

Spaceland Embedding of Sparse Stochastic Graphs

Nikos Pitsianis^{*†} Alexandros-Stavros Iliopoulos[†] Dimitris Floros^{*} Xiaobai Sun[†]

^{*}Department of Electrical and Computer Engineering
Aristotle University of Thessaloniki
Thessaloniki 54124, Greece

[†]Department of Computer Science
Duke University
Durham, NC 27708, USA

Abstract—We introduce SG-t-SNE, a nonlinear method for embedding stochastic graphs/networks into d -dimensional spaces, $d = 1, 2, 3$, without requiring vertex features to reside in, or be transformed into, a metric space. Graphs/networks are relational data, prevalent in real-world applications. Graph embedding is fundamental to many graph analysis tasks, besides graph visualization. SG-t-SNE follows and builds upon the core principle of t-SNE, which is a widely used method for visualizing high-dimensional data. We also introduce SG-t-SNE-II, a high-performance software for rapid d -dimensional embedding of large, sparse, stochastic graphs on personal computers with superior efficiency. It empowers SG-t-SNE with modern computing techniques exploiting matrix structures in tandem with memory architectures. We present elucidating graph embedding results with several synthetic graphs and real-world networks in this paper and its Supplementary Material.¹

I. INTRODUCTION

Big and sparse graphs/networks are relational data arising in various research fields and real-world applications. Such data have great diversity in attributes, contents and properties, such as biological networks, social, friend or co-author networks, commercial product graphs, food webs, ecological networks, telecommunication networks, information networks, transportation networks, word co-occurrence graphs, and image networks [1]–[15]. Graph/network analysis plays an important role in data analysis.

A graph $\mathcal{G}(V, E)$ has a set V of vertices and a set E of edges. The vertices (nodes) are an abstraction of entities or objects that have concrete forms or possess particular attributes in a real-world context. The vertices may stand for molecules, proteins, neuron cells, species, products, customers, words, documents, signals or time series, images or pixels. An edge (link) connecting two vertices represents a certain relationship, an interaction, or proximity between them. Additional graph information may include vertex attributes and edge weights.

Fundamental to many graph analysis tasks, graph embedding renders a mapping from vertices to their coded (feature) vectors in a code space (the embedding space) with code length L . The mapping is also known as vertex embedding. It is subject to one or more conditions to preserve or reconstruct certain graph properties in the embedding space. For example, if the edge weights are pairwise (geometric or geodesic) distances between nodes, one wishes to preserve the pairwise distance relationships as much as possible.

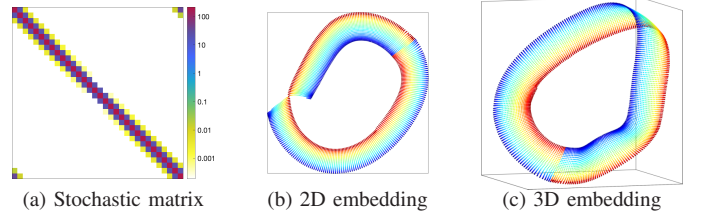


Fig. 1: Essential neighbor and structural connections in a k NN graph for a Möbius strip lattice, with $n = 8,192$ nodes on a 256×32 lattice and $k = 150$, are obscured or crumbled by a 2D embedding, yet principally captured by a 3D embedding. (a) The stochastic matrix of the k NN graph (by Euclidean distance) is displayed in a 32×32 pixel array. Each pixel represents a 256×256 submatrix, the pixel value (color-coded) is the sum of the submatrix elements. (b)–(c) The 2D and 3D embeddings by SG-t-SNE with $\lambda = 100$, taking 1,000 iterations, including 250 early exaggeration ones with $\alpha = 12$. The initial coordinates are drawn randomly from a uniform distribution. (It is shown in Supplementary Material that the embedding of the same graph by t-SNE is more susceptible to distortion, within the feasible perplexity range.)

The feature/code vector length L is chosen sufficiently high for large graphs/networks and multiple analytical tasks. For example, $L = 300$ in word embedding with `word2vec` [16] and `GloVe` [17]. Various graph analysis tasks are subsequently carried out in an embedding space. They include (semi-, un-) supervised classification or stratification, abnormality detection, noise reduction, propagation patterns, content recommendation, and prediction. In recent years, numerous types of high-dimensional graph embedding techniques are tried out with neural networks [18]–[28].

High-dimensional graph embedding is often followed and accompanied by dimension reduction and low-dimensional embedding for multiple purposes: (i) making visual assessment, inspection, and summary of the analysis results; (ii) facilitating interactive exploration; and (iii) discovering connections to the original, interpretable attributes. There is a plethora of low-dimensional graph embedding techniques [29]–[32]. Among the notable ones are the original stochastic neighbor embedding (SNE) method by Hinton and Roweis [33] and the remarkable variant t-distributed SNE (t-SNE) by van der Maaten and Hinton [34]. In particular, t-SNE is widely used or customized for nonlinear dimensionality reduction and data visualization [35]–[38]. It has successfully assisted scientific discoveries, as reported in numerous articles in *Nature* and *Science* magazines [39]–[41].

Previous t-SNE algorithms and software, however, are limited in two aspects: (i) They require that the data points be

¹Supplementary Material is at <http://t-sne-pi.cs.duke.edu>.

in a geometric space and the associated graph (internally generated) be the k -nearest neighbor (k NN) graph, regular with a constant degree k . In many real-world networks, the vertex attributes do not readily reside in a metric space; the vertex degrees vary greatly, far from constant. (ii) The software is limited in practical use either to small graphs/networks or to one or two dimensional embeddings. We demonstrate in Fig. 1 that three-dimensional (3D) embedding has greater capacity of preserving local and global connectivities. We show further in Section III that three-dimensional (3D) embedding elucidates more structural information, which may be valuable to analysis and discovery.

We make two major contributions in the present work. First, we introduce a novel nonlinear approach, SG-t-SNE, for embedding large, sparse, stochastic graphs into d -dimensional spaces, $d = 1, 2, 3$, without requiring vertex features to reside in, or be transformed into, a geometric space. SG-t-SNE follows and builds upon the core principle of SNE. It extends the use of t-SNE to the entire realm of stochastic graphs, including but no longer limited to distance-based k NN graphs. We equip SG-t-SNE with a parametrized rescaling mechanism in order to explore the graph sparsity. In the special case of k NN graph embedding, the results by the conventional t-SNE with k neighbors can be well matched or outmatched by SG-t-SNE with k' neighbors, with k' much smaller than k .

