., Sander, J., Xiaowei, X. (1996). A Density-Based Algorithm for Discovering Clusters in
192 Large Spatial Databases with Noise. Institute for Computer Science, University of Munich. *Proceedings of 2nd*
193 *International Conference on Knowledge Discovery and Data Mining (KDD-96)*, 226–
194 231. <https://dl.acm.org/doi/10.5555/3001460.3001507>
- 195 Hahsler, M., Piekenbrock, M., Doran, D. (2019). "dbscan: Fast Density-Based Clustering with R." *Journal of*
196 *Statistical Software*, *91*(1), 1-30. doi:10.18637/jss.v091.i01 <<https://doi.org/10.18637/jss.v091.i01>>.
- 197 Hartigan, J.A. and Wong, M. A. (1979). Algorithm AS 136: A K-means clustering algorithm. *Applied Statistics*, 28,
198 100–108. doi:10.2307/2346830.
- 199 Kubara, K. (2020). Gaussian Mixture Models vs K-Means. Which One to Choose? Comparing the Performance
200 of Two Popular Clustering Algorithms. Written 08-10-2020, available at [https://medium.com/data-](https://medium.com/data-science/gaussian-mixture-models-vs-k-means-which-one-to-choose-62f2736025f0)
201 [science/gaussian-mixture-models-vs-k-means-which-one-to-choose-62f2736025f0](https://medium.com/data-science/gaussian-mixture-models-vs-k-means-which-one-to-choose-62f2736025f0)
- 202 Kullback, S., Leibler, R.A. (1951). "On information and sufficiency". *Annals of Mathematical Statistics*. 22(1):79–
203 86. doi:10.1214/aoms/1177729694. JSTOR 2236703. MR 0039968
- 204 Liang, J., Cui, J., Wang, J., Wei, W. (2021). Graph-based semi-supervised learning via improving the quality of
205 the graph dynamically. *Machine Learning*. 110. 1-44. 10.1007/s10994-021-05975-y.
- 206 Maechler, M., Rousseeuw, P., Struyf, A., Hubert, M., Hornik, K.(2023). *cluster: Cluster Analysis Basics and*
207 *Extensions*. R package version 2.1.6.
- 208 Moon, T.K. (1996). "The expectation-maximization algorithm," in *IEEE Signal Processing Magazine*, vol. 13, no.
209 6, pp. 47-60, doi: 10.1109/79.543975.
- 210 R Core Team (2024). R: A language and environment for statistical computing. R Foundation for Statistical
211 Computing, Vienna, Austria. URL: <https://www.R-project.org/>
- 212 Rousseeuw, P.J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J.*
213 *Comput. Appl. Math.*, 20, 53–65.
- 214 Scrucca L., Fraley C., Murphy T.B., Raftery A.E. (2023). *Model-Based Clustering, Classification, and Density*
215 *Estimation Using mclust in R*. Chapman & Hall/CRC, ISBN: 978-1032234953, <https://mclust-org.github.io/book/>
- 216 Venables, W.N., Ripley, B.D. (2002). *Modern Applied Statistics with S*, Springer-Verlag.

217 Wit, E., van den Heuvel, E., Romeyn, J. (2012). "All models are wrong...': an introduction to model
218 uncertainty" (PDF). *StatisticaNeerlandica*. 66 (3): 217–236. doi:10.1111/j.1467-9574.2012.00530.x.,
219 S2CID7793470

220