



Reduced multidimensional scaling

Emmanuel Paradis¹ 

Received: 22 May 2020 / Accepted: 27 May 2021 / Published online: 5 June 2021

© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2021

Abstract

Dimension reduction is a common problem when analysing large data sets. The present paper proposes a method called reduced multidimensional scaling based on performing an initial standard multidimensional scaling on a reduced data set. This method faces the problem of finding a representative reduced sample. An algorithm is presented to perform this selection based on alternating sampling in outlier areas and observations in high density areas. A space is then constructed with the selected reduced sample by standard multidimensional scaling using pairwise distances. The observations not included in the reduced sample are then projected on the constructed space using Gower's formula in order to obtain a final representation of the whole data set. The only requirement is the ability to compute distances among observations. A simulation study showed that the proposed algorithm results performs well to detect outliers. Evaluation of running times suggests that the proposed method could run in a few hours with data sets that would take more than one year to analyse with standard multidimensional scaling. An application is presented with a dataset of 9547 DNA sequences of human immunodeficiency viruses.

Keywords Dimension reduction · Distance data · HIV · Multidimensional scaling

1 Introduction

Analysing very large data sets is now a common problem in many scientific fields. For instance, research projects in environmental sciences or in genomics typically require to analyse data sets made of millions of observations each with hundreds or thousands of variables (Stephens et al. 2015). A common objective of such analyses is to find *structure* defined as clusters of discrete groups of observations. In this situation, the main output is a graphical representation into a space made of a few dimensions which can be displayed as standard scatterplots.

✉ Emmanuel Paradis
Emmanuel.Paradis@ird.fr

¹ ISEM, IRD, CNRS, EPHE, University of Montpellier, Montpellier, France

The methods used for such analyses face three main challenges. First, handling, analysing, and identifying observations in the final space could be very costly in terms of computational resources. As a matter of fact, traditional multivariate methods of aggregation or projection are not operational with data sets made of millions of observations. Second, researchers want to represent data sets with many variables into a small number of dimensions, either to obtain a meaningful graphical representation, or to find a small subset of the original variables. To be effective, this dimension reduction should be based on an assessment of how much of the original variation in the whole data set is represented in the reduced space. Third, whether a specific data set can be represented as discrete clusters is not a general rule and depends on the data themselves.

A number of algorithms have been developed to handle large matrices and, in particular, decompose them in order to perform multivariate analyses on large data sets. Eigendecomposition of a matrix can be done iteratively by expectation–maximization which avoids inverting the matrix of eigenvectors (Roweis 1998). Standard eigendecomposition is computationally costly when there are many variables implying to calculate a large variance–covariance matrix. An approach based on bidiagonalization was proposed by Baglama and Lothar (2005) for the iterative computations of singular values of large matrices. Halko et al. (2011) reviewed several algorithms based on random matrices that perform various matrix decompositions. There are several implementations of these methods, in particular in R (R Core Team 2021), with the packages *RSpectra* (Qiu and Mei 2019), *irlba* (Baglama et al. 2019), *flashpcaR* (Abraham and Inouye 2014), *onlinePCA* (Degras and Cardot 2016), *idm* (D’Enza et al. 2018), and *rsvd* (Erichson et al. 2019).

From a statistical inferential point of view, and without consideration on sample size, there are two main approaches to the problem of dimension reduction that differ mainly by their respective objective. In the first approach, the reduced space aims to be representative, as much as possible, of the dispersion of the observations in the original space (e.g., principal component analysis, PCA). In the second approach, some groups actually exist and are known in the data, and the reduced space optimizes the discrimination of these groups using data with known assignment (e.g., discriminant analysis). This second approach is related to, but distinct from, a third approach where it is assumed that some grouping structure exists but the assignments of the observed data to these groups are unknown (e.g., k-means clustering): in this situation, group assignment, instead of dimension reduction, is the main objective. In this third case, although the basic approach assumes that groups actually exist (e.g., Lloyd 1982; Venables and Ripley 2002), statistical testing of the reality of these groups is possible, for instance, Beugin et al. (2018) developed an approach to test the statistical significance of groupings inferred by k-means clustering with genetic data.

The transposition of this framework to the situation of very large data sets is not straightforward. In this paper, I present an approach that aims to address the above issues. This approach is very general and can be applied to any type of data as long as it is possible to calculate a distance between two observations. The next section explains the computational details of the proposed approach. A simulation study aimed at characterizing some of the statistical properties of the approach is presented in Sect. 3.

An application with a data set of 9547 DNA sequences is given in Sect. 4, and Sect. 5 discusses the general implications of this work.

2 Computational details

The data are assumed to be made of a matrix X with n rows and p columns. It is also assumed that it is possible to compute a square $n \times n$ matrix denoted as Δ which stores the pairwise, symmetric distances among observations ($\delta_{ij} = \delta_{ji}$, with i and $j = 1, \dots, n$). Our goal is to represent the n observations into a Euclidean space of dimension k ($< p$). A widely used method to achieve this goal is multidimensional analysis scaling (MDS) which is performed by the eigendecomposition of the doubly-centred distance matrix:

$$-\frac{1}{2}J\Delta^2J, \tag{1}$$

with $J = I - \frac{1}{n}11^T$. The matrix of coordinates in the final Euclidean space, denoted as Z , are computed with:

$$Z = V\Lambda^{\frac{1}{2}},$$

where V is an $n \times k$ matrix with the first k eigenvectors extracted from the decomposition of (1), and Λ is a diagonal $k \times k$ matrix with the first k eigenvalues ($\lambda_1, \dots, \lambda_k$). Z has thus n rows and k columns. Note that we make no assumption on the type of variables in X which can be continuous and/or discrete. This last feature explains, at least in part, the success of MDS because many fields (e.g., psychology, molecular biology) consider variables which are not continuous but there are methods to compute distances among observations (e.g., subjects, molecular sequences) so that the matrix Δ can be calculated in many situations.