Second, we introduce the novel components in a high-performance software, SG-t-SNE-II [42]. The novelty lies in how we explore and utilize data locality and parallelism in computation with large sparse matrices and compressible matrices. Two-dimensional (2D) graph embedding is sped up multiple times, up to $5\times$, by the new software, in comparison to the best previously existing t-SNE software [38]. Furthermore, with superior efficiency, SG-t-SNE-II enables practical 3D embedding of large sparse graphs on personal computers. We also present experimental results in this paper on several real-world graphs/networks: two gene-cell co-expression networks, a social network created by Google, and an Amazon product network. More experimental results and steerable views of 3D embeddings are available in the Supplementary Material.

II. EMBEDDING SPARSE STOCHASTIC GRAPHS

We introduce SG-t-SNE. We apply and extend the essential principle of SNE/t-SNE to the entire realm of sparse stochastic graphs. A stochastic graph/network $\mathcal{G} = (V, E, \mathbf{P}_c)$ has $n = |V|$ vertices and $m = |E|$ stochastically weighted edges. The stochastic weights are specified by matrix $\mathbf{P}_c = [p_{j|i}]$. At each vertex i , $\{p_{j|i}\}_{j=1}^n$ is the probability distribution conditional on i . The stochastic interpretation is specific to the particular graph/network analysis problem under consideration. We denote by $\mathbf{A} = [a_{ij}]$ the binary-valued adjacency matrix (the sparsity mask) of the graph. We assume that \mathcal{G} has no isolated (0-degree) vertices. We are especially concerned with large and sparse real-world graphs. Each typically has a large vertex set and sparse connections, i.e., $n = |V|$ is large and $m = |E| = O(n)$. The graph is not necessarily

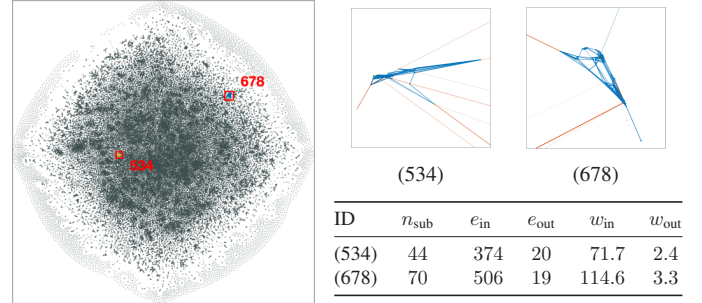


Fig. 2: Two-dimensional embedding of an Amazon product network, with $n = 334,863$ products and $m = 1,851,744$ edges [5]. Each edge connects two products that are frequently purchased together. **Left:** Vertex embedding. The highlighted vertices are two disjoint product communities with IDs 534 and 678. The embedding is by SG-t-SNE, with $\lambda = 10$, taking 8,000 iterations, including 2,000 early exaggeration iterations with $\alpha = 12$. The initial coordinates are drawn randomly from a Gaussian distribution. The graph is inadmissible to the conventional t-SNE. **Right:** The subgraphs of the two communities with intra-community edges in blue and external edges in red. The table details each of the communities in numbers of nodes, internal edges and external edges, and the sum of weights over the internal and external edges.

regular with constant degree, or with vertex attribute vectors in a metric space.

A. An investigative review of t-SNE

We describe first the essence of t-SNE. Provided with a graph \mathcal{G} with stochastic weights at each vertex, described by a stochastic matrix $\mathbf{P}_c = [p_{j|i}]$, and given a prescribed modest dimensionality d , t-SNE generates vertex embedding coordinate vectors $\mathcal{Y} = \{\mathbf{y}_i\}_{i=1}^n$ in \mathbb{R}^d , regulated by the t-distribution,

$$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}}, \quad i \neq j. \quad (1)$$

Define $\mathbf{Q}(\mathcal{Y}) = [q_{ij}]$. The embedding arrives at the optimal matching to the joint distribution $\mathbf{P} = (\mathbf{P}_c + \mathbf{P}_c^T)/(2n)$ in the sense that

$$\mathcal{Y}^* = \arg \min_{\mathcal{Y}} \text{KL}(\mathbf{P} \parallel \mathbf{Q}(\mathcal{Y})), \quad (2)$$

where KL denotes the Kullback-Leibler divergence.

The standard version of t-SNE, however, is developed, used, viewed and reviewed as a nonlinear dimension reduction algorithm, due primarily to the following particular interface to the distribution matching objective (2). The input is a set of data points in terms of feature vectors $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$ in an L -dimensional space \mathbb{R}^L endowed with a distance function $d(\cdot, \cdot)$. The feature vector length L is (much) larger than the embedding dimension d . The algorithm locates the k -nearest neighbors (k NN) of each vertex, and integrates them into the k NN graph, which is regular (with constant degree k). Denote by \mathbf{D} the matrix of squared pairwise distances, $\mathbf{D} = [d_{ij}^2] = [d^2(\mathbf{x}_i, \mathbf{x}_j)]$. Then, the distance-weighted k NN matrix $\mathbf{D} \odot \mathbf{A}$ (Hadamard product of \mathbf{D} and \mathbf{A}) is converted to a stochastic matrix \mathbf{P}_c by

$$p_{j|i}(\sigma_i) = \frac{a_{ij} \exp(-d_{ij}^2/2\sigma_i^2)}{\sum_l a_{il} \exp(-d_{il}^2/2\sigma_i^2)}. \quad (3)$$

