

Fast Multiscale Neighbor Embedding

Cyril de Bodt, Dounia Mulders, Michel Verleysen, *Fellow, IEEE*, and John A. Lee

Abstract—Dimension reduction (DR) computes faithful low-dimensional (LD) representations of high-dimensional (HD) data. Outstanding performances are achieved by recent neighbor embedding (NE) algorithms like t -SNE, which mitigate the curse of dimensionality. The single- or multi-scale nature of NE schemes drives the HD neighborhood preservation in the LD space. Whereas single-scale methods focus on single-sized neighborhoods through the concept of perplexity, multi-scale ones preserve neighborhoods in a broader range of sizes and account for the global HD organization to define the LD space. For both single- and multi-scale methods however, their time complexity in the number of samples is unaffordable for big data sets. Single-scale methods can be accelerated by relying on the inherent sparsity of the HD similarities they involve. On the other hand, the dense structure of the multi-scale HD similarities prevents developing fast multi-scale schemes in a similar way. This paper addresses this difficulty by designing randomized accelerations of the multi-scale methods. To account for all levels of interactions, the HD data are first subsampled at different scales, enabling to identify small and relevant neighbor sets for each data point thanks to vantage-point trees. Afterwards, these sets are employed with a Barnes-Hut algorithm to cheaply evaluate the considered cost function and its gradient, enabling large-scale use of multi-scale NE schemes. Extensive experiments demonstrate that the proposed accelerations are, statistically significantly, both faster than the original multi-scale methods by orders of magnitude, and better preserving the HD neighborhoods than state-of-the-art single-scale schemes, leading to high-quality LD embeddings. Public codes are freely available (<https://github.com/cdebodt>).

Index Terms—Dimensionality reduction, neighbor embedding, t -SNE, multi-scale methods, vantage-point trees, Barnes-Hut.

I. INTRODUCTION

DIMENSIONALITY reduction (DR) represents high-dimensional (HD) data sets into low-dimensional (LD) spaces, mostly for exploratory visualization or to dismiss the curse of dimensionality [1]. This curse encompasses the inherent difficulties to cope with HD data and motivated the development of adequate approaches to extract meaningful LD features [2], [3]. In visualization context, the relevance of a LD embedding is typically related to the HD neighborhood preservation. Mappings from HD to LD coordinates [1] formalize this neighborhood preservation principle in the context of paradigms such as the reproduction of distances [4] or neighborhoods [5], [6]. Linear projections of the HD vectors include early principal component analysis (PCA) [7] and classical metric multidimensional scaling (MDS) [4], driven by the variance or dot product preservation. Nonlinear metric MDS extensions [8] define (weighted) distance preservation schemes relying on either Euclidean or approximated geodesic

measures [9]. Affinity matrices may also be computed to guide the LD embedding tuning [10], [11]. However, these approaches are hardly superior to the older methods in visualization tasks [1], [12], [13], potentially because they can be expressed as classical MDS applied in an unknown feature space [14]. Distance-preserving schemes are particularly affected by the norm concentration phenomenon [15], which leads pairwise distances to become more and more similar as the dimension increases [16]. Meanwhile, neighbor embedding (NE) techniques as Stochastic Neighbor Embedding (SNE) [6] and variants [13], [17] alleviate this phenomenon by matching neighbor probability distributions defined in both spaces to compute the LD points [16]. The resulting outstanding performances motivated developing alternative SNE-based models, with heavy-tailed distributions as in t -SNE [13], [18], [19], divergence mixtures as cost functions [17], [20], [21], missing data management [22], enhanced optimization [23]–[26], etc.

Originally, NE methods required the user to specify some perplexity for the neighborhood probability distributions, resulting in a preservation oriented toward neighborhoods of the corresponding size. These methods may hence be viewed as single-scale, with fixed resolution. More recently, multi-scale NE approaches were introduced by combining neighborhood probabilities tuned at various data scales [27], [28], leading to meta-parameter-free techniques. Compared to their single-scale counterparts, the multi-scale schemes substantially better preserve neighborhoods of all sizes thanks to their ability to consider both local and global structures while computing the LD embedding. This enables for instance to reproduce small neighborhoods as well as or better than t -SNE, while considerably improving the preservation of larger ones [27].

Despite their impressive results, single- and multi-scale NE methods respectively have $\mathcal{O}(N^2)$ and $\mathcal{O}(N^2 \log N)$ time complexity when employed with N data samples. Although the logarithmic factor further induced by the multi-scale behavior has a minor impact, the quadratic dependency is not suited to big data sets and confines the use of NE methods to a few thousands data points. An obvious workaround is to employ a NE method solely on a small random subset of the database, enabling its visualization for exploratory analysis. However, analyzing a reduced part of a data set often only reveals a limited fraction of the HD data relationships [29]. Another option is to use the HD and LD positions of the selected data samples to project the remaining data points after learning a parametric function between the HD and LD spaces [30], [31]. Nevertheless, learning such mappings is typically difficult since the NE method only processes a small part of the database [32], hence ignoring the information carried by most of the data points on the underlying HD structures [13]. Some

CdB and DM are Research Fellows of the Fonds de la Recherche Scientifique-FNRS. JAL is a Senior Research Associate with the FNRS. CdB, DM, MV and JAL are with the ICTEAM at UCLouvain, Belgium (e-mail: {cyril.debodt; dounia.mulders; michel.verleysen; john.lee}@uclouvain.be).

Please cite as: C. de Bodt, D. Mulders, M. Verleysen, and J. A. Lee, “Fast Multiscale Neighbor Embedding,” *IEEE Transactions on Neural Networks and Learning Systems*, 2020, DOI: 10.1109/TNNLS.2020.3042807.

Article abstract in IEEE Xplore: <https://ieeexplore.ieee.org/document/9308987>.

2162-237X © 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.

See <https://www.ieee.org/publications/rights/index.html> for more information.

studies experimentally show that applying NE methods on a subset of a database poorly generalizes as it largely neglects the HD data interactions [33], affecting the projection quality of the remaining samples and altering the structure of the data. Considering big data sets is further demonstrated to always be advantageous to tune visualizations, regardless of whether the whole database or only a portion needs to be analyzed [33]. Therefore, some works attempt to incorporate information from the entire data structure when visualizing a part of the samples, such as landmark t -SNE [13]. However, while preventing to observe the whole database, this approach entails considerable running times and memory requirements [33].

For these reasons, lower-complexity approximations were introduced for single-scale methods, either relying on tree-based computations [32], [33], fast multipole methods (FMMs) [34], negative sampling techniques (NSTs) [35] or Gaussian mixture modeling approximations [29]. Although they effectively reduce computation times by reaching near-linear time complexity with respect to the number of data samples, Gaussian mixtures are typically difficult to tune in (very) HD spaces [36] and require to determine their number of components. A poor tuning or an inadequate number of mixture components deteriorates the quality of the HD modeling and hence of the LD embedding. On the other hand, tree-based schemes, FMMs and NSTs achieve impressive performances in terms of both quality and speedup, ensuing in methods with $\mathcal{O}(N \log N)$ and $\mathcal{O}(N)$ time complexity provided that the HD neighbor probabilities can be sparsely encoded. The tree-based algorithms and FMMs poorly scale with the LD dimension, but the latter is typically as small as 2 or 3 for visualization purposes. However, the sparsity of the HD neighbor probability distributions is a key ingredient of tree-based methods, FMMs and NSTs. This fits the single-scale framework well, in which the neighbor probability distributions remain local around the data points, but not the multi-scale one. Indeed, multi-scale neighbor probability distributions span much larger levels of data organization to account for both local and global neighboring structures when tuning the LD embedding. This defines dense neighbor probability distributions, preventing to accelerate multi-scale NE methods as the single-scale ones.

This paper develops fast multi-scale NE schemes with $\mathcal{O}(N \log^2 N)$ time complexity instead of $\mathcal{O}(N^2 \log N)$, able to manage large-scale databases with hundreds of thousands data points for visualization. Appropriate neighbor sets are first computed for all samples using vantage-point trees defined on random subsamplings of the data set at all scales. This enables preserving the multi-scale behavior of the methods by accounting for global data structures in the obtained neighbor sets. The cost function of the considered multi-scale scheme is then minimized using these sets with a Barnes-Hut (BH) algorithm to efficiently approximate the LD data interactions and avoid LD crushing phenomena, i.e., overlapping LD points. Simulations on numerous data sets evidence that the proposed fast multi-scale methods outperform state-of-the-art single-scale approaches such as t -SNE in terms of LD embedding quality quantified using neighborhood preservation criteria, while at the same time being considerably faster than non-accelerated NE algorithms. These results are statistically significant with

respect to the random subsamplings of the data. Free codes are publicly available online (<https://github.com/cdebodt>), enabling to easily employ the presented algorithms.

This paper is organized as follows: Section II reviews neighbor embedding methods in both single-scale (Section II-A) and multi-scale (Section II-B) settings. Section III first describes lower-complexity approximations for fast single-scale schemes (Section III-A), and then introduces the proposed randomized multi-scale accelerations (Section III-B). Section IV details experimental results comparing the performances of (single-scale) t -SNE, multi-scale SNE and multi-scale t -SNE with their respective fast versions on several data sets, in terms of both computation time and DR quality, for varying values of the BH algorithm meta-parameter. The accelerated methods are also validated on large databases on which it is unaffordable to employ the original, non-fast NE algorithms. Section V draws final conclusions.

II. NEIGHBOR EMBEDDING

DR aims to map HD data sets in LD spaces by best preserving their structure. The data points HD neighborhoods should hence be well reflected by the LD ones. In this view, rigid distance preservation methods are outperformed by the successful neighbor embedding approaches [1]. They are based on the concept of neighbor probabilities, first introduced by SNE [6] and refined in more recent multi-scale techniques, yielding impressive performances [27]. SNE and its t -SNE variant are first reviewed in Section II-A. Their multi-scale extensions are then presented in Section II-B.

A. Single-scale methods

Given a set $\Xi = \{\xi_i\}_{i=1}^N$ of N points lying in a M -dimensional space (HDS), let $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ refer to its representation in a P -dimensional space, termed as the LD space (LDS), with $P \leq M$. The distances between the i^{th} and j^{th} points are denoted by δ_{ij} and d_{ij} in the HDS and LDS respectively. Single-scale pairwise HD and LD similarities are defined by SNE for each $i \in \mathcal{I} := \{1, \dots, N\}$ and $j \in \mathcal{I} \setminus \{i\}$,

$$\sigma_{ij*} = \frac{\exp(-\pi_{i*} \delta_{ij}^2 / 2)}{\sum_{k \in \mathcal{I} \setminus \{i\}} \exp(-\pi_{i*} \delta_{ik}^2 / 2)}, \quad \sigma_{ii*} = 0, \quad (1)$$

$$s_{ij*} = \frac{\exp(-p_{i*} d_{ij}^2 / 2)}{\sum_{k \in \mathcal{I} \setminus \{i\}} \exp(-p_{i*} d_{ik}^2 / 2)}, \quad s_{ii*} = 0. \quad (2)$$

A binary search sets the precision π_{i*} to obtain a user-provided perplexity K_* for the distribution $\sigma_{i*} = \{\sigma_{ij*}, j \in \mathcal{I} \setminus \{i\}\}$,

$$\pi_{i*} \text{ s.t. } \log K_* = - \sum_{j \in \mathcal{I} \setminus \{i\}} \sigma_{ij*} \log \sigma_{ij*}. \quad (3)$$

In practice, only a finite maximum number of iterations is tolerated during the binary search, to avoid infinite loops when it is impossible to tune π_{i*} in order to obtain a perplexity K_* , for instance when the HD distances in $\{\delta_{ij}, j \in \mathcal{I} \setminus \{i\}\}$ are all equal. The LD precisions p_{i*} are fixed to 1. The soft Gaussian neighborhood size is quantified by the perplexity, meaning that (1) and (2) can be interpreted as probabilities for j of being a neighbor of i in the HDS and LDS. The star index

in K_* is for consistency with Section II-B. To compute $\mathbf{X}$, the sum over $i \in \mathcal{I}$ of the Kullback-Leibler (KL) divergences between the distributions σ_{i*} and $s_{i*} = \{s_{ij*}, j \in \mathcal{I} \setminus \{i\}\}$,

$$C_* = \sum_{i \in \mathcal{I}, j \in \mathcal{I} \setminus \{i\}} \sigma_{ij*} \log(\sigma_{ij*} / s_{ij*}), \quad (4)$$

is minimized by SNE through gradient-based optimization [6].

Besides symmetrizing the similarities, t -SNE uses a Student t -distribution with one degree of freedom in the LDS, circumventing the crowding problem [13], which refers to the difficulty to reproduce data similarities from large HD volumes to exponentially smaller LD ones: for $i \in \mathcal{I}$, $j \in \mathcal{I} \setminus \{i\}$,

$$\tau_{ij*} = (\sigma_{ij*} + \sigma_{ji*}) / (2N), \quad \tau_{ii*} = 0, \quad (5)$$

$$t_{ij} = \frac{(1 + d_{ij}^2)^{-1}}{\sum_{k \in \mathcal{I}, l \in \mathcal{I} \setminus \{k\}} (1 + d_{kl}^2)^{-1}}, \quad t_{ii} = 0. \quad (6)$$

The t -SNE cost function remains as in SNE:

$$C_{t*} = \sum_{i \in \mathcal{I}, j \in \mathcal{I} \setminus \{i\}} \tau_{ij*} \log(\tau_{ij*} / t_{ij}). \quad (7)$$

Other variants of SNE were defined by modifying its cost function using mixtures of divergences [20], such as the Neighborhood Retrieval Visualizer (NeRV) [17] and Jensen-Shannon embedding (JSE) [21]. For all these SNE-based methods, evaluating the cost function or its gradient scales in $\mathcal{O}(N^2)$, since it requires computing all similarities.