Matrix decomposition is usually computationally costly. For instance eigendecomposition is limited to square matrices with a few thousands rows (see an overview of these methods applied to population genomic data in Paradis 2020). Instead of considering the full matrix Δ for decomposition, an alternative approach is to consider a subset of it made of m rows and m columns (with $m \ll n$), calculate the required numbers of axes from these m observations, and project the remaining $n - m$ observations on these axes. The core of this approach is an algorithm that subsamples X and was first given in Paradis (2018). This algorithm is based on distances among observations but does not require to compute the full matrix Δ , only some of its elements δ_{ij} . This is detailed in Algorithm 1.

The aim of this algorithm is to select a few observations that are representative, in a specific way, of the overall distribution of the n rows of X in a multidimensional space. Step 4 selects observations that are at the periphery (or outliers) of this space, while step 6 selects those that are close to its center. The alternance of these two steps results in an equal representation of these two types of observations.

Algorithm 1 Selection of a reduced sample of observations from X .

- 1: Choose m so that $1 < m < n$. Define a vector ζ of length m .
 - 2: Select one observation i at random among the n . Store i in ζ .
 - 3: Compute the vector of distances δ_{ij} with $j = 1, \dots, n$ and $j \notin \zeta$.
 - 4: Find j' so that $\delta_{ij'}$ is max of the distances calculated at step 3. Store j' in ζ .
 - 5: Compute the vector of distances $\delta_{j'i}$ with $j = 1, \dots, n$ and $j \notin \zeta$.
 - 6: Select i so that $\delta_{j'i}$ is median of the distances calculated at step 5. Store i in ζ .
 - 7: Repeat steps 3–6 until m values are stored in ζ .
-

A version of the above algorithm has step 6 modified so that i is chosen randomly which is more likely to select observations that are present in regions of the space with high density, so that the alternance of steps 4 and 6 now gives equal representation of outliers and high density areas. This second version of the algorithm was used in the simulations and the application below.

Because the above algorithm uses distances among individuals, it can be applied to any type of data as long as pairwise distances can be calculated. Furthermore, it does not require to compute the full matrix Δ which can be very costly if n is large. Four versions of Algorithm 1 have been implemented in R for the present study:

- A general version with an argument `FUN` which accepts any R function to compute the distance between two vectors.
- A version that uses Euclidean distance coded with efficient R vectorized operations.
- A version that calls a C routine implementing Euclidean distance.
- A version for DNA sequences using C code borrowed from `ape` (Paradis and Schliep 2019).

Figure 1 shows Algorithm 1 in action. Two data sets were simulated with $n = 10,000$ or $n = 1,000,000$, respectively, and $p = 10$ in both cases. Half of the points were drawn from a normal distribution with mean equal to -2 and the other half with mean equal to 2 . The algorithm was set with $m = 100$. This illustrates the ability of the algorithm to select observations spreading the range of the data independently of sample size as well.

Clearly, it is important to assess whether a subsample of the data is likely to represent correctly the complete matrix X . A solution to this problem is to assess whether increasing the value of m provides a stable representation of the subsample after performing the MDS. The logic of this approach is that if an MDS performed on m observations represents correctly the data, then an MDS on $m + 1$ observations should give a similar representation. To assess this, we compute the discrepancy between the original distances and the Euclidean distances inferred from the MDS calculated with:

$$d_{ij} = \sqrt{(z_i - z_j)^T(z_i - z_j)},$$

where z_i and z_j are the vectors of coordinates for observations i and j in the MDS-based space. At this stage, there is no consideration of the final number of dimensions, so these coordinates are based on the m dimensions resulting from the decomposition of the $m \times m$ distance matrix. The sum of the squared differences between the two sets of pairwise distances are used to assess the ‘quality’ of the representation for each

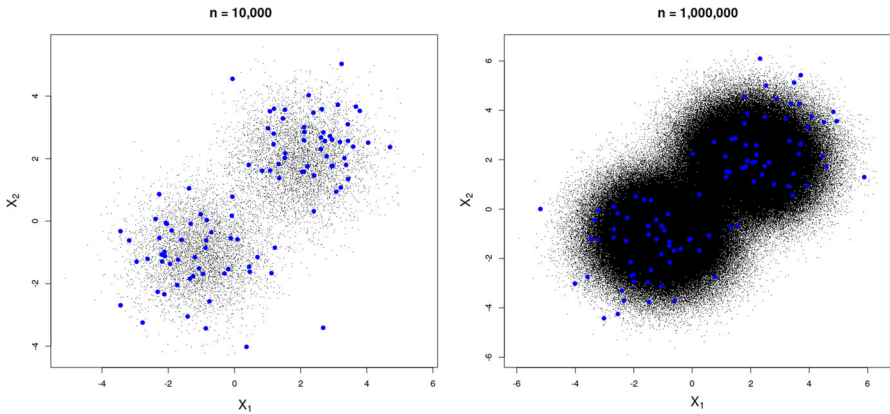


Fig. 1 Illustrations of the selection algorithm presented in the text with $n = 10^4$ and $n = 10^6$ observations (shown with small black dots); the selected observations are shown with large blue dots ($p = 10$ and $m = 100$ in both cases; only the first two dimensions are shown) (color figure online)