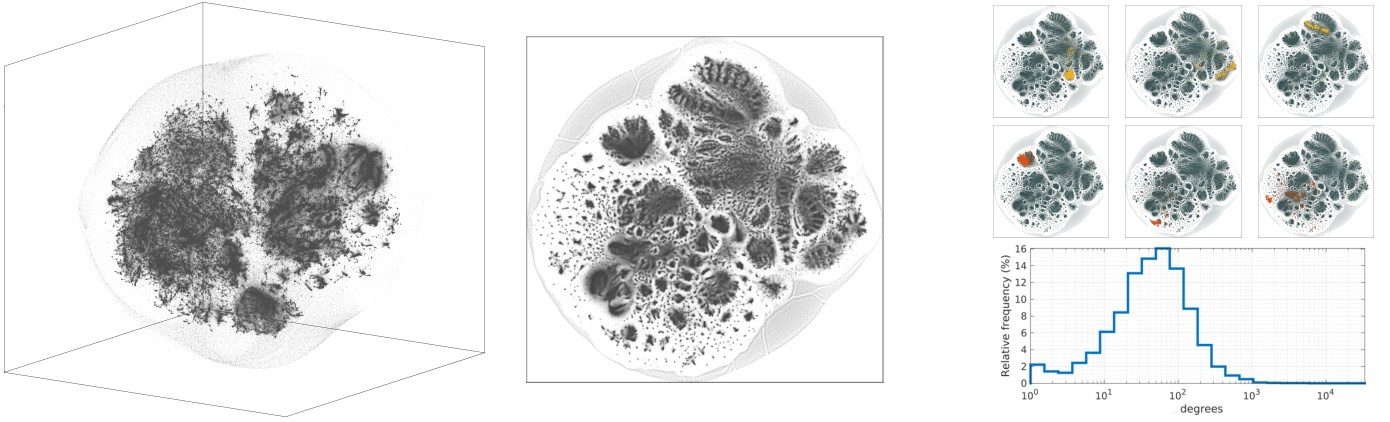


Fig. 3: Spaceland (3D) and Flatland (2D) embedding of the social network `orkut` as of 2012 [5]. The network has $n = 3,072,441$ user nodes and $m = 237,442,607$ friendship links. The degree varies from 1 (leaf nodes) to 33,313 (hub node). The graph is inadmissible to the conventional t-SNE. The embedding is by SG-t-SNE with $\lambda = 10$, taking 8,000 iterations, with initial coordinates drawn randomly from a Gaussian distribution. The 2D embedding takes only 50 minutes on the Intel Xeon E5 (Table I). **Left–Middle:** The embeddings display community structures. The leaf nodes (67,767 of them) are in the halo-like peripheral area. The rest can be roughly split into two hemispherical regions, which may likely correspond to the largest user populations in two geographical areas; one is in and around Brazil, the other is in and around India and some other countries in Asia. **Right:** The scatter diagrams in the thumbnails show six identified communities with IDs 107, 325, 688 (top row) and 2196, 3208, 3405 (bottom row). The top ones reside in one hemispherical region, the bottom ones in the other. This pattern indicates the impact of geographical regions, cultures, and societies on social network structures. The histogram shows the degree distribution in logarithmic scale, of the friendship links over all users.

At every vertex i , the Gaussian parameter σ_i is determined by

$$-\sum_j a_{ij} p_{j|i}(\sigma_i) \log(p_{j|i}(\sigma_i)) = \log(u), \quad (4)$$

where u is a prescribed parameter interpreted as the perplexity, i.e., $\log(u)$ is the entropy. In the sense of equation (4), the conditional distributions at all vertices are equalized.

We make a couple of remarks on the relationship between u and k . Commonly, the heuristic rule $k = \lceil 3u \rceil$ is used. Veritably, the condition $0 < u < k$ is necessary for the solutions to (4) to exist at all vertices. With a larger perplexity u , one would increase k proportionally to get a better embedding in the sense of (2), at the expense of a denser graph, larger memory requirement, and longer execution time.

To unleash the greater potential of t-SNE, we must unravel two restraining knots. One is the coupling between distance and stochastics by (3). The other is the pairing between vertex degree and perplexity by (4).

B. Advantages of admitting stochastic graphs

The stochastic k NN graphs generated by (3) and (4) are only a subset of stochastic graphs. While they play a key role for t-SNE being seen and used as a tool for nonlinear dimensionality reduction and data visualization, they obscure the essential principle of t-SNE in several ways. (i) Real-world graph data stem from various sources [1]–[12]. Often, the native and interpretable descriptors of the entity vertices are not readily in a metric space. The descriptors may contain both numerical and categorical data attributes. For such data, t-SNE may be preceded by a high-dimensional metric-space embedding, using machine learning tools (e.g., neural networks) and techniques [16], [17], such as in parametrized t-SNE [43], at a very high computational cost. (ii) In the case where the feature descriptors are in a metric space, constructing the k NN

graph is computationally expensive. Randomized approximate methods are used to reduce the asymptotic complexity to $O(n \log n)$ [44], [45]. However, such methods do not leverage domain-specific spaces of random variables [46]. (iii) The k NN graph and the outcome of t-SNE vary with the choice of vector space, distance function, and k NN search methods. (iv) The stochastic recasting of a k NN graph is exclusively confined in conventional t-SNE to the particular form (4), not inclusive of other probabilistic models. The above limitations are common to many t-SNE variants [34]–[41] and other multi-dimensional scaling (MDS) algorithms [29]–[31]. There exist nonlinear dimension reduction methods that attempt to circumvent the k NN search with certain sampling strategies [32].

Stochastic graphs are found or generated in a multitude of other ways. We give two examples. First, for high-security information networks, the vertex features are not publicly accessible and their links are stochastically encrypted [47]. Second, network data are often available only with binary-valued links. Nonetheless, the stochastic weights of the links may be recovered to a significant extent from the local and global connection topologies. We use in this paper two such stochastic graphs from a recent study [48]: one is of a social network, the other of an e-commerce network. In addition, real-world graphs are typically irregular, with high-degree hub nodes and 1-degree leaf nodes. See, for example, the degree distribution of the `orkut` network in Fig. 3.

We effectively distinguish and separate, in principled concepts as well as in practical operations, stochastic graph modeling from graph embedding. This disentangling allows for exploring the potential in each and integrating the best from each. We explore the potential in d -dimensional graph embedding of a stochastic graph for graph/network analysis.