B. Multi-scale methods

Small and arbitrary perplexities K_* are typically used in SNE, inducing local HD distance focus in (1), mainly preserving small neighborhoods. Indeed, most HD similarities are infinitesimal, hiding the associated LD similarities to the SNE cost function. Such local distance transformations hence neglect the mid- and large-scale interactions when designing the embedding global structure. The long-range neighborhood preservation is therefore considerably affected. On the other hand, nearly-uniform HD similarities result from too large perplexities, harming the small neighborhood preservation.

However, the multi-scale SNE method [27] combines various SNE similarities relying on increasing perplexities, to model both local and global trends and to retrieve several data scales simultaneously. Single-scale similarities are now indexed using a scale counter h , for $i \in \mathcal{I}$ and $j \in \mathcal{I} \setminus \{i\}$, as

$$\sigma_{ijh} = \frac{\exp(-\pi_{ih} \delta_{ij}^2 / 2)}{\sum_{k \in \mathcal{I} \setminus \{i\}} \exp(-\pi_{ih} \delta_{ik}^2 / 2)}, \quad \sigma_{iih} = 0, \quad (8)$$

$$s_{ijh} = \frac{\exp(-p_{ih} d_{ij}^2 / 2)}{\sum_{k \in \mathcal{I} \setminus \{i\}} \exp(-p_{ih} d_{ik}^2 / 2)}, \quad s_{iih} = 0. \quad (9)$$

Multi-scale similarities average the single-scale ones as

$$\sigma_{ij} = H^{-1} \sum_{h=1}^H \sigma_{ijh}, \quad s_{ij} = H^{-1} \sum_{h=1}^H s_{ijh}, \quad (10)$$

where the number of scales is denoted by H . One can set precisions π_{ih} as in SNE using perplexities summarizing the whole data range, selected in a data-driven way without any user intervention. Indexed perplexities are introduced as $K_h = 2^{h-1} K_1$ [27], with a small base perplexity such as

$K_1 = 2$ and $1 \leq h \leq H = \lceil \log_2(N/K_1) \rceil$, where $\lceil \cdot \rceil$ represents rounding. The entropy of the HD similarity distributions hence linearly increases across scales. The LD precisions $p_{ih} = p_h = K_h^{-2U/P}$ follow the exponential growth of the perplexities, with U fixed from the HD precisions as in [27].

Multi-scale SNE keeps the SNE cost function, developing as

$$C = - \sum_{i \in \mathcal{I}, j \in \mathcal{I} \setminus \{i\}} \sigma_{ij} \log s_{ij}, \quad (11)$$

since the entropy of the HD similarities does not depend on $\mathbf{X}$. On the other hand, multi-scale t -SNE [28] minimizes

$$C_t = \sum_{i \in \mathcal{I}, j \in \mathcal{I} \setminus \{i\}} \tau_{ij} \log \frac{1}{t_{ij}}, \quad \text{with } \tau_{ij} = \frac{\sigma_{ij} + \sigma_{ji}}{2N}. \quad (12)$$

Multi-scale NeRV and JSE can also be defined [27], by replacing their single-scale similarities with multi-scale ones.

The optimization of SNE-like methods turns out to be tough [6], [17], [23], [26], in particular when small perplexities are considered since they induce high-frequency oscillations in the cost functions [27]. Heuristics to attempt avoiding shallow local minima are usually helpful and include early exaggeration [13] and multi-scale optimization [27], which starts the minimization with the largest scale and then progressively introduces the smaller ones.

The $\mathcal{O}(\log N)$ scales entail a $\mathcal{O}(N^2 \log N)$ time complexity for the multi-scale NE schemes. Whereas single-scale algorithms best reproduce neighborhood sizes close to or smaller than K_* but poorly preserve larger ones, broader ranges of neighborhoods are retrieved by the multi-scale methods. This enables retaining both local and global data structures [27], while relieving the user from setting parameter K_* .

III. ACCELERATING NEIGHBOR EMBEDDING

The quadratic time complexity of both single- and multi-scale NE schemes with respect to N limits their applicability to moderate-sized data sets. This motivated accelerating single-scale methods, by harnessing the typical smallness of the user-defined perplexity K_* with respect to N . This observation enables defining sparse HD similarities in $\mathcal{O}(N \log N)$ time. The cost functions of single-scale approaches and their gradients can then be estimated using tree-based algorithms [32], [33] or fast multipole methods [34] in $\mathcal{O}(N \log N)$ and $\mathcal{O}(N)$ time respectively. On the other hand, the multi-scale framework relies on perplexities covering all data scales. The induced dense HD similarities hinder designing lower-complexity multi-scale techniques as in the single-scale case. Section III-A focuses on tree-based approximations for SNE and t -SNE, while Section III-B details the proposed randomized acceleration of multi-scale methods, with $\mathcal{O}(N \log^2 N)$ time complexity. In both Sections III-A and III-B, Euclidean LD distances are assumed to compute the gradients of the cost functions.

A. Fast single-scale methods

Due to its normalized Gaussian nature, σ_{ij*} vanishes as δ_{ij} grows [33]. A sparse approximation of σ_{ij*} then consists in

$$\tilde{\sigma}_{ij*} = \begin{cases} \frac{\exp(-\tilde{\pi}_{i*} \delta_{ij}^2 / 2)}{\sum_{k \in \mathcal{I}_{i*}} \exp(-\tilde{\pi}_{i*} \delta_{ik}^2 / 2)} & \text{if } j \in \mathcal{I}_{i*} \\ 0 & \text{otherwise} \end{cases}, \quad (13)$$

with

$$\tilde{\pi}_{i*} \text{ s.t. } \log K_* = - \sum_{j \in \mathcal{I}_{i*}} \tilde{\sigma}_{ij*} \log \tilde{\sigma}_{ij*}, \quad (14)$$

where $\mathcal{I}_{i*}$ indexes the $\lfloor 3K_* \rfloor$ nearest neighbors of ξ_i in $\Xi \setminus \{\xi_i\}$ [32]. The number of nearest neighbors of ξ_i to consider should indeed increase with K_* , since the entropy of σ_{i*} at the right-hand side of (3) is monotonically decreasing with π_{i*} when the distances in $\{\delta_{ij}, j \in \mathcal{I} \setminus \{i\}\}$ are not all equal [25]. Therefore, the greater K_* , the smaller π_{i*} , the flatter the HD similarities in σ_{i*} and the more nearest neighbors should be accounted for in (13) for the faithfulness of the approximation. The $\lfloor 3K_* \rfloor$ number is suggested in [32] as it yields impressive DR performances, and is employed in this work to ease the comparison with state-of-the-art literature results. The nearest neighbor sets are exactly derived by creating a vantage-point tree [37] on Ξ and performing N depth-first searches, each yielding the $\lfloor 3K_* \rfloor$ nearest neighbors of a datum. A sparse estimate for τ_{ij*} then develops as $\tilde{\tau}_{ij*} = (\tilde{\sigma}_{ij*} + \tilde{\sigma}_{ji*}) / (2N)$. Both matrices $[\tilde{\sigma}_{ij*}]_{i,j \in \mathcal{I}}$ and $[\tilde{\tau}_{ij*}]_{i,j \in \mathcal{I}}$ have $\mathcal{O}(K_* N)$ non-zero entries instead of $\mathcal{O}(N^2)$ for their non-sparse versions. The tree is created in $\mathcal{O}(N \log N)$ time and each search costs $\mathcal{O}(K_* \log N)$ time, hence providing $\mathcal{I}_{i*}$ for all $i \in \mathcal{I}$ in $\mathcal{O}(K_* N \log N)$ time.

Efficiently evaluating the SNE and t -SNE cost functions and their gradients still entails cheaply estimating their remaining non-sparse, LD terms. A Barnes-Hut (BH) algorithm [38] can be developed for this purpose, leading to BH SNE [6], [33] and BH t -SNE [32]. The computations are made explicit for the gradient of t -SNE cost function in Appendix A, and can be adapted to the SNE case and to estimate the cost functions. Figure 1 sketches the main principles of the BH algorithm. When the LD dimension P is set to 2, a quadtree is first created on the LD embedding in $\mathcal{O}(N)$ time [32], as illustrated in Fig. 1(a). An octree may similarly be constructed if $P = 3$, but the exponential increase of the number of cells limits the suitability of such space-partitioning trees for larger LD dimensions. Then, instead of evaluating exactly all pairwise interactions between the LD points, which is depicted in Fig. 1(b), they are approximated for faraway data samples using the centers-of-mass of the LD coordinates in the cells of the quadtree [33], as outlined in Fig. 1(c). While conducting exact computations as in Fig. 1(b) leads to $\mathcal{O}(N^2)$ time complexity when performed on the entire data set, the BH estimation scheme of Fig. 1(c) runs in $\mathcal{O}(N \log N)$ average time [32]. A threshold hyper-parameter $\theta \in [0, 1]$ determines when to launch the approximation while traversing the tree, and thus balances the speed and the accuracy of the evaluation. Larger θ speeds up computations with rougher approximations. The DR quality of the t -SNE acceleration remains, however, stable under variations of θ [39].

Assuming that K_* is small compared to N , approximations of C_{t*} and its gradient are then obtained in $\mathcal{O}(N \log N)$ average time [32], and can be used in gradient-based minimization schemes to tune the LD embeddings of BH t -SNE.

B. Fast multi-scale methods

A crucial aspect of fast single-scale methods stems from their sparse HD similarities. These rely on nearest neighbor

sets $\mathcal{I}_{i*}$ for each $i \in \mathcal{I}$, with $|\mathcal{I}_{i*}| = \lfloor 3K_* \rfloor$. A vantage-point tree [37] computes these sets in $\mathcal{O}(K_* N \log N)$ time, which reduces to $\mathcal{O}(N \log N)$ if $K_* \ll N$.

Multi-scale methods combine single-scale similarities tuned at several data scales, employing exponentially growing perplexities. In particular, K_H lies in the $\mathcal{O}(N)$ range, leading to neighbor sets $\mathcal{I}_{iH}$ with $\mathcal{O}(N)$ size for each $i \in \mathcal{I}$, since $|\mathcal{I}_{iH}| = \min(\lfloor 3K_H \rfloor, N - 1)$. As multi-scale HD similarities average the single-scale ones in (10), full multi-scale neighbor sets are derived as $\mathcal{I}_i = \cup_{h=1}^H \mathcal{I}_{ih} = \mathcal{I} \setminus \{i\}$, to gather the neighbors from all scales. Therefore, the multi-scale HD similarity matrix determined by these neighbor sets has $\mathcal{O}(N^2)$ non-zero entries. This prevents designing sparse multi-scale HD similarities as in the single-scale framework.

Single-scale HD similarities with small perplexities excel at modeling local interactions around each data point, while reducing the influence of farther away HD objects. They indeed concentrate their probability mass on the nearest neighbors of each datum, which can hence in turn serve as a remarkable basis to produce sparse approximations. As local structures often rapidly fluctuate, it is in fact crucial to consider the closest neighbors to guarantee the faithfulness of the sparse estimates when processing small scales. On the other hand, single-scale similarities based on large perplexities capture global, large scale interactions in the data. As these global structures emerge from average trends across important amounts of data points, it is little convincing that their estimation requires exhaustive consideration of all samples. Indeed, whereas local interactions are short range and induced by very specific sets of neighbors, global ones result from overall consensuses which can be fairly well approximated using appropriate subsets of data points. In the cost functions of neighbor embedding methods, evaluating the divergences between probability distributions defined on such subsets may reveal sufficient to relevantly shape the LD embedding according to global HD properties.

However, multi-scale similarities combine both types of small and large perplexities, respectively focusing on local and global HD characteristics. As a consequence, to faithfully accelerate multi-scale methods, the design of small and relevant multi-scale neighbor sets, denoted by $\tilde{\mathcal{I}}_i$ for $i \in \mathcal{I}$, entails purposely gathering data points which reasonably model the whole data cloud, while at the same time accurately target the nearest HD neighbors of each ξ_i . The computation of NE cost functions could then be restricted to these subsets $\tilde{\mathcal{I}}_i$ for each $i \in \mathcal{I}$, which would efficiently score the adequacy of the LD coordinates $\mathbf{x}_i$ in the multi-scale spirit, by accounting for both global trends and local interactions in the data.