value of m . However, this criterion is expected to increase with increasing values of m , thus it is scaled by the mean of the differences:

$$\frac{\sqrt{\frac{1}{m'} \sum_{i,j} (\delta_{ij} - d_{ij})^2}}{\frac{1}{m'} \sum_{i,j} \delta_{ij} - d_{ij}}, \tag{2}$$

with $m' = m(m - 1)/2$ is the number of pairs among the m observations. An appropriate value of m is found when Eq. 2 gives a similar value for $m + 1$; in practice, the absolute difference between the two criteria is used.

Once a representative sample has been obtained, it is possible to project the remaining $n - m$ observations using Gower’s (1968) formula. Suppose we performed an MDS on m observations resulting in the $m \times k$ matrix of coordinates Z , and that we have one additional observation for which we have the distances to the m previous ones stored in the vector $\tilde{\delta}$. Then, the k coordinates of this additional observation are computed with:

$$\frac{1}{2} \left(Z^T Z \right)^{-1} Z \left[\text{diag} \left(Z Z^T \right) - \tilde{\delta}^2 \right]. \tag{3}$$

By contrast to a previous approach (Paradis 2018), the reduced space may have more than two dimensions.

3 Simulation study

Two sets of simulations were run to assess some properties of the method presented in this paper. The basic model used in these simulations was to simulate a matrix X with n rows and p columns where the elements x_{ij} follow a normal distribution

Table 1 Computing times (in seconds)

n	Method	p				
		100	1000	10^4	10^5	10^6
100	MDS	0.005	0.025	0.26	5.9	74
1000	MDS	0.9	5.6	66	1422	–
	RMDS	0.44	9	110	2315	–
10^4	MDS	1582	–	–	–	–
	RMDS	4.2	102	1602	–	–
10^5	RMDS	47	1328	–	–	–
10^6	RMDS	410	–	–	–	–

All times less than one minute were averaged over ten replications
MDS standard MDS, *RMDS* reduced MDS (present method), n number of observations, p number of variables

with variance unity and means determined by the model. The details of the simulation protocols are further explained in the following subsections.

3.1 Computing times

The first set of simulations aimed at assessing computing times of the method presented in this paper compared to standard MDS. X was simulated with $n = \{100, 1000, 10^4, 10^5, 10^6\}$, and the same set of values for p , although not all combinations of these two parameters were considered. The variables were independently drawn from a normal distribution with mean equal to zero and standard-deviation equal to one. The simulated matrix X was analysed with a standard MDS and with the present method (RMDS). The computing times were recorded; in the case of MDS, these included the time needed to compute Δ and perform the standard MDS as described above; in the case of RMDS, these included the times to perform Algorithm 1, find the optimal value of m , and project the observations on the reduced MDS-based space.

The results are presented in Table 1. The computing times of the standard MDS scaled roughly linearly with p . On the other hand, they appeared to increase dramatically with n so that they were not assessed with $n > 10^4$. On the other hand, the present method (RMDS) scaled linearly with n and was almost 400 times faster than MDS with $n = 10^4$ and $p = 100$.

3.2 Precision

In the second set of simulations, we addressed the question of the precision of the results. Specifically, we assessed whether the representations of the observations in the MDS-based space is close to the original pairwise distances. The size of the data was fixed to $n = p = 1000$. A fraction of the n observations, denoted as g (< 1), was simulated with a mean equal to two while the other observations were simulated with mean zero. This made possible to simulate data sets with two groups

Table 2 Precision of three MDS-based methods

<i>g</i>	Method	<i>S</i>		<i>S'</i>	
		Mean	SD	Mean	SD
0	MDS	0.922	0.000	19 × 10 ⁶	17,602
	RMDS	0.953	0.002	20 × 10 ⁶	82,217
	RMDS(2)	0.954	0.002	20 × 10 ⁶	74,724
0.001	MDS	0.933	0.001	19 × 10 ⁶	24,487
	RMDS	0.913	0.006	18 × 10 ⁶	241,665
	RMDS(2)	0.953	0.007	20 × 10 ⁶	292,649
0.01	MDS	0.913	0.000	19 × 10 ⁶	23,618
	RMDS	0.916	0.007	19 × 10 ⁶	297,440
	RMDS(2)	0.945	0.018	20 × 10 ⁶	593,160
0.1	MDS	0.738	0.001	16 × 10 ⁶	21,397
	RMDS	0.752	0.002	16 × 10 ⁶	62,871
	RMDS(2)	0.783	0.088	17 × 10 ⁶	2,717,905