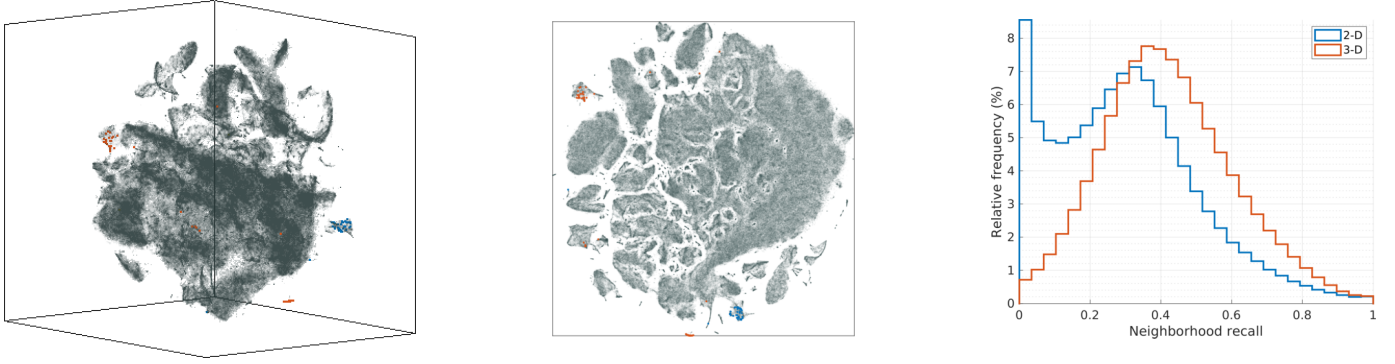


Fig. 4: Spaceland (3D) and Flatland (2D) embedding of the k NN graph ($k = 90$) associated with $n = 1,306,127$ RNA sequences of E18 mouse brain cells [49]. The k -nearest neighbors ($k = 90$) are located in the space of the 50 principle components of the top 1,000 variable genes [50]. **Left-Middle:** The embeddings are by SG-t-SNE-II with perplexity $u = 30$, taking 8,000 iterations, including 2,000 early exaggeration iterations with $\alpha = 12$. The initial coordinates are drawn randomly from a uniform distribution. Two subpopulations are color-annotated: GABAergic subtype (SNCG, SLC17A8) in blue and VLMC subtype (SPP1, COL15A1) in red. They are clearly separated from the rest in 3D embedding. **Right:** Histograms of the stochastic k -neighbors recall values over all vertices; recall(i) = $\sum_j p_{j|i} b_{ij}$, where $\mathbf{B} = [b_{ij}]$ is the adjacency matrix of the k NN graph of the embedding points $\{\mathbf{y}_i\}$. The stochastic weights $p_{j|i}$ are common to both 2D and 3D embeddings. The histogram for the 3D embedding (red) shows relatively higher recall scores than the 2D embedding (blue). The quantitative comparison is consistent with that by visual inspection.

C. Introduction of SG-t-SNE

Our algorithm SG-t-SNE follows the essential principle of t-SNE and removes the barriers in its conventional version. SG-t-SNE admits any stochastic graph $\mathcal{G} = (V, E, \mathbf{P}_c)$, including stochastic k NN graphs. It is capable of not only adapting to the sparse pattern in the input graph, but also exploiting the pattern for better distribution matching and vertex embedding with a nonlinear rescaling mechanism.

Specifically, we generate from \mathbf{P}_c a family of parametrized stochastic matrices, $\mathbf{P}_c(\lambda)$. The role of the rescaling mechanism is to explore and exploit the potential in distribution matching (2) from the \mathbf{P} side, given the fixed t-distribution (a Cauchy distribution) on the \mathbf{Q} side. Unlike the perplexity u in t-SNE, the parameter λ for SG-t-SNE exploits the sparsity without imposing any constraint. In the special case of a weighted k NN graph, λ is untangled from k .

We describe the mechanism and effect of nonlinear rescaling with a single parameter λ . Let $\phi(\cdot)$ be a rescaling kernel function that is monotonically increasing over \mathbb{R}_+ , with $\phi(0) \geq 0$. For any $\lambda > 0$, SG-t-SNE generates $\mathbf{P}_c(\lambda) = [p_{j|i}(\lambda)]$ as follows. We determine the rescaling exponent γ_i at each vertex i by the equation

$$\sum_j a_{ij} \phi(p_{j|i}^{\gamma_i}) = \lambda, \quad (5)$$

and then reshape and rescale the conditional probability

$$p_{j|i}(\lambda) = \frac{a_{ij} \phi(p_{j|i}^{\gamma_i})}{\lambda}, \quad (6)$$

where \mathbf{A} is the binary-valued adjacency matrix (the sparsity mask) of graph \mathcal{G} , which is invariant to any change in ϕ or λ . Unlike (4), the rescaling solutions exist unconditionally.

The rescaling mechanism introduces an additional degree of freedom to exploit the graph sparsity, without restriction on the sparsity pattern. In this paper, we use the identity function $\phi(x) = x$ as the rescaling kernel for all experiments. At $\lambda = 1$,

we get $\gamma_i = 1$ and $\mathbf{P}_c(1) = \mathbf{P}_c$, the input matrix. At an integer value $\lambda = k > 1$, if node ℓ has degree k , then $\gamma_\ell = 0$ and $p_{j|\ell} = 1/k$. In the special case of a regular graph with degree k , if we set $\lambda = k$, then $\gamma_i = 0$ at all vertices, i.e., $\mathbf{P}_c(k) = \mathbf{A}/k$. In general, at $\lambda \neq 1$, the ratio between every nonzero element to the peak element ($p_{j|i}/\max_\ell p_{\ell|i}$) is changed to $(p_{j|i}/\max_\ell p_{\ell|i})^{\gamma_i}$. The weight distribution for each vertex is reshaped, according to the rescaling equation (5), in adaptation to local connections.

In SG-t-SNE, only two changes are made from t-SNE. One is at the interface. SG-t-SNE admits any stochastic graph, including but not limited to stochastic k NN graphs. The other is the introduction of the nonlinear scaling equation (5). Equation (4) can still be used as a particular mechanism to convert k NN graphs into stochastic ones.

We show in Figs. 2 and 3 the embeddings of two real-world networks, enabled by SG-t-SNE. Each exhibits a distinctive cluster structure. We show in Fig. 5 that even with k NN graphs at input, the embedding by t-SNE at $k = 150$, $u = 50$ is well matched or outmatched by that with SG-t-SNE at $k = 30$, $\lambda = 80$, on a sparser matrix and at a lower computation cost. We recommend multiple views at various values of λ .

III. RAPID SPACELAND EMBEDDING

The practical use of t-SNE was largely confined to Flatland (2D) or Lineland (1D) embedding, despite the prototype implementation by van der Maaten² supporting embedding of several dimensions. We advocate Spaceland (3D or higher) embedding and endorse “the enlargement of the imagination” [51]. Among abundant evidence, we illustrate in Figs. 1, 3, and 4 the extended capacity to capture and reveal multi-fold connections between (overlapping) subpopulations, which were not previously seen with network data visualization.

There were multiple obstacles to Spaceland embedding. The journey of t-SNE is marked by continuous efforts and progress

²<https://lvdmaaten.github.io/tsne>

in reducing the arithmetic complexity and execution time of searching for an optimal embedding. We present SG-t-SNE-II, a software renovation. It enables rapid Spaceland embedding of large, sparse graphs on desktop or laptop computers, which are available and affordable to researchers by and large.