The hierarchical structure of the multi-scale HD similarities, from small to large perplexities, suggests that the set $\tilde{\mathcal{I}}_i$ should gather data points lying at different ranges of ξ_i . Indeed, as small scales require fine levels of details, $\tilde{\mathcal{I}}_i$ must precisely contain the very close neighbors of ξ_i . But as the scale index increases, only rougher HD characterization is required and a few farther data points from ξ_i can be sampled, providing a higher resolution view. Inspired by this intuition, one can first define a hierarchy $\{\Xi_h\}_{h=1}^H$ of subsampled HD data sets at each scale, where Ξ_h is a random sample of Ξ with $\lfloor 2^{1-h} N \rfloor$ elements drawn without replacement. Note that $\Xi_1 = \Xi$. This

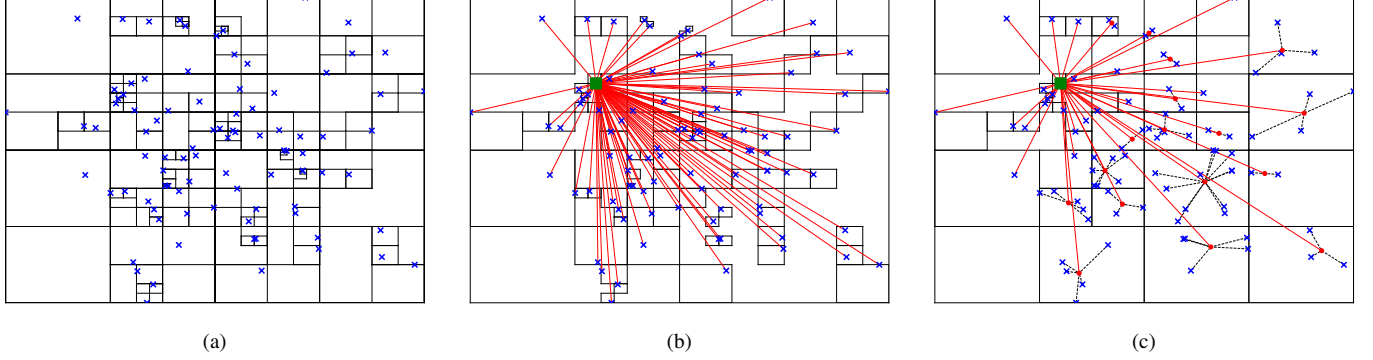


Fig. 1. (a) Quadtree of a 2-D data set with crosses as data points. (b) Complete pairwise interactions between the square data point and the remaining crosses. (c) Barnes-Hut approximation of the complete pairwise interactions with the square data point, induced by some threshold θ . The centers-of-mass employed for the approximation are depicted as dots, linked with dotted lines to the crosses belonging to their cells.

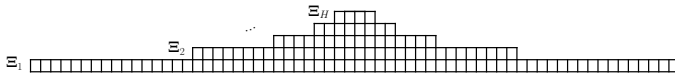


Fig. 2. Hierarchy of HD data sets at all scales. A square refers to a HD sample.

collection of HD data sets is illustrated in Fig. 2.

For each $h \in \{1, \dots, H\}$, a vantage-point tree [37] can be efficiently created on Ξ_h in $\mathcal{O}(2^{1-h}N \log(2^{1-h}N))$ time. All the trees are then defined in

$$\mathcal{O}\left(\sum_{h=1}^H \frac{N}{2^{h-1}} \log \frac{N}{2^{h-1}}\right) = \mathcal{O}\left(N \log N \sum_{h=1}^H \frac{1}{2^h}\right) \quad (15)$$

$$= \mathcal{O}(N \log N) \quad (16)$$

time. The h^{th} tree enables computing the set $\tilde{\mathcal{I}}_{ih}$ of the $\lfloor 3K_1 \rfloor$ nearest neighbors of ξ_i in $\Xi_h \setminus \{\xi_i\}$ in $\mathcal{O}(K_1 \log(2^{1-h}N))$ time. The sets $\tilde{\mathcal{I}}_{ih}$ for all $i \in \mathcal{I}$ and $h \in \{1, \dots, H\}$ are hence obtained in $\mathcal{O}(N \log^2 N)$ time, as $K_1 = 2 \ll N$ and as there are $\mathcal{O}(\log N)$ scales. It is worth mentioning that, on average, $\tilde{\mathcal{I}}_{ih}$ subsamples the set $\mathcal{I}_{ih}$ of the $\lfloor 3K_h \rfloor$ nearest neighbors of ξ_i in $\Xi \setminus \{\xi_i\}$, as $\tilde{\mathcal{I}}_{ih}$ is based on Ξ_h which contains a fraction $2^{1-h} = K_1/K_h$ of Ξ .

Defining $\tilde{\mathcal{I}}_i := \cup_{h=1}^H \tilde{\mathcal{I}}_{ih}$ then follows the above intuition: the smallest scale is represented by the exact $\lfloor 3K_1 \rfloor$ nearest neighbors of ξ_i in $\Xi \setminus \{\xi_i\}$. But as the scale grows, Ξ_h more and more roughly summarizes Ξ in such a way that the accounted neighbors $\tilde{\mathcal{I}}_{ih}$ are more and more dispersed in the data cloud in the sake of capturing its global properties.

The sets $\tilde{\mathcal{I}}_i$ can be employed to define sparse HD similarities focusing on both local and global data relationships. At scale $h \in \{1, \dots, H\}$, one can establish, for $i, j \in \mathcal{I}$,

$$\tilde{\sigma}_{ijh} = \begin{cases} \frac{\exp(-\tilde{\pi}_{ih}\delta_{ij}^2/2)}{\sum_{k \in \tilde{\mathcal{I}}_i} \exp(-\tilde{\pi}_{ih}\delta_{ik}^2/2)} & \text{if } j \in \tilde{\mathcal{I}}_i \\ 0 & \text{otherwise} \end{cases}, \quad (17)$$

with

$$\tilde{\pi}_{ih} \text{ s.t. } \log K_1 = - \sum_{j \in \tilde{\mathcal{I}}_i} \tilde{\sigma}_{ijh}^\pi \log \tilde{\sigma}_{ijh}^\pi, \quad (18)$$

where

$$\tilde{\sigma}_{ijh}^\pi = \begin{cases} \frac{\exp(-\tilde{\pi}_{ih}\delta_{ij}^2/2)}{\sum_{k \in \tilde{\mathcal{I}}_i} \exp(-\tilde{\pi}_{ih}\delta_{ik}^2/2)} & \text{if } j \in \tilde{\mathcal{I}}_i \\ 0 & \text{otherwise} \end{cases}. \quad (19)$$

Precision $\tilde{\pi}_{ih}$ is fixed in (18) thanks to the neighbors $\tilde{\mathcal{I}}_{ih}$. Sparse multi-scale HD similarities then follow as, for $i, j \in \mathcal{I}$,

$$\tilde{\sigma}_{ij} = H^{-1} \sum_{h=1}^H \tilde{\sigma}_{ijh}, \quad \tilde{\tau}_{ij} = (\tilde{\sigma}_{ij} + \tilde{\sigma}_{ji}) / (2N). \quad (20)$$

Their matrices have $\mathcal{O}(N \log N)$ non-zero entries, all derived in $\mathcal{O}(N \log^2 N)$ total time, as $|\tilde{\mathcal{I}}_i|$ scales as $\mathcal{O}(\log N)$.

Sparse multi-scale LD similarities could be defined in a similar way. To match them with (20), they should rely on the same sets $\tilde{\mathcal{I}}_i$ for $i \in \mathcal{I}$, possibly extending each of them with $\mathcal{O}(\log N)$ elements to capture the local and global properties of the LDS. These $\mathcal{O}(\log N)$ additional elements should be iteratively adapted according to the LD embedding modification after each optimization step. Nevertheless, experiments revealed that minimizing the divergence between (20) and such LD similarities leads to LD crushing phenomena, meaning that LD coordinates are overlapping. Indeed, the size of each set $\tilde{\mathcal{I}}_i$ for $i \in \mathcal{I}$ remains limited to $\mathcal{O}(\log N)$, whether adding the above mentioned $\mathcal{O}(\log N)$ elements or not. Therefore, it is likely that $\tilde{\mathcal{I}}_k \cap \tilde{\mathcal{I}}_l = \emptyset$ for $k, l \in \mathcal{I}$. In this case, LD similarities based on the sets $\tilde{\mathcal{I}}_k$ and $\tilde{\mathcal{I}}_l$ would make the LD coordinates $\mathbf{x}_k$ and $\mathbf{x}_l$ independent one to another. The lack of LD repulsive forces between them hence enables their LD superposition. Extending this reasoning to a potentially very large number of data point pairs leads to crushed LD embeddings, with poor HD neighborhood preservation. On the other hand, LD interactions can be efficiently estimated with inspiration from the single-scale setting. This motivates defining hybrid cost functions for fast versions of multi-scale SNE and t -SNE as

$$\tilde{C} = - \sum_{i \in \mathcal{I}, j \in \mathcal{I} \setminus \{i\}} \tilde{\sigma}_{ij} \log s_{ij} \quad (21)$$

and

$$\tilde{C}_t = - \sum_{i \in \mathcal{I}, j \in \mathcal{I} \setminus \{i\}} \tilde{\tau}_{ij} \log t_{ij}, \quad (22)$$

which quantify the mismatch between the sparse multi-scale HD similarities (20) and the LD ones of genuine multi-scale

TABLE I
TIME COMPLEXITIES OF THE NE SCHEMES FROM SECTIONS II AND III.

DR method	Time complexity	
	HD similarities computation at initialization	Iterations in gradient-based optimization
SNE, t -SNE	$\mathcal{O}(N^2)$	$\mathcal{O}(N^2)$
BH SNE, BH t -SNE	$\mathcal{O}(N \log N)$	$\mathcal{O}(N \log N)$
Multi-scale SNE	$\mathcal{O}(N^2 \log N)$	$\mathcal{O}(N^2 \log N)$
Fast Multi-scale SNE	$\mathcal{O}(N \log^2 N)$	$\mathcal{O}(N \log^2 N)$
Multi-scale t -SNE	$\mathcal{O}(N^2 \log N)$	$\mathcal{O}(N^2)$
Fast Multi-scale t -SNE	$\mathcal{O}(N \log^2 N)$	$\mathcal{O}(N \log N)$

SNE and t -SNE. In (21) and (22), the HD similarities are normalized only through the data points in $\tilde{\mathcal{I}}_i$ for $i \in \mathcal{I}$ whereas all samples are involved in the LD similarity normalization. Matching these HD and LD similarities then decreases the amplitude of the additional terms in the denominators of the LD similarities, favoring LD spreading, i.e., the tendency to expand the LDS, reported as beneficial to DR quality [13].

After computing (20), a Barnes-Hut (BH) algorithm [38] is adapted from Appendix A to estimate (22) as well as its gradient in $\mathcal{O}(N \log N)$ average time. As to (21) and its gradient, we develop their BH approximation in $\mathcal{O}(N \log^2 N)$ average time. It is detailed in Appendix B for the gradient of (21), from which the BH estimation of (21) itself can be easily derived. As presented in Appendix B, it is noteworthy that the BH estimation of the gradient of (21) entails two tree traversals instead of one for the BH approximation of (22) and its gradient. One can thus expect an increased sensitivity of the multi-scale SNE acceleration with respect to the BH threshold θ , compared to fast single- and multi-scale t -SNE, since the estimation quality of the first traversal affects the second one.

The overall time complexity of the presented fast multi-scale schemes evolves as $\mathcal{O}(N \log^2 N)$, importantly without quadratic dependence with N . Note that in fast multi-scale t -SNE, although determining the HD similarities (20) takes $\mathcal{O}(N \log^2 N)$ time once at initialization, only $\mathcal{O}(N \log N)$ average time is required to approximate the gradient of (22), which is iteratively performed in gradient-based optimization. This is emphasized in Table I, which gathers the time complexities of the NE algorithms from Sections II and III.

It is noteworthy that $j \in \tilde{\mathcal{I}}_i$ does not imply $i \in \tilde{\mathcal{I}}_j$. One may easily extend the multi-scale neighbor sets to ensure this property, but experiments revealed that it does not influence the performances much: only tiny improvements in terms of DR quality were observed, at the expense of slightly increased computation times. As multi-scale t -SNE is defined through symmetric similarities, its fast version extends the sets $\tilde{\mathcal{I}}_i$ to ensure the above property, but not fast multi-scale SNE.

The proposed accelerations are based on the definition of multi-scale neighbor sets $\tilde{\mathcal{I}}_i \subset \mathcal{I} \setminus \{i\}$ for all $i \in \mathcal{I}$, with a limited number of elements emulating the intrinsic hierarchy of the data scales. Indeed, $\tilde{\mathcal{I}}_i$ gathers the closest neighbors of ξ_i as well as farther away HD samples, more diffused in the data cloud. This enables accurately describing the local, quickly fluctuating properties around ξ_i while providing a succinct,

higher-order summary of the global HD characteristics. The induced sparse multi-scale HD similarities (20) hence model data proximities at different ranges, unlike their single-scale counterparts (13). An additional $\mathcal{O}(\log N)$ factor arises in the time complexity with respect to fast single-scale approaches, with minor impact compared to the linear dependency in N . Matching the HD similarities (20) with BH estimates of the multi-scale LD similarities enables efficiently mapping both local and global HD structures in the LDS, while avoiding LD crushing. Similarly to multi-scale SNE and t -SNE, multi-scale NeRV and JSE can be accelerated thanks to the multi-scale neighbor sets along with BH approximations.