MDS standard MDS, *RMDS* reduced MDS (present method), *RMDS(2)* reduced MDS with random sampling of the *m* observations

in unequal proportions. This parameter took the values $g = \{0, 0.001, 0.01, 0.1\}$. The simulated data were analysed with three methods: the standard MDS, the RMDS using Algorithm 1, and the RMDS where the *m* observations were selected randomly. In order to assess the precision of the results, two measures were used: the classical stress calculated with Kruskal’s formula (Kruskal 1964):

$$S = \sqrt{\frac{\sum_{i,j}^n (\delta_{ij} - d_{ij})^2}{\sum_{i,j}^n \delta_{ij}^2}}$$

and a measure of the accuracy of the inferred distances calculated with:

$$S' = \sum_{i,j}^n \frac{(\delta_{ij} - d_{ij})^2}{\delta_{ij}}$$

The simulations were replicated 100 times for each set of parameter values. The mean and standard-deviation (sd) of both *S* and *S'* were calculated over the 100 replications.

The sd of both *S* and *S'* were much lower for the MDS than both variants of the RMDS method. Considering that the results with the MDS method provides a measure of the ‘baseline’ precision, it appeared that the RMDS method performed usually very close in comparison (Table 2). Particularly, the present method performed very well when $g = 0.001$ or $g = 0.01$. The RMDS method with random sampling (RDMS(2) in Table 2) performed better than with Algorithm 1 only when $g = 0$; in other cases, the latter performed better although not as good as the full MDS.

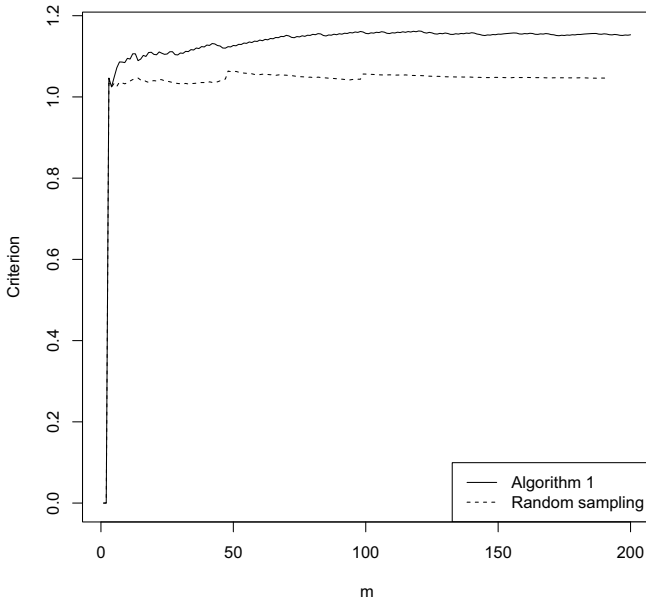


Fig. 2 Values of the criterion calculated with Eq. 2 with respect to the reduced sample size (m)

4 Application

The RMDS method was applied to a set of DNA sequences downloaded from the Los Alamos National Laboratory (<https://www.hiv.lanl.gov/content/sequence/NEWALIGN/align.html>). This data set consisted of 9547 sequences of the GAG protein of the human immunodeficiency virus (HIV) originally with 1944 sites (columns). The sequences were already aligned (alignment ID: 118AP3) although a few sites (16) consisted of alignment gaps only and were removed prior to analysis, so the data analysed had 1928 sites. In a first step, two reduced samples were selected with Algorithm 1 on one side, and by random sampling on the other. In both cases, the criterion (2) converged rather quickly: it reached a stable value with $m = 200$ for Algorithm 1, and $m = 192$ for random sampling (Fig. 2). On the other hand, the percentage of variance explained by the MDS on these respective subsamples were quite different: the first five eigenvalues explained 61, 16, 9, 3, and 2% for Algorithm 1, and 21, 15, 4, 3, and 2% for random sampling.

The plots of the 9547 projected observations were quite different between the different methods. The RMDS plots using Algorithm 1 showed one main cluster on axis 1 and a few outliers, whereas three main clusters were observed on axes 4 and 5 (Fig. 3). On the other hand, using random sampling showed three main clusters on axes 1 and 2 (Fig. 4).

Although the data set was relatively large, we analysed it with a standard MDS. This last analysis took about one hour to complete, whereas the two previous ones took each less than one minute. The MDS on the full data showed a pattern quite similar to the RMDS with random sampling (Fig. 5). Axis 3 shows evidence of a small cluster