The vertex embedding coordinate vectors $\mathcal{Y} = \{\mathbf{y}_i\}_{i=1}^n$ are determined in t-SNE by applying the gradient descent method to the minimization problem (2). The computation per iteration step is dominated by the calculation of the gradient, which is reformulated into two terms by van der Maaten [35],

$$\frac{\partial(\text{KL})}{\partial \mathbf{y}_i} = \frac{4}{Z} \underbrace{\sum_{i \neq j} p_{ij} q_{ij} (\mathbf{y}_i - \mathbf{y}_j)}_{\text{attraction term}} - \frac{4}{Z} \underbrace{\sum_{i \neq j} q_{ij}^2 (\mathbf{y}_i - \mathbf{y}_j)}_{\text{repulsion term}}, \quad (7)$$

where $Z = \sum_{i,j} q_{ij}$. The terms are named by analogy to the attractive and repulsive forces in molecular dynamics simulations. The attraction term can be cast as the sum of multiple matrix-vector products with sparse matrix $\mathbf{PQ} = [p_{ij}q_{ij}]$ and d coordinate vectors, one along each embedding dimension. Similarly, the repulsion term can be cast as the sum of multiple matrix-vector products with dense matrix $\mathbf{QQ} = [q_{ij}q_{ij}]$ and d coordinate vectors. The calculation of the attraction term takes $O(dm)$ arithmetic operations, where $m = |E|$ is the number of edges. A naive calculation of the repulsion term, however, takes $O(dn^2)$ arithmetic operations. This quadratic complexity would prohibit the use of t-SNE on large graphs.

A. Fast approximations to the repulsion term

Two existing t-SNE variants use approximate algorithms to calculate the repulsion term. They reduce the quadratic complexity to $O(c_\epsilon^d n \log n)$, where $c_\epsilon > 2$ increases reciprocally with ϵ , which is a prescribed approximation error tolerance. The specific value of c_ϵ in the prefactor c_ϵ^d depends also on the choice of approximation algorithm and its parameters. The first approximation algorithm, by van der Maaten, adopts the Barnes-Hut (BH) algorithm [52]. We refer to this version as t-SNE-BH. An alternative approximation algorithm, named FIt-SNE, emerged last year [38].

We describe the fundamental concepts and distinctive approaches behind the two approximation algorithms, instead of reviewing each in detail. The matrix-vector multiplication with \mathbf{QQ} is a convolution with the Cauchy kernel on non-equispaced, scattered data points $\{\mathbf{y}_i\}_{i=1}^n$. We refer to such a convolution as “nuConv”. A convolution may have broad or narrow (windowed) support. A narrow-support convolution has complexity $O(w^d n)$, where w is the support window size; the corresponding matrix is sparse, spatially banded. The nuConv with \mathbf{QQ} is of broad support. Fortunately, the structure of \mathbf{QQ} can be exploited.

There are two renowned and distinctive families of algorithms for fast nuConv of broad support. One family is based on the fast multipole method (FMM), with asymptotic arithmetic complexity $O(n)$ [53]–[55]. FMM splits a convolution of broad support into $O(\log n)$ convolutions of narrow support at multiple spatial levels. Its extended family includes the

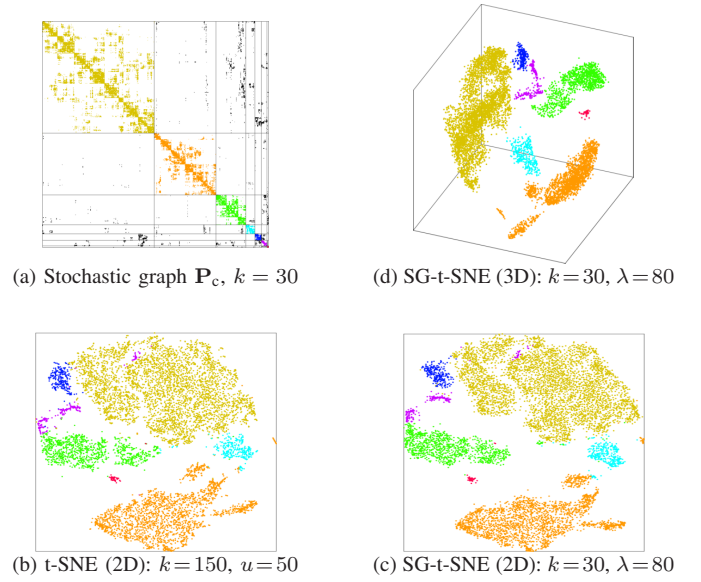


Fig. 5: Multiple views of subpopulations of $n = 8,381$ peripheral blood mononuclear cells (PBMC-8k) [50]. The subpopulations (color-coded) were found by the SD-DP classification analysis [57]. (a) The stochastic k NN graph \mathbf{P}_c ($k = 30$), with rows/columns permuted so that cells in the same subpopulation are placed close together. Each pixel corresponds to a 16×16 matrix block. (b)–(c) It is compelling that the embedding by SG-t-SNE on a sparse graph is comparable to that by t-SNE on a graph $5 \times$ as dense. (d) Rapid Spaceland (3D) embedding is enabled by SG-t-SNE-II. All three embeddings take 1,000 iterations, including 250 early exaggeration iterations with $\alpha = 12$. The initial embedding coordinates are drawn randomly from a uniform distribution.

Barnes-Hut algorithm, with complexity $O(n \log n)$ [52]. The other family is related to non-uniform fast Fourier transforms (nuFFT), with arithmetic complexity $O(n \log n)$ [21], [56]. FIt-SNE takes the latter approach. Its software implementation is available for 1D or 2D embedding only.³

As the arithmetic complexity for calculating the repulsion term (with dense \mathbf{QQ}) is reduced, the execution time for 2D embedding with FIt-SNE or t-SNE-BH becomes dominated by the computation of the attraction term (with sparse \mathbf{PQ}); see Fig. 6. Both FIt-SNE and t-SNE-BH face a steep rise in execution time for 3D embedding. FIt-SNE also suffers from exponential growth in memory requirement due to data padding for converting aperiodic convolutions to periodic.

B. Introduction of SG-t-SNE-II

With SG-t-SNE-II, we expedite the entire gradient calculation of (7), not only on each of the interaction terms but also on data relocations between the terms. We render Spaceland (3D) embedding in shorter time than Flatland (2D) embedding with FIt-SNE or t-SNE-BH. We overcome challenges at multiple algorithmic stages with innovative approaches. We utilize the matrix structure and the memory architecture in concert.