IV. EXPERIMENTAL RESULTS

The performances of the presented fast multi-scale methods are studied in the context of two main experiments, in terms of both DR quality and computation time. First, Section IV-A considers several moderate-sized databases, with a few thousands of samples each, which enables applying the original, non-accelerated versions of the algorithms for comparison purposes. The assessed methods are hence multi-scale SNE (Ms SNE), multi-scale t -SNE (Ms t -SNE) and their proposed accelerations (FMs SNE and FMs t -SNE). For the sake of comparison, the results of the single-scale t -SNE and of its Barnes-Hut version (BH t -SNE), detailed in Sections II-A and III-A, are also discussed. The behavior of the fast methods with respect to the BH threshold θ is analyzed to suggest adequate values suitable across multiple databases. Second, Section IV-B deals with large-scale data sets, with tens or hundreds of thousands samples each. The quadratic time complexity of the non-accelerated schemes prevents their application on these databases. Therefore, only scalability of the fast schemes is tested in this setting.

In both Section IV-A and IV-B, HD and LD distances are Euclidean and two-dimensional embeddings are targeted ($P = 2$), as typically desired in exploratory visualization. On each database, both fast multi-scale algorithms are applied 30 times, each with different random subsamplings of the HD data set. Average results are then reported over these replications, along with standard deviations. Moreover, thanks to the repetitions, statistical tests allow evaluating the significance of the difference in average performances of the methods. Paired tests are computed when FMs SNE and FMs t -SNE are confronted to each other, and one-sample tests for the mean otherwise. For each studied score, if their 30 values for each fast multi-scale method are normally distributed according to D'Agostino and Pearson's omnibus normality test [40], two-tailed t -tests are employed, and Wilcoxon tests otherwise. The test significance level is fixed to $\alpha = 0.05$. As multiple hypothesis tests result from each database analysis, the Holm-Bonferroni correction is used [41]. Non-significant differences are hence considered as statistically significant with probability bounded by α .

The compared single- and multi-scale algorithms are optimized as advised in their respective papers [13], [27], [32], the proposed fast multi-scale methods being minimized as their non-fast versions. In particular for the latter, K_1 and H are respectively set to 2 and $\lfloor \log_2(N/K_1) \rfloor$, multi-scale optimization [27] is executed and the limited-memory

Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm employed [42]. As to single-scale t -SNE and BH t -SNE, momentum gradient descent is used along with early exaggeration, with parameters as described in [13], [32]. The perplexity K_* is set to 50, as in [32]. In both single- and multi-scale settings, the iterations of the optimization schemes are stopped as soon as (1) the gradient infinite norm is below 10^{-5} , or (2) the cost function relative updates are bounded by 10^{-8} , or (3) a maximum number is reached. The latter maximum number must, however, be carefully defined as different data sets require, obviously, different numbers of iterations before convergence of the LD embedding. Comparing converged and non-converged results of the methods across databases may likely lead to misleading conclusions on the behavior of the algorithms, in terms of both achievable DR quality and computation time. On the other hand, large numbers of samples often require unaffordable amounts of time before reaching convergence of the optimization. For these reasons, the number of iterations of optimization algorithms is limited to 10^5 in Section IV-A. As much fewer iterations are typically required for convergence, this threshold is almost equivalent to an unbounded number of iterations. Thus, in Section IV-A, the optimization schemes terminate according to the first two above conditions, on the gradient norm and relative cost function updates. This ensures comparability of the performances across data sets while maintaining affordable computation times, since moderate-sized databases are tackled. As to Section IV-B, since huge numbers of samples are considered, the maximum number of iterations is reduced to avoid dealing with too long computation times, as one would proceed in a practical context when facing large-scale databases. For BH t -SNE, 10^3 iterations are performed, as suggested in [32]. For FMs SNE and FMs t -SNE, as few as 30 iterations per L-BFGS run are at most executed. All simulations and computation times were obtained thanks to a desktop computer with a 4-core Intel(R) Xeon(R) CPU E3-1245 v5 at 3.5 GHz, with 32 GB RAM.

A. Comparison between fast and non-accelerated methods

The data sets studied in this section originate from the UCI Machine Learning Repository [43]: Anuran calls MFCCs (Anuran), Combined cycle power plant (Plant) [44], [45], Musk version 2 (Musk), Spambase, Gesture phase segmentation (Gesture) [46], Statlog landsat satellite (Satellite), First-order theorem proving (Theorem) [47], Waveform database generator version 2 (Waveform), Isolet, Madelon [48], Abalone and Wine quality (Wine) [49]. Their numbers N and M of samples and features are provided in Table II.

DR quality is usually related to the HD neighborhood preservation in the LDS [17], [50], [51]. It is evaluated thanks to criteria [12], computed in $\mathcal{O}(N^2 \log N)$ time. The K nearest neighbors of ξ_i and $\mathbf{x}_i$ can first be denoted as ν_i^K and n_i^K , respectively in the HDS and LDS. The average normalized agreement of these K -ary neighborhoods develops as

$$Q_{\text{NX}}(K) = \frac{1}{N} \sum_{i \in \mathcal{I}} \frac{|\nu_i^K \cap n_i^K|}{K} \in [0, 1]. \quad (23)$$

Since random LD points yield $\mathbb{E}[Q_{\text{NX}}(K)] = K/(N-1)$,

$$R_{\text{NX}}(K) = \frac{(N-1) Q_{\text{NX}}(K) - K}{N-1-K} \quad (24)$$

enables comparing varying neighborhood sizes [21]. As prevailing neighborhoods are typically local, a log-scale for K is used to display $R_{\text{NX}}(K)$, whose resulting area under the curve (AUC),

$$\text{AUC} = \left(\sum_{K=1}^{N-2} K^{-1} \right)^{-1} \cdot \sum_{K=1}^{N-2} \frac{R_{\text{NX}}(K)}{K} \in [-1, 1], \quad (25)$$

grows with DR quality. The latter is hence characterized by accounting for all scales, with a focus on smaller ones [27].

Figure 3 presents the results as a function of the BH threshold θ on Anuran, Plant, Musk, Spambase and Gesture.

Non-fast methods. On all databases, Ms SNE is consistently the best method in terms of AUC, at the expense of long computation times. Indeed, multi-scale HD and LD similarities enable very accurate tuning of the LDS, with high resolution as they involve both large and small precisions. Nevertheless, the induced LD Gaussian neighborhoods likely rapidly locally fluctuate, making the gradient of Ms SNE sensitive to small LD variations. Its bad scaling hence causes a slow convergence of the optimization process. Next, the AUC of Ms t -SNE is always second to Ms SNE, except on some databases when small BH thresholds are employed with FMs SNE and on Spambase, on which FMs t -SNE performs better. In any case, Ms SNE and Ms t -SNE systematically outperform single-scale t -SNE and BH t -SNE in terms of DR quality. From the running time perspective, Ms t -SNE is much more affordable than Ms SNE. The heavy-tailed Student t -distributed LD similarities likely induce a more stable gradient regarding LD variations. Ms t -SNE is even faster than single-scale t -SNE. Although the comparison is less straightforward in this last case, since Ms t -SNE and single-scale t -SNE rely on different optimization schemes, one may argue that accounting for global properties in Ms t -SNE further constrains the LDS structure, facilitating the cost function minimization. On the other hand, as t -SNE focuses on local aspects, the global LDS organization is left undetermined. Shaping it hence probably requires additional time to fix these degrees of freedom.

Fast methods. The AUC of BH t -SNE is stable with respect to the BH threshold θ , in agreement with the results of [39]. The same remarkable behavior is preserved in FMs t -SNE, which is consistently and statistically significantly above BH t -SNE and t -SNE. FMs SNE is however much more sensitive to the BH threshold: its average AUC quickly drops with θ , with increasing standard deviation as the threshold increases. This is due to the two successive tree traversals involved in the FMs SNE gradient approximation: as the second traversal employs the results of the first one, augmenting θ not only affects the estimation quality of the first pass but also more severely deteriorates the accuracy of the second one, in a cascading effect. The DR quality of FMs SNE thus rapidly decreases with θ , while small BH thresholds yield high AUC since, in this case, both tree traversals deliver precise approximations. Smaller θ nevertheless induces larger running times

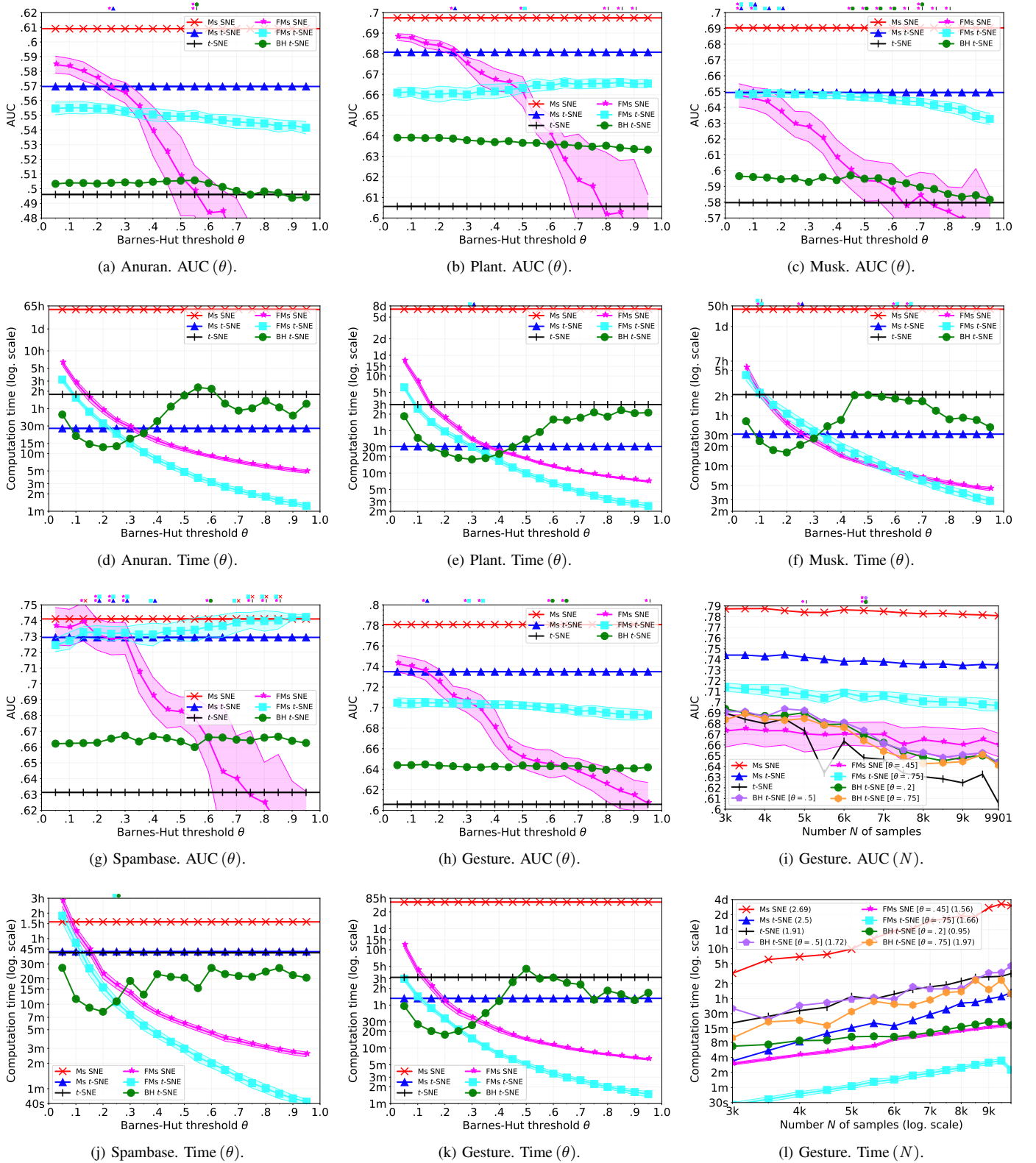


Fig. 3. (a) AUC of the DR methods on Anuran as a function of the BH threshold θ , from 0.05 to 0.95 with 0.05 step. Average performances over 30 random subsamplings are depicted for FMs SNE and FMs t -SNE, with shaded areas delimiting one standard deviation around the mean. Above the figure, pairs of markers indicate schemes that are **not** statistically significantly different, for instance Ms t -SNE and FMs SNE for $\theta = 0.25$. (b)-(c)-(g)-(h) Similar to (a) but respectively on Plant, Musk, Spambase and Gesture. (d)-(e)-(f)-(j)-(k) Analogous to (a)-(b)-(c)-(g)-(h), but for the computation time. Log-scale is used for the y -axis and s, m, h and d refer to second, minute, hour and day. (i) Similar to (h) but for subsamples of the Gesture data set, with fixed BH thresholds θ described in the legend. The number N of samples varies with steps of 500 on the x -axis. (l) Analogous to (i) but for the computation time, on the y -axis. Both axes are in log-scale. For each DR algorithm, the slope of a linear regression of its results in log-log scale is provided in parentheses in the legend.

for both FMs SNE and FMs t -SNE. Considering the AUC and computation time dependence of FMs SNE with respect to θ , its BH threshold should be carefully determined to ensure a proper balance between DR quality and running time. For this reason, a threshold $\theta = 0.45$ is considered for FMs SNE in the remaining experiments. On the other hand, FMs t -SNE can accommodate with larger BH thresholds thanks to its AUC dependence with θ , enabling to decrease computation time without suffering from significant loss in DR quality. As it can reach high DR quality with low computation time much more easily and as it is intrinsically faster since entailing only one tree traversal per gradient estimation instead of two, one can state that FMs t -SNE is superior to FMs SNE. For the subsequent experiments, $\theta = 0.75$ is used with FMs t -SNE. Noticeably, although both FMs SNE and FMs t -SNE get, as expected, faster as θ increases, BH t -SNE behaves differently. Indeed, on all data sets, the running time of BH t -SNE first decreases with θ , reaches a minimum at around $\theta = 0.2$ and then increases before remaining roughly stable. This intriguing phenomenon may be explained as larger θ accelerates each gradient estimation but also induces coarser approximations. These likely result in more iterations before reaching convergence of the momentum gradient descent. The L-BFGS scheme employed in FMs methods less suffers from these rougher estimations, probably thanks to the successive Hessian approximations it computes, which partly cancel the influence of noisy gradient estimates. Therefore, for most BH thresholds, FMs t -SNE is statistically significantly faster than BH t -SNE. For completeness of the comparison, three thresholds are further employed with BH t -SNE, namely 0.2 as it minimizes running times on Anuran, Musk, Spambase, Gesture and almost Plant, 0.5 as in [32], and 0.75 as larger value.