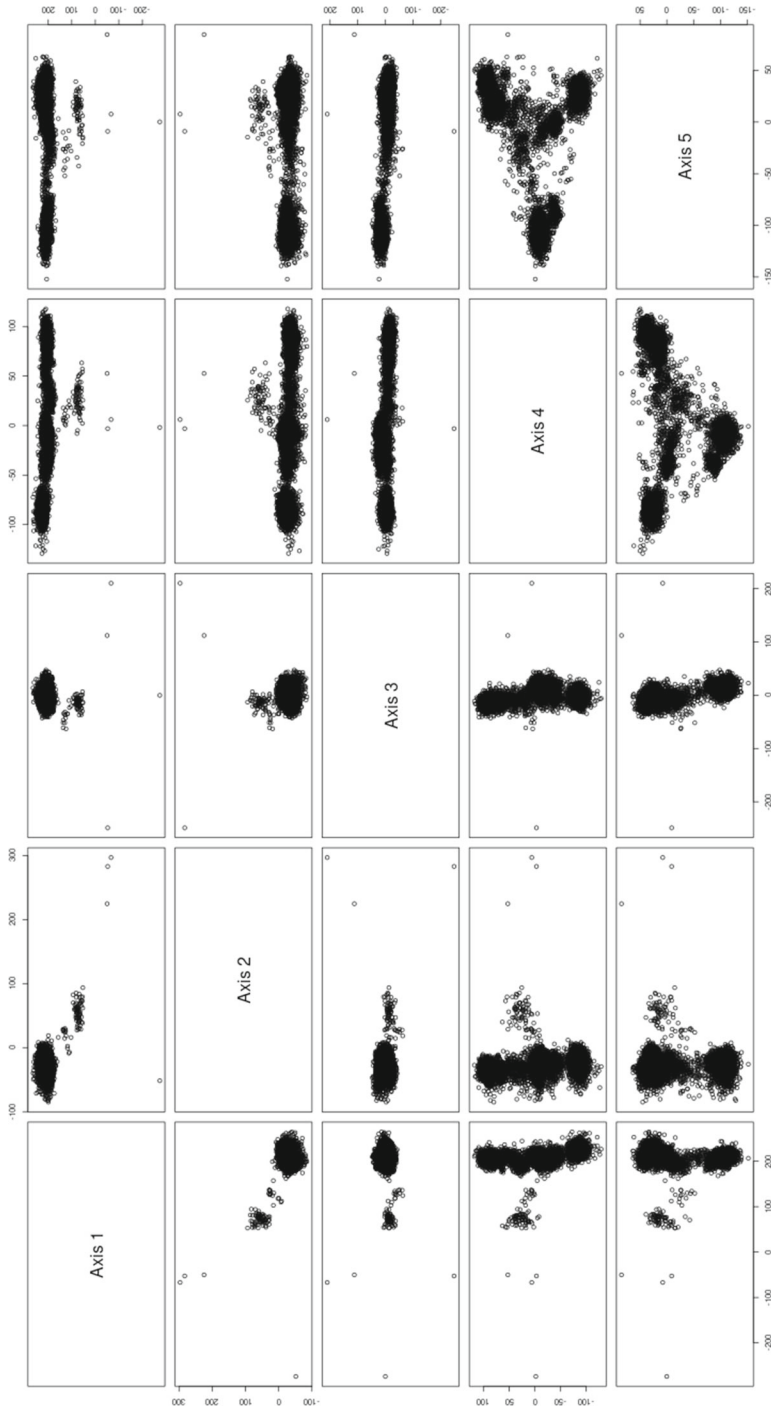


Fig. 3 Axes 1–5 of the RMDS on the HIV data using Algorithm 1

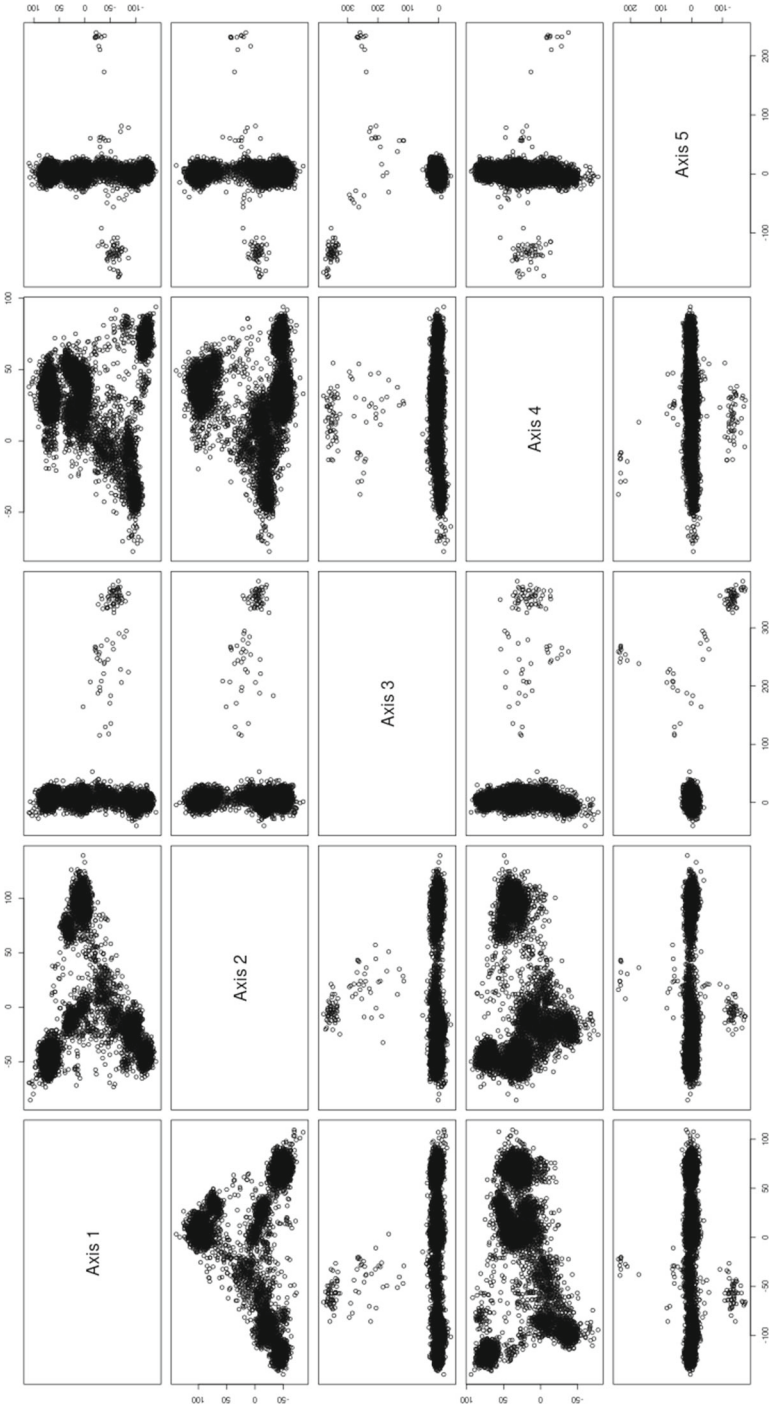


Fig. 4 Axes 1–5 of the RMDs on the HIV data using random sampling

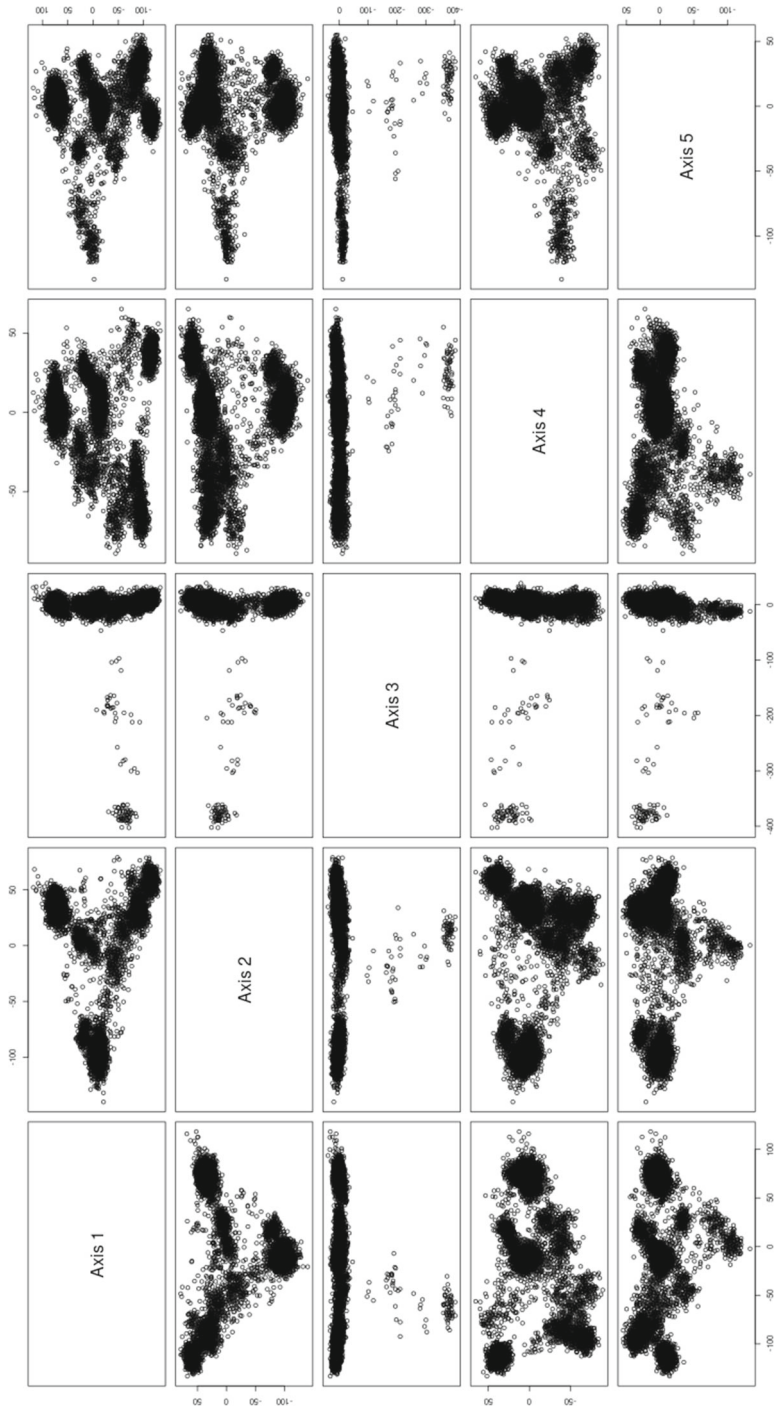


Fig. 5 Axes 1–5 of the standard MDS on HIV data

distant from the bulk of the rest of the data. Notably, the scales of axis 3 was larger than for axes 1 and 2, which will be further considered in the discussion below.