Fast computation with sparse \mathbf{PQ} : The irregular sparsity pattern of \mathbf{PQ} invokes irregular memory accesses, which incur long latency on modern computers with hierarchical

³<https://github.com/KlugerLab/FIt-SNE>

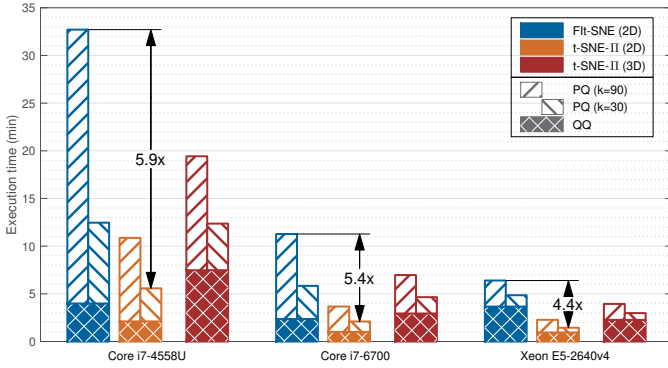


Fig. 6: Multiple comparisons in execution time for embedding of k NN graphs, $k \in \{30, 90\}$, with $n = 1,306,127$ nodes as single-cell RNA sequences of E18 mouse brain cells [49]. Each embedding takes 1,000 iterations and maintains an approximation error below the same tolerance (10^{-6}). (i) In total time, with the same sparsity parameter k , 3D embeddings by SG-t-SNE-II take less time to finish $1.5\times$ operations in comparison to 2D embeddings by t-SNE, on all three multi-core computers (Table I). (ii) In memory usage, SG-t-SNE-II does not augment grid size and thereby admits larger graphs. (iii) The computation with sparse matrix PQ dominates the t-SNE time on machines with lower bandwidth. SG-t-SNE-II reduces the PQ time further by embedding a sparser graph ($k = 30$) with results comparable to that by t-SNE on a graph $3\times$ as dense ($k = 90$). Additional notes: the SG-t-SNE-II code is to be optimized; a version utilizing graphics cards is in development.

memories [58]. The execution time of Flt-SNE is evidently dominated by the computation with PQ at $k = 90$ and $u = 30$; see Fig. 6. The computation speed is limited by the memory bandwidth and how it is utilized.

We exploit the fact that matrix PQ is sparse and inherits the fixed sparsity pattern of matrix P , despite the iterate changes of Q . We permute P such that rows and columns with similar nonzero patterns are placed closer together. The permuted matrix becomes block-sparse with denser blocks, not necessarily block diagonal. A denser block leads to better data locality in memory reads and writes. There are several efficient approaches to clustering rows and columns, see [57] and references therein. The overhead for permuting P once is amortized across all iterations, and is well paid off, see Fig. 6. In implementation, we adopt the sparse storage format and matrix computation routines provided in the Compressed Sparse Blocks (CSB) library [59], [60].

Fast computation with dense QQ : We take the spectral approach for computation with QQ , similar to Flt-SNE. We set an auxiliary grid in the embedding domain. The nuConv with QQ is then factored into three consecutive convolutional operations: S2G, G2G, and G2S. The S2G and G2S convolutions translate $\{y_i\}$ to their neighboring grid points and vice versa. G2S can be instrumented as a local interpolation; and S2G as the transpose. Both are of narrow support, taking $O(w^d n)$ arithmetic operations, where w is the local window size per side. G2G is an aperiodic convolution across the grid.

We make algorithmic innovations for each convolution factor. Instead of embedding G2G into a circulant convolution via augmenting data size with zero padding, we reformulate the aperiodic convolution as the in-place combination of two periodic convolutions, eliminating the need to augment the

TABLE I: Architecture specification of three multi-core computers used for the embedding experiments. Memory bandwidth is measured with the parallel STREAM copy benchmark [61].

CPU	Clock (GHz)	Cores	L1 (KiB) per-core	L2 (KiB) shared	L3 (MiB) shared	RAM (GiB)	BW (GiB/s)
Intel Core i7-4558U	2.80	2	64	256	4	8	10.5
Intel Core i7-6700	3.40	4	32	256	8	32	19.9
Intel Xeon E5-2640v4*	2.40	10	32	256	25	256	36.7

*2 sockets (non-uniform memory access—NUMA)

grid size and increase memory usage by a factor of 2^d . This memory saving alone allows embedding of a larger graph on a laptop or desktop computer. The execution time for S2G and G2S is dominated by latency in irregular data access, which is a typical challenge in sparse matrix calculation. Prior to S2G, the scattered data points $\{y_i\}$ are binned into the grid cells. This data partition and binning process significantly improves the data locality and parallelism scope in S2G, and thereby shortens the S2G time. The same holds true for G2S. In other words, we make another novel use of the grid for exploiting data locality and parallelism.

Fast data translocation: The orderings in accessing $\{y_i\}$ between the PQ and QQ procedures and between the translation operations within QQ differ from each other. In addition, the data set $\{y_i\}$ undergoes dynamic changes in the iterative search for the best matching (2). For the PQ term, $\{y_i\}$ are accessed according to the permuted rows and columns of P . For the QQ term, the embedding data points are accessed by their partition over grid cells. At each iteration step, we translocate the data $\{y_i\}$ back and forth between the two terms, in multiple stages. In essence, a point-to-point cross permutation is factored into layers of block-to-block permutations, according to the memory hierarchy.

Data permutations, although unaccounted for in arithmetic complexity, play a key role in accelerating sparse or compressible matrix computations on modern computers [62], [63]. We use II in the software name SG-t-SNE-II to signify their role.

IV. ADDITIONAL REMARKS

The embeddings of synthetic and real-world graphs/networks with SG-t-SNE-II show characteristic, beautiful and elucidating structures. The `orkut` embedding reveals overlapping and non-overlapping regions between social communities. The embedding of the Amazon product network can serve market analysis for assessing previous claimed product clusters and making new recommendations or predictions. With 3D embeddings enabled, we gain multiple vantage points, free of 2D confinement, to uncover previously unseen connections and separations, as shown with the biological networks PBMCs and E18-MBCs.

Acknowledgments: We thank Tiancheng Liu for providing us with several irregular stochastic graphs for the study. We thank Pantazis Vouzaxakis and Xenofon Theodoridis for assistance with the experiments. We thank the anonymous reviewers for their valuable comments.