Figures 3(i) and 3(l) depict the AUC and running time of the methods as a function of the number of points in subsamples of Gesture. In terms of AUC, Ms SNE remains the best method, followed by Ms t -SNE and then FMs t -SNE. The last three methods are close, smaller data sets favoring BH t -SNE and FMs SNE being superior on larger ones. Smaller BH thresholds would improve the accuracy of FMs SNE on smaller subsamples. As to the computation times, FMs t -SNE is consistently and statistically significantly the fastest method, which raises the interest for it in light of its AUC performances. FMs SNE is second and BH t -SNE is indeed faster with $\theta = 0.2$. Using log-log axes as in Fig. 3(l), the slope of linear regressions of the method timings should reveal their time complexity. One would expect a slope near 2 for t -SNE, above 2 for Ms SNE and Ms t -SNE, above 1 for BH t -SNE and higher than 1 for FMs SNE and FMs t -SNE, as these methods respectively have $\mathcal{O}(N^2)$, $\mathcal{O}(N^2 \log N)$, $\mathcal{O}(N \log N)$ and $\mathcal{O}(N \log^2 N)$ theoretical time complexity. Such slopes are indeed experimentally observed (larger numbers of samples would obviously yield more accurate values at the expense of much larger running times for non-fast methods).

Table II provides performances of the methods on all databases using specific BH thresholds. Ms SNE keeps the best AUC and slowest computation times. With more affordable running times, Ms t -SNE is always second in AUC except on Spambase on which FMs t -SNE is superior. Ms t -SNE

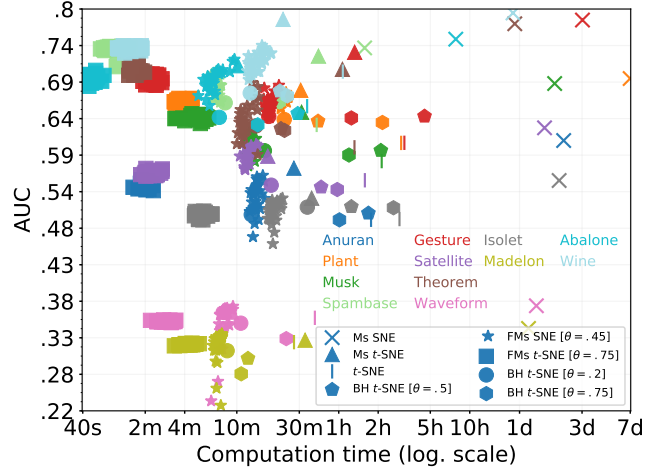


Fig. 4. AUC of the DR methods on moderate-sized data sets as a function of their computation time in log-scale, with s, m, h and d referring to second, minute, hour and day. Results of FMs SNE and FMs t -SNE are depicted for 30 different random subsamplings. The markers and colors respectively differentiate the DR methods and data sets. The data sets names are indicated using their associated colors to enable their identification. Contrasts best viewed using colors.

is also faster than t -SNE, except on Spambase and Madelon. FMs t -SNE is consistently and statistically significantly the fastest method and yields the highest AUC among accelerated schemes, except on Plant and Satellite, on which FMs SNE is superior, without statistical significance on Plant, and on Isolet, on which BH t -SNE performs well, without being statistically different from FMs SNE in AUC. FMs t -SNE also statistically significantly beats t -SNE in terms of AUC, except on Waveform and Madelon. Computation time of BH t -SNE is indeed minimized with $\theta = 0.2$, except on Waveform on which $\theta = 0.5$ is slightly faster. FMs SNE is not as fast as FMs t -SNE but still generally results in statistically significantly larger AUC than t -SNE and BH t -SNE, in less or similar time as BH t -SNE and less time than t -SNE and Ms t -SNE. The AUC difference between Ms SNE and FMs SNE is always greater than between Ms t -SNE and FMs t -SNE, probably due to the two tree traversals required for the gradient estimation.

Figure 4 illustrates the results of Table II. The high concentration of FMs t -SNE markers in the upper left part of the figure highlights its superiority with respect to the other DR schemes. The performances of FMs SNE and FMs t -SNE being reported for 30 different random subsamplings, one can visually observe the variability of their performances. While FMs SNE is stable in terms of computation time but slightly less in AUC for reasons detailed below, FMs t -SNE is remarkably consistent across the random subsamplings. All conclusions drawn from Table II are depicted in Fig. 4.

Figure 5 depicts $R_{NX}(K)$ curves for Anuran, Musk and Satellite. Both original and accelerated multi-scale methods outperform single-scale algorithms in terms of local and global neighborhood preservation. Single-scale schemes are only close or slightly above for neighborhood sizes near the perplexity. Indeed, as mentioned in [27], at the expense of a small loss around the perplexity, the multi-scale behavior

TABLE II

PERFORMANCES OF THE DR METHODS ON MODERATE-SIZED DATA SETS, IN TERMS OF AUC AND OF COMPUTATION TIME IN SECONDS. FOR FMS SNE AND FMS t -SNE, THE PROVIDED RESULTS ARE AVERAGES OVER 30 RANDOM SUBSAMPLINGS, WITH STANDARD DEVIATIONS IN PARENTHESES. IN EACH ROW, THE GREATEST AUC OR SMALLEST TIME IS INDICATED IN BOLD, AND AN ASTERISK HIGHLIGHTS THE HIGHEST AUC AMONG THE FAST SCHEMES. RESULTS ARE IN ITALIC (RESP. UNDERLINED) WHEN NOT STATISTICALLY SIGNIFICANTLY DIFFERENT FROM FMS SNE (RESP. FMS t -SNE).

		Ms SNE	FMS SNE $\theta = 0.45$	Ms t -SNE	FMS t -SNE $\theta = 0.75$	t -SNE	BH t -SNE $\theta = 0.2$	BH t -SNE $\theta = 0.5$	BH t -SNE $\theta = 0.75$
Anuran $N = 7195$ $M = 22$	AUC [%]	60.905	52.506 (2.047)	56.966	54.495 (0.388)*	49.605	50.341	50.543	49.592
	Time [s]	187629	822 (55)	1637	119 (9)	6352	773	6064	3645
Plant $N = 9568$ $M = 5$	AUC [%]	69.743	<u>66.602 (0.474)*</u>	68.072	<i>66.545 (0.244)</i>	60.563	63.901	63.658	63.476
	Time [s]	603853	1315 (69)	1845	231 (20)	10783	1381	2503	7760
Musk $N = 6598$ $M = 166$	AUC [%]	69.033	60.042 (1.191)	64.941	64.148 (0.299)*	57.973	59.453	59.484	58.835
	Time [s]	160163	736 (39)	1873	299 (40)	7660	974	7546	4310
Spambase $N = 4601$ $M = 57$	AUC [%]	74.11	68.352 (1.157)	72.942	74.019 (0.508)*	63.132	66.281	66.355	66.417
	Time [s]	5675	404 (25)	2519	70 (9)	2442	489	1238	1267
Gesture $N = 9901$ $M = 18$	AUC [%]	78.084	66.011 (1.102)	73.493	69.693 (0.518)*	60.581	64.345	64.367	64.119
	Time [s]	261167	1035 (63)	4761	136 (12)	11412	1052	16263	4492
Satellite $N = 6435$ $M = 36$	AUC [%]	62.715	58.617 (1.036)*	<i>58.672</i>	56.269 (0.264)	55.238	54.538	54.258	53.916
	Time [s]	134074	718 (49)	1034	138 (14)	5698	1098	2641	3506
Theorem $N = 6118$ $M = 53$	AUC [%]	77.553	63.577 (2.302)	<u>71.056</u>	71.031 (0.459)*	59.935	<i>62.987</i>	<i>62.601</i>	62.347
	Time [s]	79442	686 (79)	3840	101 (6)	4747	839	1304	1388
Waveform $N = 5000$ $M = 40$	AUC [%]	37.343	<u>34.885 (3.052)</u>	36.444	<i>35.136 (0.09)*</i>	<i>35.625</i>	<i>34.861</i>	<i>34.02</i>	32.656
	Time [s]	116158	467 (37)	547	176 (17)	2366	643	<i>469</i>	1431
Isolet $N = 7797$ $M = 617$	AUC [%]	55.203	<u>50.711 (1.438)</u>	52.684	<i>50.333 (0.333)</i>	49.71	<i>51.408</i>	<i>51.469*</i>	<i>51.318</i>
	Time [s]	173318	1118 (79)	2236	328 (25)	10473	2073	4494	9424
Madelon $N = 4400$ $M = 500$	AUC [%]	34.112	<u>31.367 (2.213)</u>	<i>32.442</i>	<i>31.832 (0.083)*</i>	<i>32.155</i>	<i>30.934</i>	29.834	27.637
	Time [s]	101184	420 (20)	1993	255 (25)	1640	506	730	648
Abalone $N = 4177$ $M = 7$	AUC [%]	75.36	<u>69.299 (1.509)</u>	71.576	<i>69.809 (0.46)*</i>	65.798	64.207	64.749	63.146
	Time [s]	28163	395 (60)	613	47 (4)	2065	441	1751	867
Wine $N = 6497$ $M = 11$	AUC [%]	79.093	71.949 (1.31)	78.184	74.071 (0.237)*	70.731	67.665	67.954	67.261
	Time [s]	77021	847 (105)	1353	93 (12)	3868	759	1280	1457

uplifts the curve for all other neighborhood sizes. FMs SNE is close to Ms SNE in terms of global neighborhood preservation but does not reproduces local ones as well. FMs t -SNE behaves in the opposite way when compared to Ms t -SNE.

It is noteworthy that in Fig. 3, Table II, Fig. 4 and Fig. 5, the standard deviations of FMs SNE are systematically larger than for FMs t -SNE. This indicates that FMs SNE is more sensitive to the random subsamplings than FMs t -SNE, yielding magnified variations of the small neighborhood preservation in Fig. 5. This increased sensitivity likely results from the two tree traversals of the FMs SNE gradient estimation. Indeed, the BH algorithm is directly dependent on the LD configuration, which is itself related to the considered HD data subsampling. The larger influence of the BH algorithm on FMs SNE than on FMs t -SNE, caused by the two tree passes performed per data point at each iteration, then induces an increased sensitivity of FMs SNE with respect to the HD data subsamplings.

B. Scalability analysis of the fast methods

Non-accelerated NE schemes can manage up to a few thousands samples, due to their quadratic time complexity. Fast methods better scale on large data sets, with tens or

hundreds of thousands samples. This section hence studies their performances in this last context, using data sets from the UCI Machine Learning Repository [43]: Statlog Shuttle (Shuttle), Sensorless Drive Diagnosis (Sensorless), SGEMM GPU kernel performance (Sgemm) [52], Covtype and Gas sensors for home activity monitoring (Gas) [53]. Table III provides their numbers N and M of samples and features.

To assess DR quality, the AUC is no longer suitable as its evaluation requires $\mathcal{O}(N^2 \log N)$ time. Nevertheless, the start of the $R_{\text{NX}}(K)$ curve, with K ranging from 1 to $\bar{K}$, can be computed in $\mathcal{O}(\bar{K}N \log N)$ time, thanks to vantage-point trees. The area AUC of this reduced curve develops as (25), but with the upper limits of the sums replaced with $\bar{K}$. The smaller $\bar{K}$ compared to N , the more the AUC focuses on local neighborhoods compared to AUC. In this section, $\bar{K}$ is set to 10^4 , to consider similar neighborhood sizes as in Section IV-A.