5 Discussion

The present work builds on a previous study aiming at performing MDS with very large data sets (Paradis 2018). However, the procedure presented in this previous work was limited to find a space limited to a one or two dimensions. By contrast, the method proposed in the present paper is not limited to these numbers, and it is up to the user to select the number of dimensions of the final MDS space based on the eigenvalues of the reduced sample. In addition, in Paradis (2018) the coordinates of the observations outside of the reduced sample were found by numerical optimization, whereas direct matrix calculus is used in the present work. The most computationally costly operation seems to be the matrix inverse $(Z^T Z)^{-1}$, but since Z has m rows and m columns and that in practice it is likely that $m \leq 1000$, this is expected to take typically a short time (i.e., a few seconds). Furthermore, this matrix inverse needs to be computed only once in order to apply Eq. 3 to the $n - m$ observations out of the reduced sample.

When compared to standard (i.e., full) MDS, the method proposed here (RMDS) requires to compute far less pairwise distances among observations. The standard MDS needs to compute all $(n^2 - n)/2$ pairwise distances and store them in an $n \times n$ matrix before decomposition. The number of distances computed is thus proportional to n^2 and the storage requirement is typically $8n^2$ bytes if the distances are stored as 64-bit floating-point reals. An additional matrix of the same size with the eigenvectors output by the eigendecomposition of the previous matrix is eventually also needed. By contrast, the RMDS has the same resource requirements for only the m observations in the reduced sample. In addition, the procedure computes $m(n - m)$ distances in order to apply Eq. 3. Thus, a total of $(m^2 - m)/2 + m(n - m)$ distances need to be computed by the RMDS method. For example, if $n = 10^4$ and $m = 200$ this number will be 1,979,900 whereas the number of distances computed for a standard MDS will be 49,995,000, differing by a factor of 25. Furthermore, the number of distances required for the RMDS grows linearly with n , so this factor will grow when n increases.

The crucial feature of the RMDS is the way the m observations of the reduced sample are selected. One goal of this paper was to compare two algorithms performing this selection step. Algorithm 1, which is a modified version of an algorithm presented in Paradis (2018), aims to select the most distant observations as defined by pairwise distances. The second algorithm considered in this work simply selects randomly m observations out of n . Algorithm 1 will tend to select outlying observations even if they are in low proportions in the data set and thus such observations may likely be ignored by random sampling. This is illustrated by the above second simulation experiment where two groups were simulated: if the number of observations in the smaller group is very low, they were very likely “forgotten” by random sampling and Algorithm 1 performed better in this case.

The analysis of the HIV data shows that the comparison of the results from both selection algorithms is also informative in practice. With Algorithm 1, the projection on axes 1 and 2 revealed one main cluster, a second small cluster, and a few outliers.

On the other hand, random sampling showed five main clusters on these first two axes, in agreement with the standard MDS on the full data set. However, the axis 3 of these two analyses revealed a smaller cluster and a few outliers. When looking at the scales on these three axes, it appears that the range of values on axis 3 of the standard MDS is similar to the range of values on axis 1 of the RMDS. Thus, this suggests that Algorithm 1 worked as expected in selecting the most distant observations. Interestingly, these outliers were not apparent in the first two axes of the standard MDS because they did not contribute to the overall variability within the data. Therefore, the comparison of both selection algorithms helps to reveal these patterns in the data. It is worth adding that running the RMDS, with either selection algorithm, took less than one minute, while the full MDS took more than one hour, so that it is straightforward to repeat the RMDS, or even plan to apply it to larger data sets. For instance, the RMDS applied to a data set with one million sequences is expected to take less than two hours, whereas the full MDS is predicted to take 14 months (in addition to require a computer able to store and handle two $10^6 \times 10^6$ matrices to store the distances and their eigendecomposition).

Statistical or computational methods aimed at displaying large data sets in a limited number of dimensions have received a lot of attention by researchers. Recently, uniform manifold approximation and projection (UMAP) was proposed to handle highly dimensional data with many observations (McInnes et al. 2018). UMAP was applied to analyse gene expression data (Becht et al. 2019) or remote sensing data (Franch et al. 2019), two types of applications where highly-dimensional data are common (Stephens et al. 2015). UMAP is an appropriate method when discrete clusters exist in the data. On the other hand, if the distribution of observations is continuous (i.e., no clusters exist), this method will tend to create artificial clusters, especially if the number of variables is low. In this context, methods such as PCA or MDS have the advantage of not assuming any structure *a priori*, so that structures may become apparent once dimension reduction has been performed. Furthermore, Sun et al. (2019) reviewed eighteen methods applied to single-cell sequencing data and found that PCA and MDS have overall good features compared to more recently developed methods. Interestingly, these authors found that only three out of the eighteen methods they reviewed performed well for “rare cell detection” (PCA and MDS were rated “Intermediate” on this criterion). It might be interesting to assess whether RMDS with Algorithm 1 is a good alternative for this.

The idea to take a random, reduced sample from a large data set and use it to perform preliminary analyses as a first or preliminary step, and then “map” or “project” the whole data set based on the results from this reduced sample has been explored several times in the literature. Mirarab et al. (2015) developed a method to align very large samples of molecular sequences using what they call a “backbone” alignment made from a random subset of sequences. More recently, Wan et al. (2020) proposed the SHARP method for the analysis of single-cell RNA-sequencing data based on random projection of data blocks followed by a weighted metaclustering, and then a similarity-based metaclustering. They showed their method to perform much faster than alternatives and being able to scale with several millions of cells. However, the SHARP method assumes the existence of groups of clusters in the data, thus avoiding the need to consider a continuous space like in traditional MDS.