REFERENCES

- [1] R. Albert, H. Jeong, and A.-L. Barabási, "Diameter of the world-wide web," *Nature*, vol. 401, no. 6749, pp. 130–131, 1999.
- [2] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.
- [3] J. Kunegis, "KONECT: The Koblenz network collection," in *Proceedings of the 22nd International Conference on World Wide Web*, 2013, pp. 1343–1350.
- [4] H. Kwak, C. Lee, H. Park, and S. Moon, "What is Twitter, a social network or a news media?" in *Proceedings of the 19th International Conference on World Wide Web*, 2010, pp. 591–600.
- [5] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowledge and Information Systems*, vol. 42, no. 1, pp. 181–213, 2015.
- [6] M. Cha, H. Haddadi, F. Benevenuto, and K. P. Gummadi, "Measuring user influence in Twitter: The million follower fallacy," in *Proceedings of the 4th International AAAI Conference on Weblogs and Social Media*, 2010.
- [7] M. E. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Physical Review E*, vol. 74, no. 3, 2006.
- [8] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "DBpedia: A nucleus for a web of open data," in *The Semantic Web*, 2007, pp. 722–735.
- [9] H. Yu, P. Braun, M. A. Yildirim, I. Lemmens, K. Venkatesan, J. Sahalie, T. Hirozane-Kishikawa, F. Gebreab, N. Li, N. Simonis, T. Hao, J.-F. Rual, A. Dricot, A. Vazquez, R. R. Murray, C. Simon, L. Tardivo, S. Tam, N. Szvikapa, C. Fan, A.-S. de Smet, A. Motyl, M. E. Hudson, J. Park, X. Xin, M. E. Cusick, T. Moore, C. Boone, M. Snyder, F. P. Roth, A.-L. Barabási, J. Tavernier, D. E. Hill, and M. Vidal, "High-quality binary protein interaction map of the yeast interactome network," *Science*, vol. 322, no. 5898, pp. 104–110, 2008.
- [10] S. Redner, "Citation statistics from more than a century of Physical Review," *Physics Today*, vol. 58, no. 6, pp. 49–54, 2005.
- [11] W. W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of Anthropological Research*, vol. 33, no. 4, pp. 452–473, 1977.
- [12] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
- [13] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.
- [14] G. A. Miller, "WordNet: A lexical database for English," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.
- [15] C. Fellbaum, Ed., *WordNet: An Electronic Lexical Database*. Cambridge, MA, USA: MIT press, 1998.
- [16] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems*, vol. 26, 2013, pp. 3111–3119.
- [17] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in *Empirical Methods in Natural Language Processing*, 2014, pp. 1532–1543.
- [18] M. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AIChE Journal*, vol. 37, no. 2, pp. 233–243, 1991.
- [19] B. Fritzke, "A growing neural gas network learns topologies," in *Advances in Neural Information Processing Systems*, vol. 7, 1995, pp. 625–632.
- [20] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [21] W. Wang, Y. Huang, Y. Wang, and L. Wang, "Generalized autoencoder: A neural network framework for dimensionality reduction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 490–497.
- [22] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 1067–1077.
- [23] J. H. Levine, E. F. Simonds, S. C. Bendall, K. L. Davis, E. D. Amir, M. D. Tadmor, O. Litvin, H. G. Fienberg, A. Jager, E. R. Zunder, R. Finck, A. L. Gedman, I. Radtke, J. R. Downing, D. Pe'er, and G. P. Nolan, "Data-driven phenotypic dissection of AML reveals progenitor-like cells that correlate with prognosis," *Cell*, vol. 162, no. 1, pp. 184–197, 2015.
- [24] A. M. Klein, L. Mazutis, I. Akartuna, N. Tallapragada, A. Veres, V. Li, L. Peshkin, D. A. Weitz, and M. W. Kirschner, "Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells," *Cell*, vol. 161, no. 5, pp. 1187–1201, 2015.
- [25] I. Tirosh, B. Izar, S. M. Prakadan, M. H. Wadsworth, D. Treacy, J. J. Trombetta, A. Rotem, C. Rodman, C. Lian, G. Murphy, M. Fallahi-Sichani, K. Dutton-Regester, J.-R. Lin, O. Cohen, P. Shah, D. Lu, A. S. Genshaft, T. K. Hughes, C. G. K. Ziegler, S. W. Kazer, A. Gaillard, K. E. Kolb, A.-C. Villani, C. M. Johannessen, A. Y. Andreev, E. M. Van Allen, M. Bertagnoli, P. K. Sorger, R. J. Sullivan, K. T. Flaherty, D. T. Frederick, J. Jané-Valbuena, C. H. Yoon, O. Rozenblatt-Rosen, A. K. Shalek, A. Regev, and L. A. Garraway, "Dissecting the multicellular ecosystem of metastatic melanoma by single-cell RNA-seq," *Science*, vol. 352, no. 6282, pp. 189–196, 2016.
- [26] B. Becher, A. Schlitzer, J. Chen, F. Mair, H. R. Sumatoh, K. W. W. Teng, D. Low, C. Ruedl, P. Riccardi-Castagnoli, M. Poidinger, M. Greter, F. Ginhoux, and E. W. Newell, "High-dimensional analysis of the murine myeloid cell system," *Nature Immunology*, vol. 15, no. 12, pp. 1181–1189, 2014.
- [27] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond Euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.
- [28] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2014, pp. 1724–1734.
- [29] A. Mead, "Review of the development of multidimensional scaling methods," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 41, no. 1, pp. 27–39, 1992.
- [30] Y.-H. Fua, M. O. Ward, and E. A. Rundensteiner, "Hierarchical parallel coordinates for exploration of large datasets," in *Proceedings of the Conference on Visualization*, 1999, pp. 43–50.
- [31] J.-J. Huang, G.-H. Tzeng, and C.-S. Ong, "Multidimensional data in multidimensional scaling using the analytic network process," *Pattern Recognition Letters*, vol. 26, no. 6, pp. 755–767, 2005.
- [32] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform manifold approximation and projection for dimension reduction," 2018, arXiv:1802.03426[stat.ML].
- [33] G. E. Hinton and S. T. Roweis, "Stochastic neighbor embedding," in *Advances in Neural Information Processing Systems*, vol. 15, 2003, pp. 857–864.
- [34] L. van der Maaten and G. E. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.
- [35] L. van der Maaten, "Accelerating t-SNE using tree-based algorithms," *Journal of Machine Learning Research*, vol. 15, pp. 3221–3245, 2014.
- [36] N. Pezzotti, T. Höllt, B. Lelieveldt, E. Eisemann, and A. Vilanova, "Hierarchical stochastic neighbor embedding," *Computer Graphics Forum*, vol. 35, no. 3, pp. 21–30, 2016.
- [37] N. Pezzotti, B. P. F. Lelieveldt, L. van der Maaten, T. Höllt, E. Eisemann, and A. Vilanova, "Approximated and user steerable tSNE for progressive visual analytics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 7, pp. 1739–1752, 2017.
- [38] G. C. Linderman, M. Rachh, J. G. Hoskins, S. Steinerberger, and Y. Kluger, "Fast interpolation-based t-SNE for improved visualization of single-cell RNA-seq data," *Nature Methods*, vol. 16, no. 3, pp. 243–245, 2019.
- [39] E. D. Amir, K. L. Davis, M. D. Tadmor, E. F. Simonds, J. H. Levine, S. C. Bendall, D. K. Shenfeld, S. Krishnaswamy, G. P. Nolan, and D. Pe'er, "viSNE enables visualization of high dimensional single-cell data and reveals phenotypic heterogeneity of leukemia," *Nature Biotechnology*, vol. 31, no. 6, pp. 545–552, 2013.
- [40] W. M. Abdelmoula, B. Balluff, S. Englert, J. Dijkstra, M. J. T. Reinders, A. Walch, L. A. McDonnell, and B. P. F. Lelieveldt, "Data-driven identification of prognostic tumor subpopulations using spatially mapped t-SNE of mass spectrometry imaging data," *Proceedings of the National Academy of Sciences*, vol. 113, no. 43, pp. 12 244–12 249, 2016.
- [41] V. van Unen, T. Höllt, N. Pezzotti, N. Li, M. J. T. Reinders, E. Eisemann, F. Koning, A. Vilanova, and B. P. F. Lelieveldt, "Visual analysis of mass cytometry data by hierarchical stochastic neighbour embedding reveals rare cell types," *Nature Communications*, vol. 8, no. 1, p. 1740, 2017.