Table III gathers the results. On Shuttle and Sensorless, both with tens of thousands samples, FMs SNE performs best in terms of AUC, but with the largest computation times. This is due to the $\mathcal{O}(N \log^2 N)$ time complexity of each gradient estimation, moreover involving two tree traversals per data point, whereas the optimization iterations in BH t -SNE and FMs t -SNE only cost $\mathcal{O}(N \log N)$ time, with a single tree

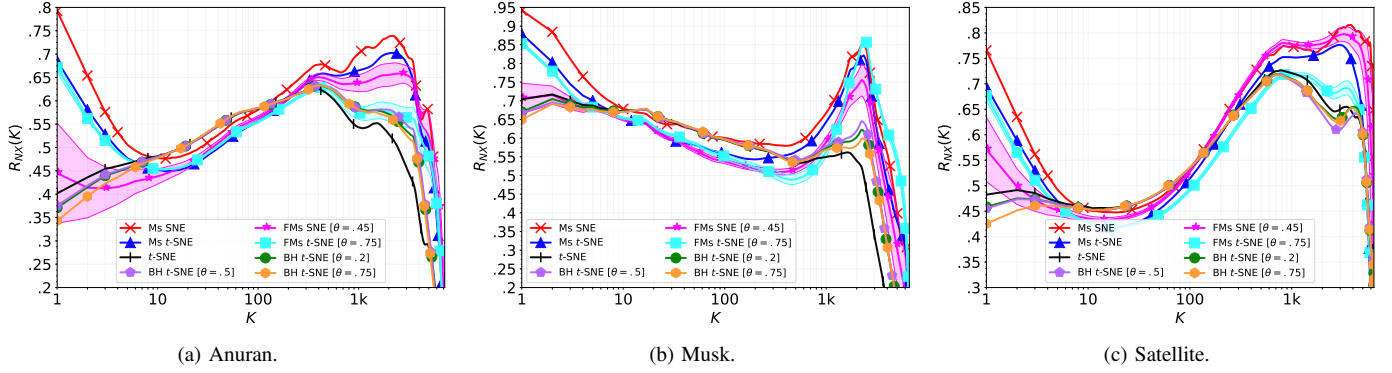


Fig. 5. (a) $R_{NX}(K)$ curves of the DR methods on Anuran as a function of the neighborhood size K , using the same BH thresholds as in Table II. Average curves over 30 random subsamplings are depicted for FMs SNE and FMs t -SNE, with shaded areas delimiting one standard deviation around the mean, as in Fig. 3. A log-scale is employed for the neighborhood size K . (b)-(c) Similar but respectively on Musk and Satellite.

TABLE III

PERFORMANCES OF THE FAST DR METHODS ON LARGE-SCALE DATA SETS, IN TERMS OF $\overline{AUC}$ AND OF COMPUTATION TIME IN SECONDS. FOR FMs SNE AND FMs t -SNE, THE PROVIDED RESULTS ARE AVERAGES OVER 30 RANDOM SUBSAMPLINGS, WITH STANDARD DEVIATIONS IN PARENTHESES. IN EACH ROW, THE GREATEST $\overline{AUC}$ OR SMALLEST TIME IS INDICATED IN BOLD. RESULTS ARE IN ITALIC (RESP. UNDERLINED) WHEN NOT STATISTICALLY SIGNIFICANTLY DIFFERENT FROM FMs SNE (RESP. FMs t -SNE).

		FMs SNE $\theta = 0.45$	FMs t -SNE $\theta = 0.75$	$\theta = 0.2$	BH t -SNE $\theta = 0.5$	$\theta = 0.75$
Shuttle $N = 58000$ $M = 9$	$\overline{AUC}$ [%]	57.684 (0.716)	54.009 (0.35)	50.404	50.448	50.396
	Time [s]	33448 (3732)	1146 (26)	5610	2234	1725
Sensorless $N = 58509$ $M = 48$	$\overline{AUC}$ [%]	43.996 (0.576)	<i>43.945 (0.593)</i>	38.934	38.91	38.74
	Time [s]	17790 (1257)	550 (21)	6350	2316	1745
Sgemm $N = 241600$ $M = 18$	$\overline{AUC}$ [%]	—	59.022 (0.057)	12.277	12.277	12.277
	Time [s]	—	11812 (254)	3946	1144	805
Coverttype $N = 581012$ $M = 54$	$\overline{AUC}$ [%]	—	42.556 (0.104)	0.756	0.756	0.756
	Time [s]	—	7083 (151)	11813	3354	2277
Gas $N = 928991$ $M = 11$	$\overline{AUC}$ [%]	—	49.559 (9.218)	5.283	5.283	5.283
	Time [s]	—	16449 (1664)	<u>16835</u>	4719	3171

pass per sample. FMs t -SNE is statistically significantly the fastest scheme, also with statistically significantly greater $\overline{AUC}$ than BH t -SNE. It is noteworthy that increasing θ decreases the running time of BH t -SNE, differently to what is observed in Section IV-A. Indeed, computing 10^3 momentum gradient descent steps does not guarantee LDS convergence. Therefore, larger BH thresholds may accelerate BH t -SNE optimization when stopped after 10^3 iterations, as increasing θ yields cheaper BH estimates and faster iteration evaluations.

Figure 6 depicts 2-D embeddings of Shuttle as computed by FMs SNE, FMs t -SNE and BH t -SNE. This database contains unbalanced labels, 80% of its samples belonging to a single class. Visually, both fast multi-scale methods tend to more regroup the classes than BH t -SNE. Figure 6(c) moreover presents cluster over-emphasis produced by BH t -SNE, which affects its $\overline{AUC}$ with respect to FMs SNE and FMs t -SNE. Indeed, although matching Gaussian similarities with heavy-tailed Student t ones is motivated by the crowding problem to compensate for the relative loss of volume from the HDS to the LDS [13], it typically introduces artificial clusters in the embedding since mid-range to distant neighbors are further repelled [28]. In Fig. 6(b), FMs t -SNE is also subject to this phenomenon but to a lesser extent, as multiple Gaussian

functions with increasing tails are averaged in the HDS, reducing the gap with the heavy-tailed Student t distribution. In contrast, as suggested by Fig. 6(a) and its dominant $\overline{AUC}$, FMs SNE is more robust to cluster over-emphasis, thanks to the multi-scale Gaussian similarities it employs in both spaces.

As to Sgemm, Coverttype and Gas, the results of FMs SNE with $\theta = 0.45$ are not reported in Table III as these databases contain hundreds of thousands samples, inducing intractable computation times when applying the method with 30 different random subsamplings. Larger thresholds could have been employed, but they yield unstable DR quality as suggested in Fig. 3. Although BH t -SNE is the fastest scheme, FMs t -SNE impressively outperforms it in terms of $\overline{AUC}$. The poor DR quality achieved by BH t -SNE, with the same $\overline{AUC}$ for all three thresholds, suggests that it is far from convergence, before θ starts influencing the LD positioning. On the other hand, FMs t -SNE delivers fair $\overline{AUC}$, for instance in 4.5 hours on average on Gas, containing almost one million samples. The 2-D embeddings of this data set illustrate in Fig. 7 these results: the LDS computed by FMs t -SNE in Fig. 7(a) indeed reveals a structure related to the three classes of Gas. Such an organization is not obtained by BH t -SNE in Fig. 7(b) and 7(c), which hardly and fuzzily arrange the data points regardless of

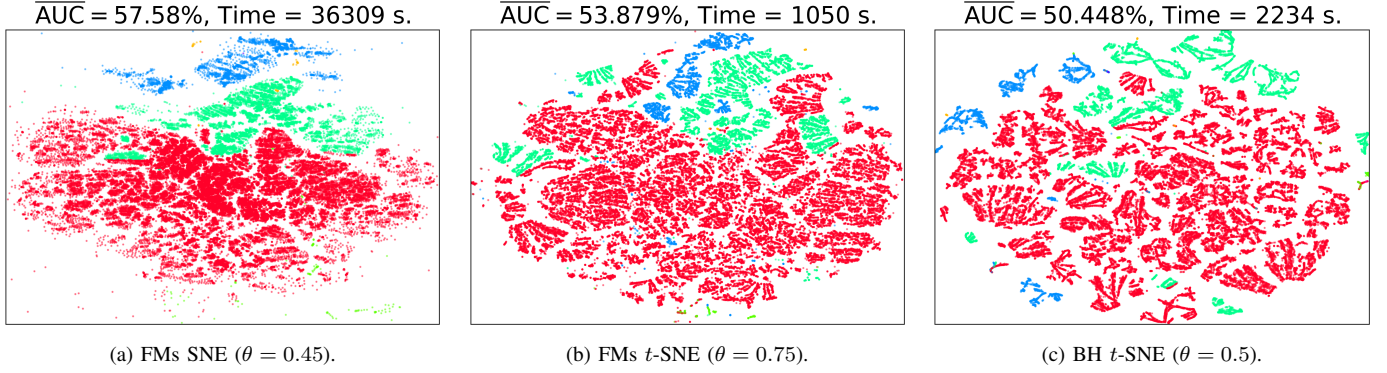


Fig. 6. (a) 2-D embedding of Shuttle obtained by one of the 30 applications of FMs SNE. The classes in the data set are depicted using different colors. The computation time provided in the title of the figure is in seconds. (b)-(c) Similar but for FMs t -SNE and BH t -SNE. The embedding of BH t -SNE is displayed with $\theta = 0.5$ as it maximizes its $\overline{\text{AUC}}$ on Shuttle among $\{0.2, 0.5, 0.75\}$ in Table III.

the BH threshold θ , leading to a weak $\overline{\text{AUC}}$.

Figure 8 further depicts the results of Table III. For Shuttle and Sensorless, FMs t -SNE indeed lies in the upper left corner of the figure, while BH t -SNE is the fastest on Sgemm, Covertype and Gas, but with poor $\overline{\text{AUC}}$ compared to FMs t -SNE. It is noteworthy that FMs t -SNE has a less stable $\overline{\text{AUC}}$ on Gas across the random subsamplings compared to all other data sets, due to the challenging nature and large size of Gas.

V. CONCLUSION

In DR, NE methods produce LD versions of HD data sets retaining HD neighborhoods impressively well. HD structures extraction drives a lot of attention, leading to efficient frameworks easing subsequent classification [54]–[56] and clustering [57]–[59]. In visualization, multi-scale schemes outperform single-scale ones by accounting for both local and global data interactions, improving the HD neighborhood preservation. Yet, their $\mathcal{O}(N^2 \log N)$ time complexity with N samples limits their applicability to a few thousands data. To tackle larger databases, this paper develops $\mathcal{O}(N \log^2 N)$ accelerations of the multi-scale NE methods. While fast single-scale methods harness the intrinsic sparsity of their HD similarities, the proposed fast multi-scale schemes account for the dense nature of the multi-scale HD similarities by defining neighbor sets for each datum incorporating samples summarizing the data cloud from different perspectives. These sets gather the closest HD neighbors to each datum to accurately capture the locally fluctuating structures, as well as HD points sampled with different resolutions to consider the global data cloud properties. With a BH algorithm avoiding LD crushing, the accounted subsets of HD data interactions enable relevantly scoring the LD coordinates quality. The LDS is hence efficiently shaped to reproduce HD relationships at all scales. Although the presented algorithms evolve in $\mathcal{O}(N \log^2 N)$, it is noteworthy that the gradient of FMs t -SNE is iteratively estimated in $\mathcal{O}(N \log N)$ average time. As it requires a single tree traversal per datum for each gradient approximation, it is superior to FMs SNE, as confirmed experimentally. Simulations highlight that fast multi-scale methods consistently and statistically significantly outperform state-of-the-art single-scale schemes in DR quality, while tremendously

accelerating their non-fast versions, enabling multi-scale visualization of large data sets.

In extremely HD spaces, the efficiency of the nearest neighbor searches through vantage-point trees may decay, due to the curse of dimensionality [1]. In such cases, one can however first reduce the dimension with PCA while preserving most information content [32]. Another option is to employ random projection trees combined with neighbor exploring techniques [35]. As to future lines of work, our methods can be extended to class-aware settings [56], inspired by recent successful works [60]. Weights could also be associated to the multi-scale sampled neighbors, proportionally to the number of neglected interactions they represent on average. These interactions would also influence the LDS computation, to improve DR quality. The design of such weights should however consider both the HD and LD coordinates, which are iteratively updated. Another perspective is to study fast multipole methods [34] and negative sampling techniques [35], to analyze whether they yield better trade-offs between running time and DR quality. In this line, variants of the BH approximation condition will be investigated as well, according to the alternatives outlined in Appendix A. These refinements would indeed ease meeting the BH approximation launching, hence possibly further reducing computation time without harming the DR performances.