The possibility to display many kinds of data in a limited number of dimensions is an attractive feature of MDS. It makes possible to represent observations in a scatter plot and to assess potential structures in the data. Therefore, the method presented in this paper has a very large potential range of applications.

6 Supplementary Material

The code used in this paper is available on GitHub: <https://github.com/emmanuelparadis/rmds>.

Acknowledgements I am grateful to two anonymous reviewers for their constructive comments on a previous version of this article. This is publication ISEM 2021-118.

References

- Abraham G, Inouye M (2014) Fast principal component analysis of large-scale genome-wide data. *PLoS ONE* 9(4):e93766. <https://doi.org/10.1371/journal.pone.0093766>
- Baglama J, Lothar R (2005) Augmented implicitly restarted Lanczos bidiagonalization methods. *SIAM J Sci Comput* 27(1):19–42
- Baglama J, Reichel L, Lewis BW (2019) irlba: fast truncated singular value decomposition and principal components analysis for large dense and sparse matrices. <https://CRAN.R-project.org/package=irlba>, R package version 2.3.3
- Becht E, McInnes L, Healy J, Dutertre CA, Kwok IWH, Ng LG, Ginhoux F, Newell EW (2019) Dimensionality reduction for visualizing single-cell data using UMAP. *Nat Biotechnol* 37:38–44. <https://doi.org/10.1038/nbt.4314>
- Beugin MP, Gayet T, Pontier D, Devillard S, Jombart T (2018) A fast likelihood solution to the genetic clustering problem. *Methods Ecol Evol* 9(4):1006–1016. <https://doi.org/10.1111/2041-210X.12968>
- Degras D, Cardot H (2016) Online principal component analysis. <https://CRAN.R-project.org/package=onlinePCA>, r package version 1.3.1
- D’Enza AI, Markos A, Buttarazzi D (2018) The idm package: incremental decomposition methods in R. *J Stat Softw Code Snippets* 86(4):1–24. <https://doi.org/10.18637/jss.v086.c04>
- Erichson NB, Voronin S, Brunton SL, Kutz JN (2019) Randomized matrix decompositions using R. *J Stat Softw* 89(11):1–48. <https://doi.org/10.18637/jss.v089.i11>
- Franch G, Jurman G, Coviello L, Pendesini M, Furlanello C (2019) MASS-UMAP: fast and accurate analog ensemble search in weather radar archives. *Remote Sens* 11(24):2922. <https://doi.org/10.3390/rs11242922>
- Gower JC (1968) Adding a point to vector diagrams in multivariate analysis. *Biometrika* 55(3):582–585
- Halko N, Martinsson PG, Tropp JA (2011) Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev* 53(2):217–288. <https://doi.org/10.1137/090771806>
- Kruskal JB (1964) Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika* 29(1):1–27
- Lloyd SP (1982) Least squares quantization in PCM. *IEEE Trans Inf Theory* 28(2):129–137. <https://doi.org/10.1109/TIT.1982.1056489>
- McInnes L, Healy J, Saul N, Großberger L (2018) UMAP: uniform manifold approximation and projection. *J Open Sour Softw* 3:861. <https://doi.org/10.21105/joss.00861>
- Mirarab S, Nguyen N, Guo S, Wang LS, Kim J, Warnow T (2015) PASTA: ultra-large multiple sequence alignment for nucleotide and amino-acid sequences. *J Comput Biol* 22(5):377–386. <https://doi.org/10.1089/cmb.2014.0156>
- Paradis E (2018) Multidimensional scaling with very large data sets. *J Comput Gr Stat* 27(4):935–939. <https://doi.org/10.1080/10618600.2018.1470001>
- Paradis E (2020) *Population genomics with R*. Chapman & Hall, Boca Raton, FL

- Paradis E, Schliep K (2019) ape 5.0: an environment for modern phylogenetics and evolutionary analyses in R. *Bioinformatics* 35(3):526–528. <https://doi.org/10.1093/bioinformatics/bty633>
- Qiu Y, Mei J (2019) RSpectra: solvers for large-scale eigenvalue and SVD problems. <https://CRAN.R-project.org/package=RSpectra>, r package version 0.16-0
- R Core Team (2021) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. <http://www.R-project.org>
- Roweis S (1998) EM algorithms for PCA and SPCA. In: *Neural Information Processing Systems 10 (NIPS'97)*, pp 626–632
- Stephens ZD, Lee SY, Faghri F, Campbell RH, Zhai C, Efron MJ, Iyer R, Schatz MC, Sinha S, Robinson GE (2015) Big data: astronomical or genetical? *PLoS Biol* 13(7):e1002195
- Sun S, Zhu J, Ma Y, Zhou X (2019) Accuracy, robustness and scalability of dimensionality reduction methods for single-cell RNA-seq analysis. *Genome Biol* 20:269. <https://doi.org/10.1186/s13059-019-1898-6>
- Venables WN, Ripley BD (2002) *Modern applied statistics with S*, 4th edn. Springer, New York
- Wan S, Kim J, Won KJ (2020) SHARP: hyperfast and accurate processing of single-cell RNA-seq via ensemble random projection. *Genome Res* 30:205–213. <https://doi.org/10.1101/gr.254557.119>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.