- [42] N. Pitsianis, D. Floros, A.-S. Iliopoulos, and X. Sun, “SG-t-SNE-II: Swift neighbor embedding of sparse stochastic graphs,” *Journal of Open Source Software*, vol. 4, no. 39, p. 1577, 2019.
- [43] L. van der Maaten, “Learning a parametric embedding by preserving local structure,” in *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, vol. 5, 2009, pp. 384–391.
- [44] M. Muja and D. G. Lowe, “Scalable nearest neighbor algorithms for high dimensional data,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2227–2240, 2014.
- [45] —, “Fast approximate nearest neighbors with automatic algorithm configuration,” in *International Conference on Computer Vision Theory and Application*, 2009, pp. 331–340.
- [46] T. U. Consortium, “Reorganizing the protein space at the Universal Protein Resource (UniProt),” *Nucleic Acids Research*, vol. 40, no. D1, pp. D71–D75, 2011.
- [47] M. Su, H. Zhang, X. Duy, and Q. Dai, “A novel stochastic-encryption-based P2P digital rights management scheme,” in *IEEE International Conference on Communications*, 2015, pp. 5541–5545.
- [48] T. Liu and X. Sun, “Density-based detection of community hierarchy from graph description,” manuscript in preparation.
- [49] 10x Genomics, “Transcriptional profiling of 1.3 million brain cells with the Chromium single cell 3’ solution,” *LIT000015 Chromium Million Brain Cells Application Note*, 2017.
- [50] G. X. Y. Zheng, J. M. Terry, P. Belgrader, P. Ryvkin, Z. W. Bent, R. Wilson, S. B. Ziraldo, T. D. Wheeler, G. P. McDermott, J. Zhu, M. T. Gregory, J. Shuga, L. Montesclaros, J. G. Underwood, D. A. Masquelier, S. Y. Nishimura, M. Schnall-Levin, P. W. Wyatt, C. M. Hindson, R. Bharadwaj, A. Wong, K. D. Ness, L. W. Beppu, H. J. Deeg, C. McFarland, K. R. Loeb, W. J. Valente, N. G. Ericson, E. A. Stevens, J. P. Radich, T. S. Mikkelsen, B. J. Hindson, and J. H. Bielas, “Massively parallel digital transcriptional profiling of single cells,” *Nature Communications*, vol. 8, p. 14049, 2017.
- [51] E. A. Abbott, *Flatland: A Romance of Many Dimensions*. London, UK: Seeley & Co, 1884.
- [52] J. Barnes and P. Hut, “A hierarchical $O(N \log N)$ force-calculation algorithm,” *Nature*, vol. 324, no. 6096, pp. 446–449, 1986.
- [53] L. Greengard and V. Rokhlin, “A fast algorithm for particle simulations,” *Journal of Computational Physics*, vol. 73, no. 2, pp. 325–348, 1987.
- [54] X. Sun and N. P. Pitsianis, “A matrix version of the fast multipole method,” *SIAM Review*, vol. 43, no. 2, pp. 289–300, 2001.
- [55] B. Zhang, J. Huang, N. P. Pitsianis, and X. Sun, “recFMM: Recursive parallelization of the adaptive fast multipole method for Coulomb and screened Coulomb interactions,” *Communications in Computational Physics*, vol. 20, no. 2, pp. 534–550, 2016.
- [56] Y. Zhang, J. Liu, E. Kultursay, M. Kandemir, N. Pitsianis, and X. Sun, “Automatic parallel code generation for nuFFT data translation on multicores,” *Journal of Circuits, Systems, and Computers*, vol. 21, no. 2, p. 1240004, 2012.
- [57] D. Floros, T. Liu, N. Pitsianis, and X. Sun, “Sparse dual of the density peaks algorithm for cluster analysis of high-dimensional data,” in *IEEE High Performance Extreme Computing Conference*, 2018.
- [58] J. Mellor-Crummey, D. Whalley, and K. Kennedy, “Improving memory hierarchy performance for irregular applications using data and computation reorderings,” *International Journal of Parallel Programming*, vol. 29, no. 3, pp. 217–247, 2001.
- [59] A. Buluç, J. T. Fineman, M. Frigo, J. R. Gilbert, and C. E. Leiserson, “Parallel sparse matrix-vector and matrix-transpose-vector multiplication using compressed sparse blocks,” in *Proceedings of Annual Symposium on Parallelism in Algorithms and Architectures*, 2009, pp. 233–244.
- [60] R. D. Blumofe, C. F. Joerg, B. C. Kuszmaul, C. E. Leiserson, K. H. Randall, and Y. Zhou, “Cilk: An efficient multithreaded runtime system,” *Journal of Parallel and Distributed Computing*, vol. 37, no. 1, pp. 55–69, 1996.
- [61] J. D. McCalpin, “Memory bandwidth and machine balance in current high performance computers,” *IEEE Computer Society Technical Committee on Computer Architecture (TCAA) Newsletter*, 1995.
- [62] D. A. Patterson and J. L. Hennessy, *Computer Organization and Design: The Hardware/Software Interface*, fifth edition ed., ser. The Morgan Kaufmann Series in Computer Architecture and Design. Amsterdam; Boston: Elsevier/Morgan Kaufmann, 2014.
- [63] A. Yzelman and R. H. Bisseling, “Cache-oblivious sparse matrix-vector multiplication by using sparse matrix partitioning methods,” *SIAM Journal on Scientific Computing*, vol. 31, no. 4, pp. 3128–3154, 2009.