APPENDIX A

BARNES-HUT ALGORITHM FOR t -SNE

The gradient of $C_{t\star}$ with respect to the LD points extends as

$$\frac{1}{4} \frac{\partial C_{t\star}}{\partial \mathbf{x}_v} = \sum_{i \in \mathcal{I} \setminus \{v\}} (\tau_{vi\star} - t_{vi}) (\mathbf{x}_v - \mathbf{x}_i) (1 + d_{vi}^2)^{-1} \quad (26)$$

$$= \underbrace{\sum_{i \in \mathcal{I} \setminus \{v\}} \tau_{vi\star} \frac{\mathbf{x}_v - \mathbf{x}_i}{1 + d_{vi}^2}}_{\mathbf{f}_v^{t\star}} - \underbrace{\frac{\sum_{i \in \mathcal{I} \setminus \{v\}} \frac{\mathbf{x}_v - \mathbf{x}_i}{(1 + d_{vi}^2)^2}}{\sum_{k \in \mathcal{I}, l \in \mathcal{I} \setminus \{k\}} (1 + d_{kl}^2)^{-1}}}_{g^t}. \quad (27)$$

After replacing $\tau_{vi\star}$ with $\tilde{\tau}_{vi\star}$, the sparsity of $\tilde{\tau}_{vi\star}$ enables estimating $\mathbf{f}_v^{t\star}$ for all $v \in \mathcal{I}$ in $\mathcal{O}(K_\star N)$ time. To estimate

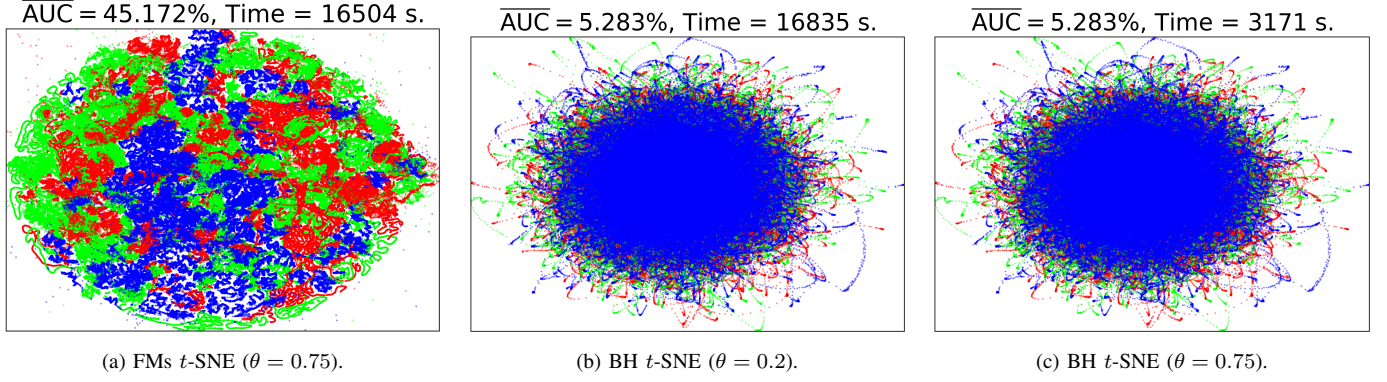


Fig. 7. (a) 2-D embedding of Gas obtained by one of the 30 applications of FMs t -SNE. The classes in the data set are depicted using different colors. The computation time provided in the title of the figure is in seconds. (b)-(c) Similar but for BH t -SNE, respectively with $\theta = 0.2$ and $\theta = 0.75$.

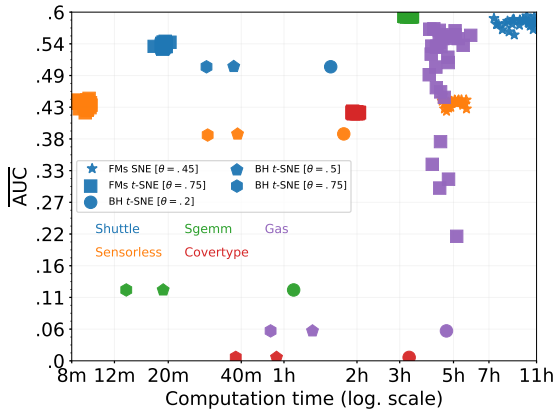


Fig. 8. Similar to Fig. 4 but for the fast DR methods on large-scale data sets, with their AUC along the y -axis. Contrasts best viewed using colors.

$\mathbf{g}_v^t / g_v^t$ for all $v \in \mathcal{I}$ in less than $\mathcal{O}(N^2)$ time, first note that if $d_{ij} \ll d_{vi} \approx d_{vj}$, then $t_{vi} \approx t_{vj}$ [32]. To benefit from this observation and as the LD dimension P is small in visualization tasks, a quad- ($P = 2$) or octree ($P = 3$) can be constructed on $\mathbf{X}$ in $\mathcal{O}(N)$ time [32], as illustrated in Fig. 1(a). Each node of the tree is associated with a rectangular cell in the LDS, the root containing the whole LD embedding. Each non-leaf node has 4 ($P = 2$) or 8 ($P = 3$) children partitioning the cell into 4 or 8 equal parts at its corners. A leaf contains at most one LD point. For each cell c , let $\mathcal{T}_c \subset \mathcal{I}$ index the LD points inside it. In the node related to c , one can store $|\mathcal{T}_c|$ and the center-of-mass $\mathbf{m}_c := \frac{1}{|\mathcal{T}_c|} \sum_{i \in \mathcal{T}_c} \mathbf{x}_i$ of the points indexed in $\mathcal{T}_c$. Creating the quad- or octree consists in inserting the LD points one after the other, dividing leaves as soon as they contain two LD points, while updating $|\mathcal{T}_c|$ and $\mathbf{m}_c$ of the traversed nodes [32]. Quadrees can be generalized to any dimension P but are not suitable to large ones, as the number of children per node grows exponentially with P .

Once the tree is defined, a depth-first search enables estimating $\mathbf{g}_v^t$ in $\mathcal{O}(\log N)$ average time, if the splits in the tree partition the data into roughly equal parts [33]. At each node with associated cell c during the traversal, one determines whether the terms over $\mathcal{T}_c$ in $\mathbf{g}_v^t$ are similar to each other, which happens when the cell c is sufficiently small and far enough

from $\mathbf{x}_v$, in light of the above observation ($d_{ij} \ll d_{vi} \approx d_{vj}$ with i and j in $\mathcal{T}_c$). The condition [38]

$$r_c / d_{v, \mathbf{m}_c} < \theta \quad (28)$$

formalizes this idea, with r_c the cell diagonal length, $d_{v, \mathbf{m}_c}$ the LD distance between $\mathbf{x}_v$ and $\mathbf{m}_c$, and $\theta \in [0, 1]$ a user-defined threshold parameter trading off accuracy and speed. One could also replace r_c in (28) with the average distance between $\mathbf{m}_c$ and the LD points indexed in $\mathcal{T}_c$, or the diagonal of the smallest box in cell c which contains all points in $\mathcal{T}_c$. These quantities can be easily derived, respectively in $\mathcal{O}(N \log N)$ time and when creating the tree, and could improve the trade-off between computation time and DR quality.

As soon as (28) holds, the depth-first search stops and the part of $\mathbf{g}_v^t$ over $\mathcal{T}_c$ is estimated as

$$\sum_{i \in \mathcal{T}_c} \frac{\mathbf{x}_v - \mathbf{x}_i}{(1 + d_{vi}^2)^2} \approx |\mathcal{T}_c| \frac{\mathbf{x}_v - \mathbf{m}_c}{(1 + d_{v, \mathbf{m}_c}^2)^2}, \quad (29)$$

since the terms over $\mathcal{T}_c$ are similar to each other and hence close to their approximation using $\mathbf{m}_c$ as representative for the LD points indexed in $\mathcal{T}_c$. During the same traversal, one can also estimate the sum $g_v^t := \sum_{l \in \mathcal{I} \setminus \{v\}} \frac{1}{1 + d_{vl}^2}$. After completing N passes, approximations $\tilde{\mathbf{g}}_v^t$ and $\tilde{g}_v^t$ of $\mathbf{g}_v^t$ and g_v^t are available for all $v \in \mathcal{I}$. The sum $\tilde{g}^t := \sum_{v \in \mathcal{I}} \tilde{g}_v^t$ estimates g^t . Since each traversal costs $\mathcal{O}(\log N)$ average time, $\tilde{g}^t$ and $\tilde{\mathbf{g}}_v^t$ for all $v \in \mathcal{I}$ are obtained in $\mathcal{O}(N \log N)$ average time.

Figure 1 illustrates the computation of $\tilde{\mathbf{g}}_v^t$ and $\tilde{g}_v^t$. The square data point corresponds to $\mathbf{x}_v$ in Fig. 1(b) and 1(c). Figure 1(b) computes $\mathbf{g}_v^t$ and g_v^t exactly, leading to $\mathcal{O}(N^2)$ time complexity when performed for all $v \in \mathcal{I}$. Figure 1(c) sketches their BH approximation in $\mathcal{O}(\log N)$ average time, in which the centers-of-mass of the LD points in farther cells to $\mathbf{x}_v$ summarize their contributions in $\mathbf{g}_v^t$ and g_v^t .

APPENDIX B

BARNES-HUT ALGORITHM FOR MULTI-SCALE SNE

The gradient of (21) with respect to the LD points writes

$$H \frac{\partial \tilde{\mathcal{C}}}{\partial \mathbf{x}_v} = \sum_{i \in \mathcal{I} \setminus \{v\}} \underbrace{\left(\sum_{h=1}^H p_h s_{vhi} \right)}_{f_{vi}} \frac{\tilde{\sigma}_{vi}}{s_{vi}} (\mathbf{x}_v - \mathbf{x}_i) \quad (30)$$

$$- \sum_{h=1}^H p_h \underbrace{\left[\sum_{i \in \mathcal{I} \setminus \{v\}} \frac{\tilde{\sigma}_{vi}}{s_{vi}} s_{vih} \right]}_{g_{vh}} \underbrace{\left[\sum_{i \in \mathcal{I} \setminus \{v\}} s_{vih} (\mathbf{x}_v - \mathbf{x}_i) \right]}_{\mathbf{g}_{vh}} \quad (31)$$

$$+ \sum_{i \in \mathcal{I} \setminus \{v\}} \underbrace{\left(\sum_{h=1}^H p_h s_{ivh} \right)}_{f_{iv}} \frac{\tilde{\sigma}_{iv}}{s_{iv}} (\mathbf{x}_v - \mathbf{x}_i) \quad (32)$$

$$- \sum_{h=1}^H p_h \sum_{i \in \mathcal{I} \setminus \{v\}} s_{ivh} (\mathbf{x}_v - \mathbf{x}_i) \underbrace{\left[\sum_{j \in \mathcal{I} \setminus \{i\}} \frac{\tilde{\sigma}_{ij}}{s_{ij}} s_{ijh} \right]}_{g_{ih}}. \quad (33)$$

The denominator of (9), $\bar{s}_{ih} := \sum_{k \in \mathcal{I} \setminus \{i\}} \exp(-p_h d_{ik}^2/2)$, can first be estimated for all $i \in \mathcal{I}$ and $h = 1, \dots, H$ in $\mathcal{O}(N \log^2 N)$ average time by extending the BH algorithm from Appendix A. Then, f_{vi} and s_{vi} can be approximated for all $v \in \mathcal{I}$ and $i \in \tilde{\mathcal{I}}_v$ in $\mathcal{O}(N \log^2 N)$ time as $|\tilde{\mathcal{I}}_v|$ is $\mathcal{O}(\log N)$. Afterwards, estimating the right-hand-side of (30) for all $v \in \mathcal{I}$ takes $\mathcal{O}(N \log N)$ time, thanks to the sparsity of $\tilde{\sigma}_{vi}$ which reduces the sum over $\mathcal{I} \setminus \{v\}$ to a sum over $\tilde{\mathcal{I}}_v$. Similarly, (32) is approximated for all $v \in \mathcal{I}$ in $\mathcal{O}(N \log N)$ time, and g_{vh} for all $v \in \mathcal{I}$ and $h = 1, \dots, H$ in $\mathcal{O}(N \log^2 N)$ time. On the other hand, $\mathbf{g}_{vh}$ satisfies

$$\bar{s}_{vh} \mathbf{g}_{vh} = \sum_{i \in \mathcal{I} \setminus \{v\}} \exp(-p_h d_{vi}^2/2) (\mathbf{x}_v - \mathbf{x}_i), \quad (34)$$

where the right-hand-side is estimated in $\mathcal{O}(\log N)$ average time thanks to the BH algorithm. Thus, (31) is estimated for all $v \in \mathcal{I}$ in $\mathcal{O}(N \log^2 N)$ average time. Finally, (33) extends as

$$\sum_{i \in \mathcal{I} \setminus \{v\}} \left[\sum_{h=1}^H p_h \exp(-p_h d_{iv}^2/2) \frac{g_{ih}}{\bar{s}_{ih}} \right] (\mathbf{x}_v - \mathbf{x}_i). \quad (35)$$

To approximate (35), the quadtree is extended by storing, in each cell c , $\mathcal{O}(\log N)$ additional attributes consisting of

$$a_h^c := |\mathcal{T}_c|^{-1} \sum_{i \in \mathcal{T}_c} g_{ih} / \bar{s}_{ih}, \quad (36)$$

for $h = 1, \dots, H$, yielding a tree with $\mathcal{O}(N \log N)$ memory cost [33]. Depth-first searching the resulting tree enables estimating (35) in $\mathcal{O}(\log^2 N)$ average time, by stopping the traversal as soon as (28) holds and approximating the part of (35) with $i \in \mathcal{T}_c$ through a sum of $\mathcal{O}(\log N)$ terms,

$$|\mathcal{T}_c| \left[\sum_{h=1}^H p_h \exp(-p_h d_{v, \mathbf{m}_c}^2/2) a_h^c \right] (\mathbf{x}_v - \mathbf{m}_c). \quad (37)$$

Estimating (33) for all $v \in \mathcal{I}$ hence takes $\mathcal{O}(N \log^2 N)$ average time, which is the time complexity of approximating the gradient of (21). It is noteworthy that the estimation entails two tree traversals for each $v \in \mathcal{I}$, to first approximate (30), (31), (32) and (36), and then estimate (33) during the second search.

REFERENCES

- [1] J. A. Lee and M. Verleysen, *Nonlinear dimensionality reduction*. Springer Science & Business Media, 2007.
- [2] X. Li, H. Zhang, R. Zhang, Y. Liu, and F. Nie, "Generalized Uncorrelated Regression with Adaptive Graph for Unsupervised Feature Selection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 5, pp. 1587–1595, 2018.
- [3] R. Zhang, F. Nie, Y. Wang, and X. Li, "Unsupervised Feature Selection via Adaptive Multimeasure Fusion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 9, pp. 2886–2892, 2019.
- [4] I. Borg and P. J. F. Groenen, *Modern Multidimensional Scaling: Theory and applications*. Springer Science & Business Media, 2005.
- [5] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, Sept. 1990, DOI: 10.1109/5.58325.
- [6] G. Hinton and S. Roweis, "Stochastic neighbor embedding," in *NIPS*, vol. 15, 2002, pp. 833–840.
- [7] I. T. Jolliffe, "Principal component analysis and factor analysis," in *Principal component analysis*. Springer, 1986, pp. 115–128.
- [8] J. W. Sammon, "A nonlinear mapping for data structure analysis," *IEEE Trans. Comput.*, vol. 100, no. 5, pp. 401–409, 1969.
- [9] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000, DOI: 10.1126/science.290.5500.2319.
- [10] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in neural information processing systems*, 2002, pp. 585–591.
- [11] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, Dec. 2000, DOI: 10.1126/science.290.5500.2323.
- [12] J. A. Lee and M. Verleysen, "Quality assessment of dimensionality reduction: Rank-based criteria," *Neurocomputing*, vol. 72, no. 7, pp. 1431–1443, 2009.
- [13] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.
- [14] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural computation*, vol. 10, no. 5, pp. 1299–1319, 1998.
- [15] D. Francois, V. Wertz, and M. Verleysen, "The concentration of fractional distances," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 7, pp. 873–886, 2007.
- [16] J. A. Lee and M. Verleysen, "Shift-invariant similarities circumvent distance concentration in stochastic neighbor embedding and variants," *Procedia Computer Science*, vol. 4, pp. 538–547, 2011.
- [17] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski, "Information retrieval perspective to nonlinear dimensionality reduction for data visualization," *Journal of Machine Learning Research*, vol. 11, no. 13, pp. 451–490, 2010.
- [18] Z. Yang, I. King, Z. Xu, and E. Oja, "Heavy-tailed symmetric stochastic neighbor embedding," in *NIPS*, 2009, pp. 2169–2177.
- [19] C. de Bodt, D. Mulders, D. López-Sánchez, M. Verleysen, and J. A. Lee, "Class-aware t-SNE: cat-SNE," in *ESANN*, 2019, pp. 409–414.
- [20] K. Bunte, S. Haase, M. Biehl, and T. Villmann, "Stochastic neighbor embedding (SNE) for dimension reduction and visualization using arbitrary divergences," *Neurocomputing*, vol. 90, pp. 23–45, 2012.
- [21] J. A. Lee, E. Renard, G. Bernard, P. Dupont, and M. Verleysen, "Type 1 and 2 mixtures of kullback-leibler divergences as cost functions in dimensionality reduction based on similarity preservation," *Neurocomputing*, vol. 112, pp. 92–108, 2013.
- [22] C. de Bodt, D. Mulders, M. Verleysen, and J. A. Lee, "Nonlinear Dimensionality Reduction with Missing Data using Parametric Multiple Imputations," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 4, pp. 1166–1179, April 2019, DOI: 10.1109/TNNLS.2018.2861891.
- [23] M. Carreira-Perpinán, "The elastic embedding algorithm for dimensionality reduction," in *ICML*, vol. 10, 2010, pp. 167–174.
- [24] L. v. d. Maaten, "Fast optimization for t-SNE," in *NIPS Workshop on Challenges in Data Visualization*, 2010.
- [25] M. Vladymyrov and M. Carreira-Perpinán, "Entropic affinities: Properties and efficient numerical computation," in *ICML*, 2013, pp. 477–485.
- [26] —, "Partial-hessian strategies for fast learning of nonlinear embeddings," in *ICML*, vol. 29, 2012, pp. 345–352.
- [27] J. A. Lee, D. H. Peluffo-Ordóñez, and M. Verleysen, "Multi-scale similarities in stochastic neighbour embedding: Reducing dimensionality while preserving both local and global structure," *Neurocomputing*, vol. 169, pp. 246–261, 2015.
- [28] C. de Bodt, D. Mulders, M. Verleysen, and J. A. Lee, "Perplexity-free t-SNE and twice Student tt-SNE," in *ESANN*, 2018, pp. 123–128.
- [29] J. Peltonen and K. Georgatzis, "Efficient optimization for data visualization as an information retrieval task," in *MLSP*. Santander, Spain: IEEE, September 2012, pp. 1–6.
- [30] L. v. d. Maaten, "Learning a parametric embedding by preserving local structure," in *Artificial Intelligence and Statistics*, 2009, pp. 384–391.

- [31] F. Crecchi, C. de Bodt, M. Verleysen, J. A. Lee, and D. Bacciu, "Perplexity-free Parametric t-SNE," in *ESANN*, 2020, pp. 387–392.
- [32] L. v. d. Maaten, "Accelerating t-SNE using tree-based algorithms," *J. of machine learning research*, vol. 15, no. 1, pp. 3221–3245, 2014.
- [33] Z. Yang, J. Pelttonen, and S. Kaski, "Scalable optimization of neighbor embedding for visualization," in *ICML (2)*, 2013, pp. 127–135.
- [34] M. Vladymyrov and M. Carreira-Perpinan, "Linear-time training of nonlinear low-dimensional embeddings," in *Artificial Intelligence and Statistics*, 2014, pp. 968–977.
- [35] J. Tang, J. Liu, M. Zhang, and Q. Mei, "Visualizing large-scale and high-dimensional data," in *Proc. of the 25th int. conf. on WWW*. International World Wide Web Conferences Steering Committee, 2016, pp. 287–297.
- [36] C. Bouveyron, S. Girard, and C. Schmid, "High-dimensional data clustering," *COMPUT STAT DATA AN*, vol. 52, no. 1, pp. 502–519, 2007.
- [37] P. Yianilos, "Data structures and algorithms for nearest neighbor search in general metric spaces," in *SODA*, vol. 93, no. 194, 1993, pp. 311–321.
- [38] J. Barnes and P. Hut, "A hierarchical $O(N \log N)$ force-calculation algorithm," *nature*, vol. 324, no. 6096, pp. 446–449, 1986.
- [39] C. de Bodt, D. Mulders, M. Verleysen, and J. A. Lee, "Extensive assessment of Barnes-Hut t-SNE," in *ESANN*, 2018, pp. 135–140.
- [40] R. B. d'Agostino, "An omnibus test of normality for moderate and large size samples," *Biometrika*, vol. 58, no. 2, pp. 341–348, 1971.
- [41] J. P. Shaffer, "Multiple hypothesis testing," *Annual review of psychology*, vol. 46, no. 1, pp. 561–584, 1995.
- [42] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, 1995.
- [43] M. Lichman, "UCI Machine Learning repository," 2013. [Online]. Available: <http://archive.ics.uci.edu/ml>
- [44] P. Tüfekci, "Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods," *Int. J. Electrical Power & Energy Systems*, vol. 60, pp. 126–140, 2014.
- [45] H. Kaya, P. Tüfekci, and S. F. Gürgen, "Local and global learning methods for predicting power of a combined gas & steam turbine," in *Int. Conf. ETCEE*, 2012, pp. 13–18.
- [46] R. Madeo, C. Lima, and S. Peres, "Gesture unit segmentation using support vector machines: segmenting gestures from rest positions," in *Proc. SAC*. ACM, 2013, pp. 46–52.
- [47] J. Bridge, S. Holden, and L. Paulson, "Machine learning for first-order theorem proving: learning to select a good heuristic," *Journal of automated reasoning*, vol. 53, no. 2, pp. 141–172, 2014.
- [48] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror, "Result analysis of the nips 2003 feature selection challenge," in *NIPS*, 2004, pp. 545–552.
- [49] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decision Support Systems*, vol. 47, no. 4, pp. 547–553, 2009.
- [50] L. Chen and A. Buja, "Local multidimensional scaling for nonlinear dimension reduction, graph drawing, and proximity analysis," *J AM STAT ASSOC*, vol. 104, no. 485, pp. 209–219, 2009.
- [51] B. Mokbel, W. Lueks, A. Gisbrecht, and B. Hammer, "Visualizing the quality of dimensionality reduction," *Neurocomputing*, vol. 112, pp. 109–123, 2013.
- [52] C. Nugteren and V. Codreanu, "Clutune: A generic auto-tuner for opencv kernels," in *MCSoc*. IEEE, 2015, pp. 195–202.
- [53] R. Huerta, T. Mosquero, J. Fonollosa, N. Rulkov, and I. Rodríguez-Lujan, "Online decorrelation of humidity and temperature in chemical sensors for continuous monitoring," *Chemometrics and Intelligent Laboratory Systems*, vol. 157, pp. 169–176, 2016.
- [54] R. Zhang, F. Nie, and X. Li, "Self-Weighted Supervised Discriminative Feature Selection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 8, pp. 3913–3918, 2017.
- [55] —, "Semisupervised Learning With Parameter-Free Similarity of Label and Side Information," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 2, pp. 405–414, 2018.
- [56] —, "Regularized Class-Specific Subspace Classifier," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 11, pp. 2738–2747, 2016.
- [57] R. Zhang, F. Nie, M. Guo, X. Wei, and X. Li, "Joint Learning of Fuzzy k-Means and Nonnegative Spectral Clustering With Side Information," *IEEE Trans. Image Process.*, vol. 28, no. 5, pp. 2152–2162, 2018.
- [58] K. Zhang, X. Gao, D. Tao, and X. Li, "Single image super-resolution with multiscale similarity learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 10, pp. 1648–1659, 2013.
- [59] X. Gao, K. Zhang, D. Tao, and X. Li, "Image super-resolution with sparse neighbor embedding," *IEEE Trans. Image Process.*, vol. 21, no. 7, pp. 3194–3205, 2012.

- [60] Y. Liu, R. Zhang, F. Nie, X. Li, and C. Ding, "Supervised Dimensionality Reduction Methods via Recursive Regression," *IEEE Trans. Neural Netw. Learn. Syst.*, 2019, DOI: 10.1109/TNNLS.2019.2940088.



Cyril de Bodt was born in Brussels, Belgium, in 1993. He received the M.Sc. degree in Applied Sciences (Mathematical Engineering) in 2016 and the Ph.D. degree in Engineering Science and Technology in 2020, both from UCLouvain, Belgium. He is now a Postdoctoral Research Fellow in the Human Dynamics group of the Massachusetts Institute of Technology (MIT) Media Lab. His research interests include dimensionality reduction, machine learning, visualization, clustering, optimization and data mining.



Dounia Mulders was born in Liege, Belgium, in 1993. She received the M.Sc. degree in Applied Sciences (Mathematical Engineering) in 2016 and the Ph.D. degree in Engineering Science and Technology in 2020, both from UCLouvain, Belgium. As a Postdoctoral Research Fellow in the Fiete Lab of the Massachusetts Institute of Technology (MIT) Department of Brain and Cognitive Sciences, she studies the central mechanisms of pain perception in humans using signal processing and machine learning tools to analyze scalp and intracerebral EEG

recordings.



Michel Verleysen is a Professor of Machine Learning at UCLouvain. He has been an invited professor at the Swiss EPFL, the Université d'Evry Val d'Essonne, the Université ParisI-Panthéon-Sorbonne and the Université Paris Est. He is an Honorary Research Director of the Belgian FNRS, and the Dean of the Louvain School of Engineering. He is editor-in-chief of the Neural Processing Letters journal, chairman of the annual ESANN conference, past associate editor of the IEEE Trans. on Neural Networks journal, and member of the editorial board and program committee of several journals and conferences on neural networks and learning. He was the chairman of the IEEE Computational Intelligence Society Benelux chapter (2008–2010), and member of the executive board of the European Neural Networks Society (2005–2010). He is author of more than 250 scientific papers in international journals, books or peer-reviewed conferences. His research interests include machine learning, feature selection, dimensionality reduction, visualization, high-dimensional data analysis, self-organization, time-series forecasting and biomedical signal processing.



John Aldo Lee was born in 1976 in Brussels, Belgium. He received the M.Sc. degree in Computer Engineering in 1999 and the Ph.D. degree in Machine Learning in 2003, both from UCLouvain. His main interests are dimensionality reduction, clustering, vector quantization, and various aspects of image processing applied to cancer care. He is a member of the UCLouvain Machine Learning Group and a Senior Research Associate with the Belgian FNRS. Together with Michel Verleysen, he wrote a monography entitled 'Nonlinear Dimensionality Reduction' published by Springer in 2007. He is currently working in the UCLouvain center of Molecular Imaging, Radiotherapy, and Oncology (MIRO), where he develops specific image processing techniques to improve and speed up treatment planning in radiation oncology.